

Semi-automatic rural land cover classification from high-resolution remote sensing images

Ph.D. thesis

Classification semi-automatique du terrain en zone rurale par télédétection à haute résolution

Thèse de doctorat

Roger Trias-Sanz

2006

Université Paris 5 – René Descartes, UFR de Mathématiques et Informatique

Thèse présentée par Roger TRIAS SANZ en vue de l'obtention du grade de Docteur de l'Université Paris 5 – René Descartes dans la discipline de Sciences de la Vie et de la Matière, spécialité Mathématiques-Informatique,

dirigée par Pr. Georges STAMON et Dr. Jean LOUCHET

soutenue le 24 mars 2006 devant le jury composé de

Dr. Matthieu CORD	rapporteur
Pr. Josef KITTLER	examineur
Dr. Jean LOUCHET	directeur de thèse
Pr. Ferran MARQUÉS ACOSTA	rapporteur
Dr. Marc PIERROT-DESEILLIGNY	examineur
Pr. Georges STAMON	directeur de thèse

Numéro attribué par la bibliothèque:

Copyright © 2006 Roger Trias Sanz.

Aerial images in figures 3.5, 3.6, 3.22, 4.1, 4.3 (haute), 5.6, 5.8, 5.9, 5.10, 5.11, 5.20 (top left), 5.21 (left), 6.2, 6.3 (top), 6.5 (left), and A.1, and in part of the cover and back cover are Copyright © Institut Géographique National, used with permission.

Cette création est mise à disposition selon le Contrat Paternité - Pas d'Utilisation Commerciale - Pas de Modification disponible en ligne <http://creativecommons.org/licenses/by-nc-nd/2.0/fr/> ou par courrier postal à Creative Commons, 543 Howard Street, 5th Floor, San Francisco, California, 94105, États-Unis.

This work is licensed under the Creative Commons Attribution-NonCommercial-NoDerivs License under French jurisdiction. To view a copy of this license, visit <http://creativecommons.org/licenses/by-nc-nd/2.0/fr/> or send a letter to Creative Commons, 543 Howard Street, 5th Floor, San Francisco, California, 94105, USA.

```
<rdf:RDF xmlns="http://web.resource.org/cc/" xmlns:dc="http://purl.org/dc/elements/1.1/" xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
<Work rdf:about=""> <dc:title>Semi-automatic rural land cover classification from high-resolution remote sensing images</dc:title> <dc:date>2006</dc:date>
<dc:creator><Agent><dc:title>Roger TRIAS SANZ</dc:title></Agent></dc:creator>
<dc:rights><Agent><dc:title>Roger TRIAS SANZ</dc:title></Agent></dc:rights>
<dc:type rdf:resource="http://purl.org/dc/dcmitype/Text" /> <license rdf:resource="http://creativecommons.org/licenses/by-nc-nd/2.0/fr/" /> </Work>
<License rdf:about="http://creativecommons.org/licenses/by-nc-nd/2.0/fr/"> <permits rdf:resource="http://web.resource.org/cc/Reproduction" />
<permits rdf:resource="http://web.resource.org/cc/Distribution" /> <requires rdf:resource="http://web.resource.org/cc/Notice" />
<requires rdf:resource="http://web.resource.org/cc/Attribution" /> <prohibits rdf:resource="http://web.resource.org/cc/CommercialUse" /> </License>
</rdf:RDF>
```

BIB_{TEX} document entry / fiche bibliographique BIB_{TEX}:

```
@PhdThesis{TriasSanz:PhD,
  author = {Trias Sanz, Roger},
  title = {Semi-automatic rural land cover classification
           from high-resolution remote sensing images},
  school = {Universit{\`e} Paris 5},
  year = 2006,
  address = {Paris, France},
  month = mar
}
```

ROGER TRIAS-SANZ, *Semi-automatic rural land cover classification from high-resolution remote sensing images*, PhD thesis, Université Paris 5, Paris, France, March 2006.

Agraïments

Al llarg dels tres anys i mig que ha durat aquesta tesi, moltes persones m'han ajudat, d'una manera o d'una altra, en la seva realització.

Gràcies, en primer lloc, a la meva família, i als meus amics, especialment a la Julia, companya de pis per uns mesos, i a la Young.

Gràcies a en Georges Stamon i en Jean Louchet, directors de tesi, als meus responsables de tesi a l'IGN, en Marc Pierrot-Deseilligny i en Didier Boldo, i a en Nicolas Loménie, pels seus consells, i pels seus comentaris i indicacions sobre els diversos esborranys de tesi que han llegit.

Gràcies també a en Josef Kittler, en Ferran Marqués Acosta i en Matthieu Cord, que han acceptat formar part del tribunal de tesi.

Gràcies als membres del laboratori MATIS, tant a nivell personal, per la seva amistat, com a nivell professional, per totes les vegades que m'han ajudat a resoldre dubtes i problemes, i per la riquesa científica de les converses que hi he mantingut. Especialment, gràcies a en François Boyero per tota la seva assistència en temes pràctics. També a en Laurent Guigues, per les nostres converses científiques, i a en Franck Jung, per ajudar-me en l'obtenció d'imatges i dades de cadastre.

Gràcies a en Michel Deshayes; la meua visita al seu laboratori a Montpellier em va aportar pistes interessants per a la tesi. Gràcies a diversos professors que he tingut en etapes d'estudi anteriors, l'Oriol Serra, en Francesc Comellas, en José Mariño, en Michael R. Hansen i d'altres, que per la seva qualitat personal i la de les seves classes m'han encoratjat a continuar els estudis.

Finalment, gràcies a totes les altres persones que han estat prop meu aquests anys a París.

To see the Universe in a grain of sand
And heaven in a wild flower,
Hold infinity in the palm of your hand
And eternity in an hour.

—William Blake, "Auguries of Innocence"

Contents

Résumé	xv
1 Introduction	1
2 Literature review	7
2.1 Colour, texture, and shape	7
2.2 Segmentation	13
2.3 Registration	16
2.4 Classification	18
2.5 Data fusion, decision	27
2.6 Optimization methods	28
2.7 Complete systems	28
3 Image segmentation	29
3.1 Hierarchical segmentation	30
3.2 Segmentation energy	35
3.3 Evaluating the quality of a multiscale segmentation	47
3.4 Choosing the best segmentation parameters	49
3.5 Conclusion	78
4 Registration of external terrain partitions	83
4.1 Introduction	83
4.2 Image over-segmentation and input graphs	86
4.3 Edge-based registration algorithm	88
4.4 Region-based registration algorithm	95
4.5 Experiments and evaluation	97
4.6 Homogeneity tests	120
4.7 Discussion and conclusion	122
5 Classification	123
5.1 Introduction	123

5.2	Flat models	125
5.3	Nested models	134
5.4	Classification of nested models	138
5.5	Model estimation	149
5.6	Implementation details	155
5.7	Evaluation of the model estimation	162
5.8	Classification into forest and non-forest	180
5.9	Evaluation	182
5.10	Conclusion	215
6	Texture orientation and period estimation	219
6.1	Introduction	219
6.2	Algorithm	220
6.3	Watershed-based extrema detection	227
6.4	Evaluation	229
6.5	Conclusion	232
7	Conclusion	233
7.1	Land cover analysis	233
7.2	Contributions	234
7.3	Perspectives for future research	235
7.4	Concluding remarks	236
A	Texture descriptors and transformed colour spaces	239
A.1	Colour spaces	239
A.2	Texture parameters	247
B	Image segmentation tests	269
C	Publications	309
D	Bibliography	311
E	What it is all about: executive summary for non-scientists	337

Abstract

Much research has been done on the subject of automatically determining land use and land cover in rural areas from images. However, error rates have always been too high for industrial application.

The French National Mapping Agency (Institut Géographique National, IGN), is interested in automatic land cover classification for the purpose of speeding up production of high-resolution topographic maps. The context at IGN is, however, slightly different from that at most other research into automatic land cover classification: First, very high spatial resolution digital images are available (50 cm per pixel), but these images have a low spectral resolution (red, green, blue, and in some cases near-infrared channels), which precludes the use of standard hyperspectral classification techniques. Second, cadastre data is available; this data gives a very rough indication of field position. Finally, IGN's interest stems from practical goals: to reduce the amount of time spent by human photo-interpreters doing manual classifications. Therefore, IGN is not especially interested in obtaining a medium-quality land cover classification of the whole region of interest but would rather have a very high quality classification of only a portion of the terrain; the first one would have to be double-checked in full by a photo-interpreter, whereas the second one would not need further human verification, so that photo-interpreters would be able to concentrate on the remaining areas.

In this Ph.D. thesis, I present a complete image analysis system which, from high-resolution 3 or 4-channel digital images (50 cm, colour and optionally near infrared), and using the cadastre database, segments the images into agriculturally-homogeneous regions (fields, forests, vines, and so on), and classifies these regions, labelling each classified region with a confidence measure which indicates the system's confidence in each classification, and which can be used to filter out regions that are more likely to have been incorrectly classified.

The process starts with a hierarchical segmentation, using a colour space, texture parameters, and shape criteria adapted to the problem of segmenting agricultural regions. This segmentation is used to register the cadastre onto the image, giving large, usually homogeneous regions. Through this registration, the system can also be used to update older classifications. Then, each of these registered cadastre regions—or, if cadastre data is not available, small regions obtained by watershed segmentation—is classified using novel probabilistic per-region classification algorithms which, unlike traditional per-pixel algorithms, do not produce salt-and-pepper noise, and which also output a classification confidence measure for each classified region. These classification algorithms are supervised, and need to be trained beforehand with a ground truth defined by a photo-interpreter.

As an end product we get an image segmentation into classified agriculturally-homogeneous regions, and confidence measures for each part of the segmentation, which a human photo-interpreter can use to correct these results or to concentrate limited available time into the most likely errors.

Keywords: classification; image analysis; remote sensing; land cover; vegetation; crop classification; cadastre registration.

Résumé

De nombreux travaux d'analyse d'image ont été et continuent d'être menés pour tenter de déterminer automatiquement l'occupation du sol en milieu rural, mais aucun n'a encore produit de résultats assez fiables pour être exploitables dans une chaîne de production industrielle.

L'Institut Géographique National français (IGN) s'intéresse à la classification automatique du terrain pour accélérer la production de cartes topographiques à grande échelle. Le contexte à l'IGN est, cependant, différent de celui de la plupart des recherches dans la classification automatique : D'abord, on dispose d'images numériques à très haute résolution spatiale (50 cm par pixel), mais ces images sont à faible résolution spectrale (canaux rouge, vert, bleu et, dans certains cas, proche-infrarouge), ce qui rend impossible l'utilisation de techniques classiques de classification de données hyperspectrales. De plus, on dispose des données cadastrales, qu'on peut utiliser pour obtenir une information grossière de la position des champs. Finalement, l'IGN n'est pas particulièrement intéressé à obtenir des classifications automatiques de qualité moyenne sur l'ensemble du territoire, classifications qui devraient être vérifiées, à grand coût en temps, par un photo-interprète. Par contre, l'IGN voudrait obtenir une classification de très haute qualité, même si c'est sur seulement une partie du territoire, car cette classification n'aurait pas à être vérifiée manuellement, et les photo-interprètes pourraient concentrer leur temps à classer la partie restante.

Dans cette thèse, je présente une chaîne d'analyse d'image qui, à partir d'images numériques à haute résolution et à trois ou quatre canaux (50 cm, couleur et, dans certains cas, proche infrarouge), mais aussi en m'appuyant sur le parcellaire cadastral, rend une segmentation des images en parcelles agraires (champs, forêts, vignes, . . .), et une classification de celles-ci, avec une très haute fiabilité, et attribue à chaque segment classifié une mesure qui indique la confiance que le système a en cette classification.

Une phase initiale de segmentation hiérarchique de l'image, qui utilise un espace de couleur, des paramètres de texture, et des critères de forme adaptés à la segmentation de parcelles agraires, permet de recalculer le cadastre sur l'image, produisant des régions grandes et en général homogènes. Ce recalage permet aussi d'utiliser le système pour la mise à jour de classifications anciennes. Ensuite, chacune de ces régions de cadastre recalées —ou, si les données cadastrales ne sont pas disponibles, des petites régions issues d'une segmentation par ligne de partage des eaux— est classifiée au moyen de nouveaux algorithmes probabilistes de classification par régions qui, à la différence des algorithmes classiques par pixels, ne produisent pas du bruit poivre-et-sel, et qui génèrent aussi une mesure de confiance pour chaque région classifiée. Les régions classifiées avec une trop faible confiance peuvent ensuite, selon les besoins de l'application, être rejetées et classifiées manuellement par des photo-interprètes. Ces algorithmes doivent être entraînés auparavant à partir d'une vérité terrain définie manuellement.

À la fin on obtient une segmentation de l'image en parcelles agraires homogènes, une classification de celles-ci, et des indicateurs de confiance sur chaque partie de la segmentation, ce qui permet à un photo-interprète de réaliser les corrections nécessaires et de concentrer son temps limité sur les zones qui plus vraisemblablement contiennent des erreurs.

Mots clés : classification ; analyse d'images ; télédétection ; occupation du sol ; végétation ; classification de cultures ; recalage du cadastre.

Resum

Durant els darrers temps s'han esmerçat molts esforços en recerca sobre la determinació automàtica del tipus i ocupació del sol a partir d'imatges aèries i de satèl·lit en zones rurals. No obstant, les taxes d'error han estat sempre massa altes per a aplicacions industrials.

L'Institut Cartogràfic Nacional francès (Institut Géographique National, IGN) s'interessa a la classificació automàtica del terreny per tal d'accelerar la producció dels seus mapes topogràfics a gran escala. El context a l'IGN, però, és lleugerament diferent: Primer, l'IGN disposa d'imatges digitals de molt alta resolució espacial (50 cm per píxel), però de baixa resolució espectral (canals vermell, verd, blau i, en alguns casos, infraroig proper), cosa que impedeix utilitzar tècniques clàssiques d'anàlisi hiperespectral. A més, disposa de dades cadastrals, que donen una indicació aproximada de la localització dels camps. Finalment, a l'IGN, guiat per consideracions pràctiques, no li interessa especialment obtenir classificacions de qualitat mitjana sobre tota la zona de treball, perquè aquesta classificació hauria de ser revisada per un fotointèrpret cosa que costaria molt de temps, sinó que prefereix una classificació de molt alta qualitat sobre només una part del terreny, i deixar als fotointèrprets la tasca de classificar manualment la resta.

En aquesta tesi de doctorat, presento un sistema complet d'anàlisi d'imatges que, a partir d'imatges digitals d'alta resolució a 3 o 4 canals (50 cm, color i opcionalment infraroig proper), i utilitzant dades cadastrals, segmenta les imatges en zones agriculturalment homogènies (camps, boscs, vinyes, ...) i les classifica, indicant per a cada regió una mesura de la confiança que el sistema té en la classificació, cosa que permet de rebutjar les regions que més probablement han estat mal classificades.

El procés comença amb una segmentació jeràrquica, on s'utilitzen espais de color, paràmetres de textura, i criteris de forma adaptats al problema de segmentar zones agrícoles. Aquesta segmentació permet de registrar el cadastre sobre l'imatge, donant grans regions sovint homogènies. El procés de registració permet també d'utilitzar el sistema per a posar al dia una classificació antiga. Aleshores, cada parcel·la cadastral registrada —o, si no es disposa de dades cadastrals, cadascuna de les regions petites obtingudes per una segmentació per watershed— es classifica amb algorismes probabilístics de classificació per regions desenvolupats en aquesta tesi. Aquests algorismes, a diferència dels clàssics que operen píxel per píxel, no produeixen soroll sal-i-pebre, i calculen una mesura de confiança per cada regió classificada. Les regions amb baixa confiança poden rebutjar-se i fer-se classificar a mà per un fotointèrpret si l'aplicació ho demana. Aquests algorismes són supervisats, i s'han d'entrenar amb una veritat terreny feta a mà abans de la classificació.

Com a resultat obtenim una segmentació de l'imatge en regions classificades i homogènies agriculturalment, i una mesura de confiança per cada regió, que un fotointèrpret pot utilitzar per a corregir els resultats o per a concentrar el seu temps limitat en les zones amb més risc d'error.

Paraules clau: classificació; anàlisi d'imatge; teledetecció; ocupació del terreny; vegetació; classificació de cultius; registració del cadastre.

Résumé étendu en français

Dans une institution cartographique comme l'Institut Géographique National français, l'IGN, l'une des activités principales est la production de cartes à haute résolution de qualité.

Mais, comment ces cartes sont-elles fabriquées?

Auparavant, pour chaque carte il y avait plusieurs planches d'un matériel plastique, une pour chaque couleur. Pour certaines couleurs très utilisées, il pouvait y avoir une planche par *thème*, ou type d'information. Par exemple, il y en avait une pour le bleu, une pour le vert, une pour la toponymie en noir, une pour les bâtiments en noir, et bien d'autres. Ces planches étaient appelées *planches mères*. Ces planches étaient composées d'une couche transparente et d'une couche opaque. La couche opaque était découpée avec des outils spéciaux pour produire un *masque*. Moyennant un procédé d'impression par couches, les figures créées par les lignes et les polygones découpés des planches s'imprimaient sur une grande feuille en papier dans la couleur correspondante. Après l'impression de toutes les planches, on obtenait une carte complète sur la feuille. Régulièrement, des agents cherchaient des changements sur le terrain: des nouveaux bâtiments, des routes dédoublées, des forêts qui seraient devenues des champs, et ainsi de suite. Ces changements étaient enregistrés sur les planches mère en découpant du matériel opaque, ou, pour défaire une découpe antérieure, en collant de petites pièces opaques sur la planche.

Actuellement, bien sûr, toutes ces données sont stockées en format numérique dans des *bases de données géographiques*. Les éléments géographiques, comme les bâtiments, les toponymes, ou les routes, sont enregistrés comme des objets dans ces bases de données. Les informations pour tout le territoire géré par une agence cartographique peuvent être stockées dans une seule base de données, au lieu de devoir découper ce territoire en planches de taille physiquement gérable. Les ajouts, suppressions, et modifications coûtent seulement quelques clics de souris, et il n'est plus nécessaire de découper et recoller des morceaux de plastique. Dans la plupart des cas, des jeux de données différents sont maintenus pour chaque échelle de représentation, car on ne peut pas produire une carte à, disons, une échelle de 1:50000 tout simplement en prenant une carte plus détaillée au 1:25000 et la réduisant géométriquement: il y a des problèmes concernant le niveau de détail acceptable à chaque échelle, et la manière dont l'information doit être simplifiée. Dans d'autres cas, sont stockées seulement les données pour l'échelle la plus précise, et des algorithmes de simplification ou *généralisation* automatiques [Bar04, Duc03, DC03, Rua02, Duc01, Rua01, RM97, RP96] convertissent ces données dans des échelles moins fines.

Cependant, ces données sont encore ajoutées dans la base de données, et mises à jour, de façon manuelle. Des photointerprètes, qui sont des personnes expertes à comprendre le terrain à partir d'images prises d'un avion ou d'un satellite, examinent des images de la zone d'intérêt et, à l'aide de rapports des topographes sur le terrain, mettent à jour la base de données. Même si ce procédé est significativement plus rapide que la découpe de planches en plastique, il reste cependant très cher et lent. Ainsi, les cartes topographiques au 1:25000 de l'IGN ont, en moyenne, dix ans.

Il y a beaucoup de projets de recherche à l'IGN et ailleurs qui visent à automatiser totalement ou partiellement l'interprétation des images aériennes pour, dans un premier temps, remplir les bases de données géographiques, et, plus tard, les maintenir à jour au fur et à mesure que des nouvelles prises de vues de chaque zone deviennent disponibles. Par exemple, il y a eu des travaux sur la détection et le suivi automatique de routes [YC00, YCWZ01, Boi00, BCA99], et beaucoup de travaux sont actuellement en cours pour obtenir des modèles 3D de bâtiments [PTJ00, TD02, NN01, VD00, CPDJS02, BPPD04, BPPDH04, DPDPC04], et de superstructures de bâtiment.

Cette thèse de doctorat, qui vise l'automatisation partielle de l'extraction d'information sur des zones végétales à partir d'images aériennes, s'inscrit dans cet effort de recherche. Le but est d'obtenir des algorithmes qui, à partir d'images aériennes à haute résolution, peuvent automatiquement localiser les zones de végétation dans les images, en obtenir les limites précisément, et les classer selon le type de végétation. Les types de végétation communs, dans des applications cartographiques, sont les forêts, les champs, les vergers, et les vignes. Le système reconnaît aussi d'autres objets surfaciques qui ne sont pas végétaux, comme des zones d'eau ou de sol nu. Tous ces types de couverture du sol s'appellent *classes* ou *types de terrain*.

Cette recherche a un but clairement applicatif: l'accélération de la production de cartes. En tant que travail applicatif, nous n'avons pas hésité à prendre des raccourcis et à explorer des chemins peut-être moins beaux d'un point de vue formel ou mathématique, si ces alternatives donnent des meilleurs résultats que des algorithmes supérieurs en théorie. Nous n'avons cependant pas oublié que ce n'est pas parce que nous adoptons un point de vue applicatif que les algorithmes doivent être des "usines à gaz" mal conçues, et que, souvent, c'est l'algorithme élégant d'un point de vue théorique qui est aussi le plus performant en pratique, ou s'avère tel à plus long terme. Cette approche appliquée a influencé les recherches en trois autres aspects. D'abord, toutes les données additionnelles disponibles à l'IGN dans un contexte de production ont été utilisées, quand ceci s'est avéré utile; par exemple, nous avons utilisé les données cadastrales qui fournissent une bonne indication de la position approximative des champs. Ensuite, nous avons développé des algorithmes qui ont des temps de calcul raisonnables. Et finalement, et le plus important, nous avons conçu des algorithmes qui permettent de faire des classifications semi-automatiques. Ceci parce que l'IGN préfère avoir une classification automatique sur seulement une partie du territoire, mais de très haute qualité et qui n'aura pas à être révisée, que d'avoir une classification automatique sur tout le territoire, mais d'une qualité moindre. Dans le premier cas, les photo-interprètes peuvent concentrer leur temps dans les zones que le système a préféré laisser sans classification (parce qu'il les a jugées trop difficiles). Dans le deuxième cas, ils auraient à vérifier, et corriger, tout le territoire, à grand coût. Pour cela, les algorithmes de classification présentés fournissent, en plus d'une carte de classification, des indicateurs de confiance pour chaque zone. Selon le niveau de qualité acceptable, les zones qui ont été classifiées avec une confiance en dessous d'un certain seuil peuvent être ignorées, pour être ensuite classifiées à la main par un photo-interprète.

Plus précisément, ce système d'analyse d'images aura, en entrée, le cadastre vectorisé sous forme de graphe, et un ensemble d'images d'une zone rurale, prises d'un avion, à haute résolution et géoréférencées. Les images seront des orthophotographies: elles auront été transformées de la perspective conique donnée par les caméras de prise de vue, en une perspective parallèle avec les rayons lumineux dans la direction verticale. Les images seront à une résolution de 50 cm par pixel, et auront quatre canaux, rouge, vert, bleu et proche infrarouge, avec correction radiométrique des effets atmosphériques. La thèse inclut aussi des tests avec des images argentiques à trois canaux (rouge, vert et bleu), à 80 cm par pixel. Le système sera censé travailler sur les zones rurales, mais pas sur les zones urbaines ou suburbaines, même pas pour classifier la végétation qui se trouverait dans ces zones. Comme d'autres travaux

de recherche s'occupent de la classification de la végétation en zone urbaine, et qu'il y a déjà des travaux sur les bâtiments et les routes, nous utiliserons le cadastre pour connaître aussi la position des bâtiments et des routes, pour ne pas classer ces zones. Finalement, ce système pourra aussi être utilisé pour la mise à jour de cartes de classification du sol anciennes, lorsque de nouvelles prises de vue aériennes deviennent disponibles. Dans ce cas, au lieu du cadastre vectorisé, nous donnerions comme entrée les limites de la classification ancienne, pour obtenir des positions approximatives des champs.

Les méthodes de classification développées sont du type *supervisé*. C'est-à-dire que le système doit passer par une phase d'apprentissage dans laquelle est fournie une *vérité terrain*, une référence qui contient des images avec des régions de type de terrain connu, ce qui lui permet de reconnaître les caractéristiques discriminantes pour chaque type de terrain, pour que, dans la phase de classification, le système puisse indiquer le type de terrain d'une nouvelle zone, différente des zones de la référence, mais suffisamment similaire.

Le système analysera les données d'entrée et produira, en sortie, une segmentation des images d'entrée en régions homogènes en culture. L'homogénéité en culture est définie en termes de classes cartographiques végétales, c'est-à-dire: des forêts, des champs, des vergers, etc., et aussi en d'autres propriétés comme la couleur, ces dernières permettant de séparer des champs de la même culture, mais dans des états de croissance différents. Une région homogène en culture, donc, serait entièrement composée d'un seul de ces types de terrain, et aurait une seule couleur. Il ne s'agira pas, en principe, d'une segmentation minimale: il est possible d'obtenir, en sortie, deux régions adjacentes qui contiennent la même culture, mais ce problème peut être facilement résolu, si besoin, par un post-traitement simple. En plus du type de terrain attribué à chaque région de l'image en sortie, le système donnera aussi un indicateur de la confiance qu'il a en cette attribution, séparément pour chaque région, ce qui permettra à l'utilisateur de se concentrer seulement sur les zones sur lesquelles le système a plus de difficultés.

Organisation de ce document

La séquence d'analyse se divise en trois étapes, segmentation, recalage et classification. Chacune est décrite dans un chapitre séparé. Un chapitre supplémentaire décrit une méthode d'estimation des orientations et périodes des textures, ce qui peut améliorer la classification dans certains cas.

Cette thèse est modulaire, et chacune des étapes comprend plusieurs contributions scientifiques qui peuvent être utilisées séparément. Le document est organisé selon l'ordre dans lequel les étapes doivent être exécutées.

Après un étude bibliographique au chapitre 2, le chapitre 3 propose plusieurs indicateurs de texture et plusieurs espaces de couleur, dérivés de l'état de l'art et adaptés à notre problème. Nous étudions la configuration et les valeurs optimales pour la segmentation des images d'entrée avec l'algorithme de segmentation hiérarchique de Guigues [Gui04], et nous donnons une courte introduction à la segmentation multiéchelles. Le chapitre 4 présente deux algorithmes nouveaux pour le recalage d'un graphe sur une image, permettant de recalibrer les données cadastrales. Ses comportements et performances sous différentes conditions sont étudiés en détail. Ensuite, dans le chapitre 5 les bases de la classification pixel par pixel sont rappelées brièvement, et plusieurs nouveaux algorithmes de classification région par région sont présentés, ainsi qu'un nouveau modèle de probabilités, appelé "composé". Nous étudions l'applicabilité de ce nouveau modèle aux données qui nous concernent, et évaluons quantitativement la performance des algorithmes de classification. Ce chapitre inclut la description d'une application particulièrement intéressante, la classification du terrain en zones de forêt et de non-forêt, pour laquelle ces algorithmes donnent de très bons résultats (section 5.8). Dans

le chapitre 5 il apparaît que certaines classes sont facilement confondues, et la radiométrie et les indicateurs de texture du chapitre 3 ne suffisent pas à les distinguer; pour résoudre ce problème le chapitre 6 présente une nouvelle méthode pour obtenir une estimation précise des orientations et périodes de textures périodiques à forme et nombre de canaux arbitraires.

Chacun de ces chapitres, segmentation, recalage, classification, et estimation d'orientation, contient des évaluations quantitatives complètes en utilisant des images réelles de caractéristiques similaires à celles qui seront utilisées en production. Les résultats de ces évaluations sont donc significatifs, et les évaluations des chapitres 5 et 6 peuvent être vues comme des évaluations globales du système. Des conclusions ainsi que des perspectives pour la continuation des recherches dans cette thématique sont incluses dans le chapitre 7. Un résumé destiné à un public non scientifique dans l'annexe E clarifie l'utilité pratique de ces recherches.

Segmentation (chapitre 3)

La première opération à faire sur les images est une segmentation, qui donnera une partition de l'image en régions nécessaire pour les étapes ultérieures.

Les segmentations d'images sont faites souvent comme un prétraitement pour des systèmes d'interprétation d'images, par exemple dans certains systèmes de classification de couverture du terrain ou d'usage du terrain. L'algorithme de segmentation est utilisé tout en espérant qu'il divisera l'image en régions sémantiquement significatives, c'est à dire, en objets, qui seront identifiés dans les étapes suivantes.

La plupart des méthodes de segmentation prennent un paramètre (souvent un seuil sur la dissimilitude entre régions adjacentes) et fournissent en sortie une partition de l'image. Ceci correspond à une analyse de l'image à une seule échelle: des seuils bas donnent des segmentations avec des petites régions et beaucoup de détail, alors que des seuils plus hauts conservent seulement les régions les plus saillantes.

Le problème est que, comme il est connu depuis les débuts de l'analyse d'images [Mar82], des structures et des objets significatifs apparaissent pour plusieurs valeurs du paramètre de contrôle, et pas pour une seule valeur. En d'autres termes, il est possible de trouver des objets significatifs à plusieurs échelles d'analyse. Parfois une même zone de l'image a des interprétations différentes selon l'échelle: par exemple, en allant des échelles fines aux grossières, un même pixel peut faire partie d'une tuile, d'un toit, d'une maison, ou d'un village. Aussi, même dans les cas où il existe une seule échelle d'analyse correcte, celle-ci peut souvent seulement être déterminée après une interprétation de l'image qui requiert une analyse de haut niveau, qui sera réalisée après, pas avant, la segmentation elle-même. Pour s'affranchir de ce problème, plusieurs auteurs ont proposé des méthodes de segmentation *multiéchelles* [JM92, KLM94, SG00, GLMC03b]. Ces méthodes analysent l'image à plusieurs échelles en même temps. En sortie elles ne fournissent pas une seule partition, mais une hiérarchie de partitions, ou une autre structure de données qui capture différentes partitions pour différentes échelles d'analyse.

Une autre question à résoudre est celle des données à utiliser en entrée de l'algorithme de segmentation. Les données radiométriques fournies par le capteur (par exemple, les canaux rouge, vert et bleu) sont les candidats les plus évidents, mais il est possible que des transformations de ces données donnent de meilleurs résultats. Il s'agit de concilier le fait que les algorithmes de segmentation cherchent à trouver des régions homogènes selon leur radiométrie —ou d'autres données en entrée à l'algorithme— alors que l'utilisateur cherche à obtenir des régions homogènes selon sa sémantique. Autrement dit, l'importance d'une limite peut se définir du point de vue de l'application —plus deux cultures adjacentes sont différentes, plus la limite entre les deux est importante— mais aussi du point de vue de l'image —si une limite est trouvée

dans l'analyse de l'image à un niveau de détail plus grossier, la limite est plus importante—, et ces deux importances peuvent ne pas être équivalentes. Une façon de résoudre cela est de trouver des transformations des données d'origine pour lesquelles l'homogénéité correspond à l'homogénéité sémantique. Si, comme c'est notre cas, les données sont à haute résolution spatiale, l'utilisation de la texture en plus des, ou à la place des données radiométriques, pourrait ainsi améliorer les performances.

Ce chapitre tente de clarifier ces questions pour le cas particulier des segmentations d'images à très haute résolution spatiale (50 cm par pixel), faible résolution spectrale (canaux rouge, vert, bleu et, dans certains cas, proche-infrarouge), contenant des vues de zones rurales, avec le but d'obtenir des segments qui correspondent à des champs, des vergers, des forêts, des vignes, des lacs et autres objets cartographiques de taille moyenne.

Le chapitre commence, dans la section 3.1, avec une courte explication du mécanisme des segmentations hiérarchiques et multiéchelles, et ses avantages par rapport aux méthodes traditionnelles mono-échelle, ainsi qu'une présentation de l'algorithme "ensembles-échelles" de Laurent Guigues [GLMC03b, GLMC03a], qui sera utilisé par la suite.

Ensuite, la section 3.2 décrit l'utilisation de plusieurs transformations de l'espace radiométrique et de plusieurs descripteurs de texture en entrée de la segmentation. Ces transformations radiométriques [CJSW01, ASH03, Fin00, Fau79, BF00, VdWSLVD99, VB00, FP03, TK02, Nin03, TZS+03] et descripteurs de texture [Mac91, BS98, PNHA84, TLT00, TA95, TLT00, OPM02, Bai97, Gui00, WLL04, KP00, ZXZ03, Gar02], développés par d'autres chercheurs, sont décrits eux-mêmes dans l'annexe A. Des adaptations nécessaires de ces descripteurs pour les rendre compatibles avec notre système sont également décrites. La section 3.2.2 inclut une méthode simple qui permet d'utiliser des descripteurs de texture calculés sur des régions plus significatives que les voisinages carrés ou circulaires arbitraires utilisés habituellement.

La méthode de Guigues permet de trouver, pour toute valeur d'un paramètre d'échelle λ , la partition P qui minimise l'énergie $E(P, \lambda) = D(P) + \lambda C(P)$, où D est une mesure de l'attache aux données de la partition (plus les régions obtenues sont homogènes, plus D est faible), et C est une mesure de la complexité des frontières entre régions (plus les limites entre régions sont simples, courtes, et peu nombreuses, plus C est faible). Guigues propose une fonction D fondée sur la fonctionnelle de Mumford-Shah [MS89], et plusieurs options pour C , qui seront étudiées. De plus, seront étudiées une variante proposée par Guigues pour ses fonctions C , qui tente de faire passer les limites entre régions par les zones de l'image à forte variation, et une nouvelle variante qui favorise les limites qui sont orientées comme les directions principales du cadastre. Cependant, il semble que ces variantes n'apportent pas d'améliorations significatives.

Le coeur du chapitre est la section 3.4, dans laquelle nous explorons quelle combinaison de canaux d'entrée (les données radiométriques obtenues des senseurs, les transformations des espaces de couleur, et les descripteurs de texture) donne les meilleurs résultats. Nous explorons aussi le comportement des différents régularisateurs de forme (les fonctions C) et ses variantes, et à quel point du traitement il vaut mieux incorporer des indicateurs de texture. Pour cela nous utilisons une méthode d'évaluation quantitative de segmentations multiéchelles développée dans la section 3.3. De ces évaluations peut être conclu, entre d'autres, que l'utilisation d'indicateurs de texture pour la segmentation donne des résultats moins bons que l'utilisation des seules valeurs radiométriques, dans l'espace de couleur original ou non. Nous estimons que ceci est dû à la faible précision dans la localisation des valeurs de texture, puisque celles-ci sont définies pour un voisinage et pas pour des pixels individuels.

L'intérêt de ce chapitre est triple. D'abord, nous obtenons une partition de l'image en petites régions homogènes; si le cadastre n'est pas disponible, cette partition pourra être utilisée par les algorithmes de classification décrits ultérieurement qui, comme nous verrons, opèrent région par région. Ensuite, en considérant une segmentation multi-échelles comme une partition mono-échelle dans laquelle les limites entre régions seraient attribuées de son échelle

d'apparition, nous obtenons les limites importantes de l'image, et une indication de leurs importances, ce qui est nécessaire dans l'étape suivante, l'étape de recalage. Enfin, les indicateurs de texture et les espaces de couleur transformés peuvent être utilisés en complément ou alternative aux canaux radiométriques pour obtenir une meilleure classification.

Recalage (chapitre 4)

Pour pouvoir réaliser une classification par régions, il est nécessaire d'obtenir un découpage de l'image en régions homogènes en classe. Le parcellaire cadastral peut être utilisé comme découpage. Cependant, comme ses limites ne correspondent pas exactement aux limites des champs, les parcelles cadastrales sont souvent hétérogènes en classe. Une grande partie de ces hétérogénéités peut être résolue en recalant le graphe cadastral sur les limites saillantes de l'image. Dans le chapitre 4 cette question est étudiée. De façon alternative, nous pourrions développer des méthodes qui nous permettent de déterminer si une région est homogène en classe ou non, mais nous ne l'avons pas fait dans cette thèse.

Plus généralement, la partition d'une image en un ensemble de régions correspondant à des objets du "monde réel" est une tâche extrêmement difficile, qui a été un des objectifs majeurs de l'analyse d'image. Les causes majeures de cette difficulté sont que cette partition requiert des connaissances sémantiques a priori, et qu'il y a une interaction forte entre les procédures de partition et l'interprétation, dans le sens où l'interprétation essaye d'identifier quel objet est imagé dans chaque région d'une image partitionnée, et la partition utilise des connaissances qui seraient justement issues de cette interprétation.

Pour certaines applications, un résultat suffisant peut être obtenu en utilisant des données externes qui indiquent la position sur l'image des objets significatifs. Par exemple, dans le contexte de la télédétection, le graphe de cadastre peut être considéré comme une approximation grossière d'une partition du terrain en champs et zones agricoles. La correspondance n'est pas exacte parce que les limites des champs (l'usage du sol) ne suivent pas forcément les limites cadastrales (information foncière), un même champ peut s'étaler sur plusieurs parcelles cadastrales adjacentes, et une même parcelle peut contenir plusieurs champs.

Les deux premiers problèmes peuvent être résolus par un *recalage*, où le graphe du cadastre serait apparié à un graphe contenant les limites importantes trouvées dans l'image. À partir des couples appariés, nous obtiendrions la géométrie précise qui correspondrait aux éléments du cadastre, originalement à géométrie imprécise. De plus, les limites cadastrales entre parcelles adjacentes correspondant au même champ (le deuxième problème) ne pourront être appariées à aucune limite de l'image, ce qui nous permettra de détecter et corriger ces cas.

Le recalage de cadastre a été traité auparavant comme un problème de recalage non rigide. Dans les problèmes de recalage non rigide, le but est de trouver la transformation ou déformation (dans une classe de transformations) qui convertit au mieux une image en une image de référence. L'appariement est alors trivial. Les classes de transformations sont souvent un sous-ensemble des fonctions C^2 de $\mathbb{R}^2 \rightarrow \mathbb{R}^2$. Par exemple, Viglino et Guigues [VG02] appartiennent des graphes de cadastre aux limites de terrain en trouvant la meilleure transformation globale dans la classe des transformations polynomiales de degré fixé. Cachier *et al.* [CBD⁺03] et Goshtasby *et al.* [GSST03] donnent des revues d'autres méthodes de recalage non rigide. Ils notent, cependant, que la plupart des travaux se font dans le cadre de l'imagerie médicale. Chui et Rangarajan [CR03] proposent une méthode de recalage non rigide pour des ensembles épars de points qui pourrait être utilisée dans certains cas d'appariement de graphes.

Ces méthodes supposent que l'erreur de recalage initiale est causée par des déformations dues aux capteurs, par des erreurs d'acquisition, ou parce que l'orientation relative des données d'entrée n'est pas connue. Dans notre cas les images sont géoréférencées, et donc le troisième

problème n'a pas lieu, et en plus l'erreur initiale n'est pas due à des problèmes des capteurs ou de l'acquisition, mais au fait que les deux graphes représentent des données de nature différente (voir la figure 4.1 en page 85 pour des exemples de données d'entrée typiques). Nous voudrions recalculer les limites du cadastre sur les limites de l'image le plus possible. Plus précisément: nous voulons modifier localement la géométrie du graphe de cadastre de façon à en préserver la structure spatiale (c'est-à-dire, la topologie des faces de la représentation planaire), tout en incorporant les détails géométriques des éléments correspondants de l'image. Ce problème spécifique n'a pas encore été traité par d'autres auteurs.

Nous abordons ce problème du point de vue de l'appariement de graphes, comme Hivernat et Descombes [HD98] mais avec des modifications. Dans le contexte habituel de l'appariement de graphes, il y a deux graphes représentant la même réalité physique, et le but est d'apparier les arêtes et les nœuds qui correspondent à la même partie de cette réalité [GB03, Wal98]. Dans notre cas, nous utilisons une segmentation multiéchelles de l'image (obtenue selon les techniques du chapitre 3) pour obtenir une représentation de l'image sous forme de graphe, qui est mono-échelle mais dont les arêtes séparent les régions homogènes de l'image, et sont attribuées selon la dissimilitude entre ces régions. Ce faisant, nous supposons que toutes les limites significatives de l'image, ou au moins, celles qui devraient être appariées avec les limites du cadastre, existent dans la segmentation multiéchelles calculée. En faisant cette segmentation suffisamment fine, nous pouvons être assez sûrs que cette supposition sera vérifiée.

Deux méthodes pour résoudre le problème d'appariement sont proposées.

Dans la première, esquissée dans [TS04a], nous utilisons le recuit simulé pour trouver le meilleur appariement entre les arêtes du graphe de cadastre et les arêtes du graphe de segmentation de l'image. Elle est décrite dans la section 4.3. Cette méthode *par arêtes* conserve bien la structure des faces du graphe de cadastre, mais le graphe recalculé ne suit pas toujours la géométrie précise des limites saillantes de l'image, car parfois des arêtes auxiliaires (droites) doivent être ajoutées à la solution.

Dans la deuxième méthode, décrite dans [TSPD04], un appariement quasi optimal est trouvé entre les faces du graphe de cadastre (chaque face correspond à une parcelle cadastrale) et les régions de segmentation, des régions de l'image homogènes. Pour cela nous définissons un appariement initial, et des contraintes sur les appariements qui dépendent des degrés d'importance des arêtes entre deux faces voisines, et puis nous obtenons un appariement quasi optimal à l'aide d'un algorithme d'optimisation. Ceci est décrit dans la section 4.4. Deux variantes sont proposées, l'une en utilisant la relaxation probabiliste comme algorithme d'optimisation, l'autre en utilisant le recuit simulé. Dans la plupart des problèmes d'appariement de graphes [Wal98, TS04a, HD98, GB03], il y a deux graphes qui représentent la même réalité, et le but est d'apparier les arêtes et les nœuds qui correspondent aux mêmes parties de cette réalité. Dans ce cadre, cette deuxième méthode est une procédure d'appariement de graphes dans laquelle sont appariées des faces des graphes, et non des arêtes ou des nœuds. Cette méthode *par régions* donne un résultat qui suit toujours exactement la géométrie des limites saillantes de l'image, mais qui ne conserve pas toujours la structure spatiale des faces du cadastre.

Le meilleur algorithme, en termes quantitatifs, est la méthode par régions, avec la relaxation probabiliste comme algorithme d'optimisation. Des tests montrent une réduction de 32,2% dans la distance moyenne entre le cadastre et une référence, pour le graphe recalculé par rapport au graphe avant recalage. Si un graphe délibérément mal recalculé est utilisé en entrée, l'amélioration arrive à 43,7%. Les améliorations sont légèrement inférieures avec la méthode par arêtes, et encore moindres avec la méthode par régions avec recuit simulé. Cependant, plus important que cette évaluation quantitative est le fait que le graphe recalculé suit les limites de l'image (et ce, dans le cas de la méthode par régions, de façon exacte), ce qui nous permettra, dans les phases ultérieures du traitement, de faire des analyses statistiques pour la classification

sur des régions entières, sans les biais induits par le fait d'avoir dans une région de régions adjacentes. Même si ces méthodes ont été appliquées pour le recalage de graphes de cadastre sur des images aériennes, elles sont suffisamment générales pour pouvoir être utilisables dans d'autres contextes.

De plus, pour ces algorithmes, une mesure de qualité est calculée pour chaque arête cadastrale; ces mesures de qualité du recalage, décrites dans les sections 4.3.8 et 4.4.4, peuvent être utilisées pour déterminer quelles arêtes cadastrales n'existent pas sur l'image (parce qu'elles séparent des parcelles sur lesquelles un même champ s'étale) et peuvent donc être supprimées.

Grâce à ces techniques de recalage nous pouvons obtenir une partition de l'image en régions homogènes de taille plus grande que celles obtenues juste avec la segmentation du chapitre 3. Ces partitions sont nécessaires pour l'étape de classification.

De plus, le recalage nous permet d'utiliser le système pour la mise à jour de cartes anciennes d'occupation du sol. Si de nouvelles images sont prises sur une zone, et une classification ancienne de cette zone est disponible, la plupart des limites entre types de terrain de la classification ancienne correspondront à des limites dans les nouvelles images, avec peut-être quelques légères variations de géométrie, parce que, en général, il n'y aura pas eu des changements du terrain. En recalant les limites de la carte de classification ancienne sur les nouvelles images, nous obtiendrons donc, comme pour le cadastre, un découpage des images en régions qui seront, en général, homogènes en classe. Les régions hétérogènes pourront être traitées comme pour le cas du cadastre. Cependant, comme, pendant le développement de cette thèse, des données multitemporelles n'étaient pas disponibles, le fonctionnement du système pour la mise à jour n'a pas pu être vérifié.

Classification (chapitre 5)

La partie centrale de ce système est la classification proprement dite du terrain en classes d'occupation du sol d'intérêt topographique.

Les applications de classification utilisent normalement des algorithmes de classification pixel à pixel, des algorithmes qui donnent la classe de terrain la plus probable pour chaque pixel de l'image séparément. Ces algorithmes produisent du bruit "sel et poivre", c'est-à-dire que des pixels aléatoires dans une zone qui devrait être homogène en classe sont incorrectement classifiés comme appartenant à une autre classe, souvent la même pour tous les pixels erronés voisins. Ils ont aussi d'autres inconvénients, et leur taux d'erreur est souvent trop élevé pour une application comme la nôtre. De plus, ces algorithmes utilisent souvent des images à haute résolution spectrale, appelées images hyperspectrales. Comme nos images sont à seulement trois ou quatre canaux, les résultats seraient encore pires. Deux façons d'obtenir de meilleures performances sont explorées: la classification région par région au lieu de pixel par pixel, et l'utilisation de la texture au lieu de, ou en plus de, la radiométrie.

Quant à la première solution, nous proposons, dans le chapitre 5, des algorithmes de classification qui donnent une réponse pour chaque région au lieu de pour chaque pixel. Pour cela il est nécessaire d'obtenir un découpage de l'image en régions, selon des méthodes décrites dans les chapitre 3 et 4. Avec ceci nous évitons en grande partie le bruit "sel et poivre" et, comme la décision est prise sur la base de plus d'information, de meilleurs résultats peuvent être espérés.

Pour la deuxième solution, comme les images disponibles sont à haute résolution spatiale, il semble raisonnable d'essayer de compenser le manque de résolution spectrale par l'utilisation de la texture, qui devient disponible à ces résolutions. Dans la section 3.2, et en général dans le chapitre 3, plusieurs indicateurs de texture, la plupart dérivés de l'état de l'art avec quelques

adaptations à notre problème, sont étudiés. Dans son stage de DESS, S. Giffon [Gif04] étudie quels indicateurs de texture fournissent le plus d'information pour ces classifications.

Le chapitre 5 expose d'abord les méthodes classiques, bayésiennes, de classification pixel à pixel. Ensuite, deux nouveaux modèles de probabilité pour une classification région par région sont introduits.

Dans le premier modèle de probabilité, chaque région est classifiée dans son ensemble, soit par une approche purement bayésienne, soit en utilisant des mesures d'*attache aux données*, c'est-à-dire, en étudiant si le comportement statistique des pixels d'une région suit les comportements de référence obtenus lors d'une phase d'apprentissage préalable. Dans le deuxième modèle, qui est une amélioration du premier (du moins, en théorie), nous tenons compte séparément du comportement intra-région et du comportement inter-région des pixels appartenant à la même classe de terrain (alors que dans la première méthode, comme pour la classification pixel à pixel, les deux variations sont confondues). Nous montrons que, pour certaines classes de terrain, ce modèle *composé* décrit mieux les données d'apprentissage que les modèles standard.

Pour ces deux modèles, nous proposons plusieurs algorithmes de classification. Ces algorithmes utilisent le fait qu'une image soit découpée en régions homogènes, et non seulement affectent une classe de terrain à chaque région, mais donnent une mesure de confiance pour chaque résultat, ce qui permet de faire une classification semi-automatique. Ces algorithmes peuvent être classés en deux types. Les premiers se fondent sur la théorie des probabilités, et les deuxièmes sur la similarité entre les caractéristiques des pixels dans la région à classer et un ensemble d'entraînement. Comme ces algorithmes donnent une seule réponse pour tous les pixels d'une région, mais qu'ils examinent les données de chaque pixel individuel, ils peuvent mesurer l'adéquation de cette réponse aux caractéristiques de chaque pixel, ce qui permet de donner, en plus de la classe de la région, une mesure de la confiance qui peut être attribuée à cette classification. Ces algorithmes de classification sont du type supervisé.

Une classification implique l'estimation d'un modèle du monde, par laquelle est explicitée une relation entre les objets du monde et ce qui en est perçu. Dans notre cas, les liens seraient établis entre les types de terrain, et ses apparences en termes de radiométrie et de texture. Les algorithmes de classification diffèrent les uns des autres selon le type de modèle utilisé, selon les méthodes qui permettent d'obtenir un modèle qui décrit les données, et selon les méthodes qui permettent de déterminer la classe (le type de terrain dans notre cas) à partir des données. Dans les méthodes de classification *supervisées*, des modèles sont estimés à partir de *données d'apprentissage* ou d'*entraînement*, un sous-ensemble des données pour lesquelles un expert a préalablement fait le lien avec les classes. Ces modèles permettent alors de classer le reste des données. Dans les méthodes de classification *non supervisées*, ou de *clustering*, il n'y a pas de phase d'estimation ou d'apprentissage; les données sont groupées en classes en une seule étape, et cette classification n'associe pas d'information sémantique à chacune des classes obtenues. Ce découpage en classes, même sans lien sémantique, est en soi un modèle.

Dans la modélisation probabiliste, pour chaque classe de terrain sont estimés les paramètres d'une variable aléatoire. Alors, en supposant que chaque pixel appartenant à cette classe soit une réalisation de la variable aléatoire de sa classe il est possible de trouver la classe la plus probable pour chaque pixel. Beaucoup de méthodes de classification qui ne sont pas explicitement probabilistes peuvent être interprétées dans ce cadre, où les variables aléatoires seraient uniformes avec un support qui est déterminé par la phase d'estimation.

Le modèle composé présenté dans ce chapitre est un modèle probabiliste qui ne suppose pas que chaque pixel soit une réalisation de sa variable aléatoire de classe. Ce qu'il suppose, c'est que les pixels puissent être regroupés en régions de la même classe, qui sont imagées à partir du même objet du terrain. Par exemple, tous les pixels correspondant à un certain champ de blé seraient groupés entre eux, mais pas groupés avec ceux d'un autre champ de blé, adjacent mais différent. Nous faisons l'hypothèse que, pour certaines classes de terrain, non seulement

les distributions des valeurs des pixels dans les différentes régions sont différentes, ce que nous pourrions espérer à cause du bruit d'échantillonnage, mais que les pixels des régions différentes mais de la même classe sont des réalisations de variables aléatoires différentes. Ceci se justifie par le fait qu'une même classe de terrain, comme "forêt", peut avoir des apparences très diverses selon l'essence des arbres, la saison, la pente, la météorologie, l'heure de prise de vue, et beaucoup d'autres facteurs. Cependant, nous ne suggérons pas de diviser la classe "forêt" en sous-classes, une pour chaque valeur de ces facteurs, et d'apprendre des modèles séparés pour chaque sous-classe, ce qui demanderait des ensembles d'apprentissage énormes. Notre hypothèse est que toutes ces variables aléatoires correspondant aux diverses apparences d'une même classe de terrain ont des similitudes qui peuvent elles-mêmes être modélisées. Ce modèle composé est fondé sur cette structure en deux niveaux, en utilisant des *variables aléatoires composées*.

Le chapitre suit cette structure: D'abord, dans la section 5.2.1, nous présentons le modèle bayésien de classification pixel par pixel classique, pour établir une référence de notation. Ensuite, dans la section 5.2.2, nous revoyons les modèles de classification par région qui avaient été présentés dans [TSB05], avec quelques extensions. Dans la section 5.3 nous décrivons en détail les modèles composés de classification par région, suivis, dans la section 5.4, de méthodes de classification qui utilisent ces modèles composés. Dans la section 5.5, nous présentons la procédure qui permet d'obtenir ces modèles composés (ainsi que les modèles "simples" par région des sections antérieures) à partir de données d'entraînement. Dans les sections qui suivent, nous présentons des évaluations quantitatives de ces méthodes et modèles de classification: D'abord, dans la section 5.7, nous étudions la procédure d'estimation de modèles. Ensuite, dans la section 5.8 nous reproduisons les résultats principaux de [TSB05], où nous décrivons une procédure de classification en forêts et non-forêts, à cause de son importance pour cette thèse en tant qu'application pratique. Finalement, dans la section 5.9 nous étudions les performances en classification des méthodes et modèles présentés, qui atteignent des taux de bonne classification de 99,9% pour le site de test plus simple de Saint-Léger, et de 94,7% pour le site plus complexe de Toulouse.

Estimation d'orientation (chapitre 6)

Dans le chapitre 5, nous voyons comment obtenir une classification du terrain avec des données à haute résolution spatiale et faible résolution spectrale. Le manque de résolution spectrale est compensé par l'usage de la texture et par la possibilité de faire une classification région par région.

Cependant, si seulement la radiométrie et des indicateurs de texture classiques sont utilisés, il y a un fort chevauchement des comportements de certaines classes, ce qui rend difficile leur séparation. Par exemple, la confusion est forte entre les forêts et les vergers, et aussi entre les vergers et les vignes. Dans une moindre mesure, il y a aussi une certaine confusion entre champs, vergers et vignes, et aussi entre champs labourés et non. Le problème est que ces classes ont des caractéristiques radiométriques similaires, et peuvent être distinguées seulement par l'orientation et la période précises, si elles en ont, de ces textures: les forêts et les champs non labourés ne sont pas orientés, les champs labourés ont une orientation principale, et les vergers et vignes en ont deux. Il est possible de distinguer entre vergers et vignes par la valeur précise de la distance entre files de plants ou arbres, qui dépend des pratiques agricoles et de la législation locale.

Il est donc nécessaire d'obtenir d'estimations précises des périodes des textures périodiques, et de pouvoir distinguer entre les textures non orientées, celles orientées sur une direction, et celles orientées sur deux directions. Dans le cas à une orientation, plusieurs sous-cas

peuvent être distingués, selon que la texture est périodique ou non selon l'orientation principale ou la direction orthogonale à cette orientation. Cependant, les méthodes existantes d'estimation d'orientation ne sont pas adaptées à cette tâche. Certaines se fondent sur l'analyse de Fourier [Gar02,CRH02], ce qui présente des difficultés pour le traitement de régions à forme arbitraire; de plus, la forme des textons (ici, la forme des plantes ou arbres individuels) devient des harmoniques dans le domaine de Fourier, et est traitée comme du bruit par ces algorithmes. D'autres [DCGB02,ZXZ03,MGBdC04] utilisent des gradients de l'image, et sont adaptés aux textures statistiques mais pas aux textures à textons comme celles qui nous concernent. Chanussot, Bas et Bombrun [CBB05] utilisent une combinaison d'analyse de Fourier et de transformée de Radon.

Warner et Steinmaus [WS05] utilisent l'autocorrélation le long des directions verticales, horizontales, et les deux diagonales, dans le voisinage de chaque pixel, pour distinguer entre vergers, vignes et autres champs. Ils arrivent à un taux de bonne classification pour ces trois classes de 95%, mais ils n'obtiennent pas des estimations réalistes pour l'orientation et les périodes.

Dans le chapitre 6, donc, nous proposons une méthode adaptée à notre problème. Il s'agit d'une méthode fondée sur les variogrammes qui peut déterminer l'orientation principale d'une image multi-canal à forme arbitraire, ainsi que sa période, l'espacement entre files, et l'orientation secondaire, s'il y en a une. Elle avait été présentée, résumée, dans [TS05b].

Les paramètres obtenus par cette méthode peuvent ensuite être utilisés pour discriminer les classes problématiques listées dans les paragraphes précédents. Des évaluations montrent un taux de bonne classification de 95,5%. De plus, pour 82% des orientations estimées, l'erreur d'estimation est inférieure à 3°, et pour 81% des périodes estimées, l'erreur d'estimation est inférieure à 1 pixel.

Conclusions et perspectives (chapitre 7)

Finalement, le chapitre 7 contient un récapitulatif de la structure du système présenté, et des performances obtenues. Y sont listées les contributions faites par l'auteur dans cette thèse, et quelques perspectives de recherche.

Les contributions de cette thèse sont les suivantes: D'abord, une méthode d'évaluation de segmentations multi-échelles, et les résultats d'une évaluation exhaustive de segmentations avec cette méthode. Ensuite, une méthode par régions et une méthode par arêtes pour le recalage de graphes sur des images. Dans le domaine de la classification, j'ai présenté des algorithmes de classification par régions utilisant des modèles de probabilités classiques, un nouveau modèle de probabilités "composé", des algorithmes de classification l'utilisant, et des procédures d'estimation pour obtenir ces modèles. Finalement, j'ai proposé un nouveau estimateur d'orientation et période, pour s'affranchir des problèmes liés à l'estimation par Fourier.

Les perspectives de recherche —à part quelques développements utiles du point de vue de production mais sans intérêt scientifique, comme l'intégration de l'estimation d'orientation et période dans la chaîne de traitement, l'évaluation du système en mode mise-à-jour, la détection et découpage de parcelles hétérogènes, ou une évaluation de la chaîne complète— sont la poursuite de l'utilisation de la forme pour la segmentation, l'étude de la valeur de la forme pour la classification —puisque par exemple les limites droites sont plus caractéristiques des champs que des fleuves—, la fusion de résultats de classification, et l'usage des modèles numériques de terrain ou d'élévation, comme Arnaud Le Bris a déjà commencé à faire. La perspective la plus intéressante est l'utilisation de modèles de plus haut niveau pour la classification, des modèles qui généreraient des informations sémantiques comme des "villages", des "routes", ou

des “fleuves”.

En conclusion, en vue que les algorithmes traditionnels de classification du sol ont des performances insuffisantes pour la production cartographique, et que nous pouvions attendre des résultats encore moins bons en raison de la faible résolution spectrale de nos données, nous avons compensé cette faible résolution spectrale par l’usage de la texture et de la classification par région, deux options qui deviennent possibles grâce à la haute résolution *spatiale* de nos données. En combinant ces deux options, et en suivant une approche semi-automatique et non tout-automatique, nous obtenons des précisions de classification du terrain très bonnes, dans le contexte de la détection de végétation en zone rurale.

1 Introduction

At a mapping agency such as the Institut Géographique National (IGN), the French National Mapping Agency, producing quality high-resolution maps is one of the main activities.

How are maps made?

In olden times, for each map there used to be several large sheets of a plastic material, one for each of the colours to be used in a map; some colours with much information were split into several sheets. These sheets were composed of a transparent layer and an opaque layer. The opaque layer was cut out with special tools to produce a *mask*: in the printing process, the pattern created by the removed lines and polygons got printed onto a sheet of paper in the appropriate colour for that mask. From time to time, field surveyors would look for changes: new buildings, upgraded roads, forest areas converted to fields, and so on. These changes were recorded on the plastic sheets by removing additional opaque material, or pasting in small patches of opaque material to undo a previous cut-out.

Nowadays, all this data is stored electronically in what is known as a *geographical database*. Geographical elements are stored as objects in this database. Information for the complete territory managed by a mapping agency can be stored in a single database. Additions, deletions, and modifications are only a mouse click away, and no longer require cutting and pasting real plastic. In most cases, separate data sets are kept for each map scale, because in order to produce, say, a 1:50000 map we cannot simply take a more detailed 1:25000 map and shrink it—there are issues regarding the acceptable level of detail and information simplification for each scale. In other cases data is stored for only the most detailed scale, and semi-automatic simplification or *generalization* procedures [Bar04, Duc03, DC03, Rua02, Duc01, Rua01, RM97, RP96] are used to convert this data to a coarser scale.

However, this data is still manually entered into the database and manually updated. Human photo-interpreters examine aerial photographs of the area of interest and, complemented by reports of field surveyors, modify the computer database. While significantly faster than the previous method, this is still very slow and expensive: For example, IGN's topographic maps at 1:25000 scale are, because of this, 10 years old on average.

There are many research projects at IGN and elsewhere aiming at fully or partially automating interpretation of aerial photographs, in order to, as a first step, populate the geographical database and, later, keep it up to date as new photographs are taken of each area. For example, there has been research into automatic road detection and tracking [YC00, YCWZ01, Boi00, BCA99], and many ongoing projects on detailed 3D modelling of buildings [PTJ00, TD02, NN01, VD00, CPDJS02, BPPD04, BPPDH04, DPDPC04] and building superstructures.

This Ph.D. thesis, part of this research effort, is an attempt at partially automating the extraction of information on vegetation areas from aerial images. The goal is to obtain algorithms which, given high-resolution aerial images of an area, can automatically locate areas of vegetation in these images, precisely obtain their boundaries, and classify them according to their kind. Typical kinds of vegetation, in map-making applications, are forests, fields, vineyards, and orchards. The system also recognises other non-vegetation surface-like objects, such as water or bare soil areas. All these kinds of land cover are known as *terrain classes* or *land cover classes*.

This research has a clearly applied goal: to speed up map production. As such, I have not hesitated to take shortcuts and explore less beautiful scientific and algorithmic possibilities, when these produced better results than other prettier, more mathematically sound, and theoretically superior methods —while keeping in mind that an algorithm designed with practicality in mind need not be a kludgy contraption, and that, many times, the theoretically superior method also turns out to be the best one in practice. This practical standpoint has influenced this thesis in three more ways: First, I took advantage of additional data available at IGN whenever necessary; for example, cadastre data gives a rough indicator of the position of fields. Second, I developed algorithms which have reasonable requirements in terms of computational power. And third, and most important, I recognize the fact that IGN prefers to have a partial classification —that is, one in which only a part of the study area is classified— with a very high quality, rather than to have a lower-quality classification covering the whole area of interest. In the first case, human photo-interpreters can concentrate their limited time in the remaining, probably more complex, areas, whereas in the second case the results for the whole study area, because of their lower quality, would have to be double-checked in full by a human photo-interpreter at great cost. Therefore, the classification algorithms presented here provide, in addition, indications of the confidence that the algorithm itself has on its results; depending on the acceptable quality, the photo-interpreter can then reject the portion of the results whose confidence measure falls under a certain threshold.

More precisely, the image analysis system presented in this thesis will take, as inputs, a vectorized cadastre graph, and a set of aerial high-resolution georeferenced digital photographs of a rural area. These photographs will have been orthorectified, that is, transformed from a conical perspective into a parallel perspective, with the light rays in the vertical direction. The photographs will be at a resolution of 50 cm per pixel, and have 4 channels (red, green, blue, and near infrared) with radiometric correction of atmospheric effects. In this thesis, we have also performed tests on analog 3-channel images (red, green, and blue channels only) at 80 cm per pixel. Apart from using the cadastre to obtain a rough idea of the location of fields, the system will use it to locate buildings and roads, and exclude them and their vicinities from processing. The system is not to be used in urban or suburban areas, not even for the detection of vegetation therein: other research projects are devoted to urban vegetation analysis. Finally, the system can also be used to update an existing geographical database when new imagery becomes available. In this case, instead of the vectorized cadastre graph, the system will use the old land cover classification to obtain approximate field locations.

Classification is of the supervised kind. This means that the system must be trained by providing it with a manually defined *ground truth* prior to using it for land cover classification. A ground truth is a data set indicating, in one or more training images, the location of a certain number of regions for each terrain type. The system analyses the characteristics of these training regions and uses this knowledge to, later, give the most likely class for each region in other images.

The system will analyse this input data and give, as an end product, a segmentation of the original images into agriculturally homogeneous regions. Agricultural homogeneity is defined in terms of the vegetation cartographic class (which classifies terrain into bare land, forest, fields, orchards, vines, . . .) and also on other properties such as colour, so that fields of the

same culture in different growth states are separated. This will not, in principle, be a minimal segmentation —adjacent regions may belong to the same class— but this can be easily solved, if required, by a post-processing step. Along with the land cover class for each region, the system will provide a measure of the confidence that it itself has on the fact that this region belongs indeed to the given land cover, so that a human operator can assume that the highest-confidence areas are correct, and concentrate on the areas that the system is more unsure about.

The analysis sequence is divided into three steps: segmentation, registration, and classification, each described in a separate chapter. A further step, the estimation of texture orientations and periods, improves the classification in some cases.

The central step is, of course, the classification step. This is described in chapter 5. Typical classification applications use per-pixel classification algorithms —algorithms which give the most likely terrain class for each pixel separately—; these produce salt-and-pepper noise (individual random pixels within a region of one class are classified into a single other class) and other artifacts, and their error rates are usually too high. In addition, these typical algorithms are normally used with images of high spectral resolution. Since our images have only three or four channels, we can expect results to be even poorer. I explore two ways of obtaining higher performances: per-region classification, and texture.

First, I develop new classification algorithms which classify not a pixel at a time, but a whole set of pixels at a time. With this we avoid salt-and-pepper noise, and, because the classification decision is based on more information, we can expect better results. I present two kinds of algorithms: some based on probability theory, and some based on the similarity of behaviours of the pixels in a region and in a training set. Since these algorithms produce a single decision for a set of pixels, but can also examine each individual pixel, they can obtain a measure of how well the global decision fits each individual pixel; this, and other methods, allow these algorithms to return not just a classification for each region, but also a measure of its confidence in that classification. These classification algorithms are supervised, that is, they operate on a model of the terrain classes which is generated in a training phase using images with some areas manually labelled as belonging to each class of interest. In chapter 5 I present new probability models which may more faithfully model the behaviour of data in this problem, and thus give better classification performance; the issue of training, that is, of estimating the parameters of these models from training images, is also discussed in detail.

Second, because the available images have a very high spatial resolution, it makes sense to try to compensate for the lack of spectral resolution with the use of texture. In section 3.2, appendix A, and in general in chapter 3 I study many different texture features, mostly derived from the state of the art but with adaptations to this specific problem. Giffon's master's thesis [Gif04] studies which texture features convey the most useful information for the purposes of classification.

It turns out, however, that some terrain classes are very easily confused, and that radiometry, transformed colour spaces, and typical state-of-the-art texture features are not enough to differentiate them. The problem is that these classes can only be distinguished based on the orientation and precise periodicity, or lack thereof, of their texture. None of the existing orientation and period estimators is suitable for this task, which is why in chapter 6 I present a novel, very precise, estimator for the main and secondary orientations and the periods of a periodic texture.

Using per-region classification algorithms implies the need to partition the input image into regions. Each of these regions will be classified into one class. Therefore, each region should be class-homogeneous, that is, all of its pixels should actually belong to the same terrain class. Alternatively, methods for determining whether a region is homogeneous or not should be developed. The cadastre data available at IGN can be used for this purpose: the cadastre gives

the extent of each piece of real estate, each plot of land. This is fiscal information, indicating land ownership, and land owners do not have to follow these limits when growing their crops—they will typically leave some bare areas in the borders, grow more than one crop in one plot, or grow the same crop in adjacent plots with the same owner. Nevertheless, many real edges in land cover do correspond, approximately, to cadastre edges.

In chapter 4 I propose several graph matching algorithms which, given cadastre information and the corresponding image, can determine, for each cadastre edge, the exact position of the corresponding edge in the image, if there is one. This is known as *registration*. In this way, we obtain a partition of the image into regions whose limits follow actual image limits, and whose topology is that of the original cadastre. These registered cadastre regions, because their boundaries follow land cover boundaries, are class-homogeneous, except in the case where more than one crop is grown in a cadastre plot. This solves the problem of crops not covering exactly the extent of the land plot. That adjacent plots have the same crop is not really a problem, since they can trivially be merged after classification. The problem of having more than one crop in a land plot is not solved, so there is the need to develop methods for determining whether a region is homogeneous or not. I did not address this issue in this thesis. However, heterogeneous regions will be classified by the algorithms of chapter 5 with a very low classification confidence, since the pixels in the region do not look like any of the single-class models, and this could be used to detect heterogeneous regions.

Furthermore, registration also allows the use of this system to update an old land cover map. If an old land cover classification and recent images of the same area are available, most of the edges in the old classification will correspond to image edges, with perhaps some slight geometry change, simply because, for the most part, there will have been no changes in the terrain. Registering the limits of the old land cover map onto the new image will produce a partition of the image into regions which are, mostly, class-homogeneous. The heterogeneous regions—resulting from land cover changes greater than simply small changes in the limits of land cover regions—can be handled with the same techniques as for heterogeneous registered cadastre regions. However, since multi-temporal data was not available for this thesis, I did not test the use of this system for database updating.

The registration step registers cadastre edges onto image edges which are visually strong, or *salient*. In order to determine the position of image edges and their saliency a multi-scale segmentation algorithm is used. Basically, the scale of analysis at which an edge appears in the segmentation is used as an indication of its saliency: edges which appear at coarser scales are considered to be more salient. In chapter 3 I discuss many issues related to this segmentation. There are many parameters to set in the particular algorithm that I have chosen [Gui04], and in this chapter I study their optimal values. Furthermore, since edge saliency defined from an application point of view—that is, the more different the crops in the two sides of the edge are, the more salient the edge is—and edge saliency defined from an image point of view—saliency related to scale of analysis—may not be the same, I study whether it is better to segment the original images or derived images, where texture features are used instead of, or in addition to, the raw radiometry values. I include extensive tests that show the optimal combination of radiometry and texture channels, which need not be the same as those used for classification.

This thesis is very modular, and in each of the steps I include scientific contributions that can also be used on their own. The document is organized following the order in which the successive steps have to be executed, which is, roughly, the reverse of the natural chain of thought followed in the first part of this introduction. After a literature review in chapter 2, I start in chapter 3 by proposing several texture parameters and colour spaces, derived from the literature and adapted to this problem, and by studying the optimal parameter and input configuration to Guigues' image segmentation algorithm [Gui04]; a brief introduction to multi-

scale segmentation algorithms is also included. I continue in chapter 4 by presenting two novel algorithms for the registration of a graph onto an image, which is used to register cadastre data. The behaviour and performance of these algorithms under different conditions is studied in detail. Following this, chapter 5 gives a brief reminder of traditional per-pixel classification algorithms followed by a presentation of several novel per-region classification algorithms, and of a new nested probability model. The appropriateness of this new model is discussed, and the classification algorithms are evaluated. I include the description of a particularly useful application, the classification into forest and non-forest, for which these algorithms give especially good results (section 5.8). Since in chapter 5 it becomes apparent that some classes are easily confused, and that radiometry and the texture features of chapter 3 alone are not enough to distinguish them, in the next chapter, chapter 6, I present a novel method for obtaining a precise estimation of the orientations and periods of arbitrarily-shaped, multi-channel, periodic textures, which can be used to avoid this confusion.

Figure 1.1 shows a flowchart of the complete process.

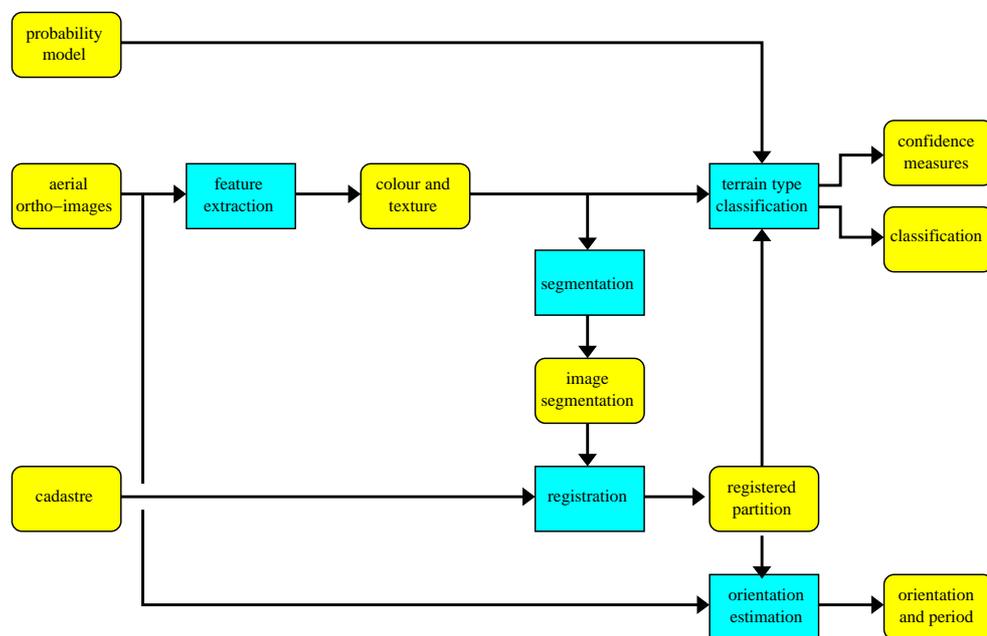


Figure 1.1: Flowchart depicting the processing steps (in blue rectangles) and intermediate results (in yellow rounded boxes) for the complete system.

Each of these chapters—segmentation, registration, classification, and orientation estimation—includes extensive quantitative evaluations using real images of similar characteristics to those that the system will handle in production use, so the results can be taken to be very significant. In that respect, the evaluations of chapters 5 and 6 can be taken to be global evaluations of the whole system. I close with final conclusions and suggestions for continuing research in this topic in chapter 7. An executive summary in appendix E tries to clarify the practical utility of this work in non-technical terms.

2 | Literature review

This chapter presents a survey of the literature available on the topics that the thesis will cover. As we have seen in the introductory chapter, this thesis contains three main parts, segmentation, registration, and classification. Here we will discuss articles found in the literature about these topics.

Furthermore, both for segmentation and classification we have suggested the use, in addition to the raw radiometry channels, of other colour channels or of texture indicators. For the segmentation step, the use of shape will also be explored, in order to obtain regions which have shapes similar to those of real fields. All these topics will be reviewed in this chapter.

Finally, we also review some topics that are more marginally related to this thesis, but which could be of use for further research, such as data fusion and decision.

2.1 Colour, texture, and shape

2.1.1 Colour

We can see objects because light from a light source is reflected or diffused from these objects into our retinas, or because the objects generate light themselves. There are other phenomena such as refraction or transmission, or even less common ones like fluorescence, which will not be considered in this review nor in the thesis. Light reaching the retina on a certain direction — a light ray — has its energy distributed in a certain way along the spectrum, which we describe using a spectral power distribution, $E(\lambda)$.

A light ray stimulates light-sensitive cells in the retina, called *cones* and *rods*. Each kind of receptive cell (usually rods and three kinds of cones) has a certain spectral sensitivity function $C_1(\lambda)$, $C_2(\lambda)$, $C_3(\lambda)$, $R(\lambda)$. The response for each sensor will be

$$\begin{aligned}c_1 &= \int_{\lambda} C_1(\lambda)E(\lambda)d\lambda, & c_2 &= \int_{\lambda} C_2(\lambda)E(\lambda)d\lambda, \\c_3 &= \int_{\lambda} C_3(\lambda)E(\lambda)d\lambda, & r &= \int_{\lambda} R(\lambda)E(\lambda)d\lambda.\end{aligned}$$

Cones have relatively narrow sensitive bands, which we roughly associate with red, green, and blue colours. Rods have wider pass-bands and are insensitive to colour. The combination

of responses from each sensor type gives the subjective colour sensation. See [FP03] for an extensive yet easy-to-follow description of colours and the image formation process.

Digital cameras work in a very similar way. For colour cameras with an extra near-infrared channel, such as the one we used in this thesis, there are four kinds of sensors—in our case, four identical CCD sensors with different colour filters on each—, with spectral sensitivity functions $R(\lambda)$, $G(\lambda)$, $B(\lambda)$, and $I(\lambda)$ with narrow pass-bands responding to red, green, blue and near-infrared light, respectively. There is no colour-insensitive (or *panchromatic*) sensor. The response for each sensor is then

$$\begin{aligned} r &= \alpha_r \int_{\lambda} R(\lambda)E(\lambda)d\lambda, & g &= \alpha_g \int_{\lambda} G(\lambda)E(\lambda)d\lambda, \\ b &= \alpha_b \int_{\lambda} B(\lambda)E(\lambda)d\lambda, & i &= \alpha_i \int_{\lambda} I(\lambda)E(\lambda)d\lambda, \end{aligned}$$

where α_r , α_g , α_b , and α_i are calibration parameters that take into account the fact that $\int R(\lambda)$, $\int G(\lambda)$, $\int B(\lambda)$, and $\int I(\lambda)$ are usually different. Images containing the responses of these four sensors are said to be in *red-green-blue-infrared* colour space, or RGBI. If there is no infrared channel, we get the commonly used RGB space.

Perceptual relevance

However, it has long been known that RGB coordinates are not directly related to the human perception of colour. That is, Euclidean distance between two colours expressed in RGB coordinates does not correspond to subjective colour difference as indicated by test subjects.

Many methods have been found to obtain *perceptually relevant* colour distances, that is, mathematical distances between colours that do correspond to subjective colour differences. This is usually done by using the Euclidean distance on a transformation of the RGB colour space, such as the HSV space, the Karhunen-Loève colour transform [VdWSLVD99], the CIE colour spaces [VB00,FP03,TK02], or chromaticity spaces [Fau79,BF00]. See also [CJSW01] for a review of colour spaces in the context of image segmentation. Since we want to separate vegetation regions which *look* different, it seems that using perceptually relevant colour spaces is important.

A large quantity of colour spaces have been proposed. A new problem appears, that of selecting the most appropriate. Vandenbroucke, Macaire, and Postaire [VMP03] give an algorithm that, for a given classification problem, selects the most adequate components—for classification purposes—from a given set of colour spaces.

Finally, Suzuki *et al.* [SSAO00] propose a complex system for dynamic shadow compensation of aerial images based on colour and spatial analysis. Because shadows modify the perceived colour of objects, it is interesting to detect them. It processes high and low frequencies separately to attain a high geometrical precision, and classifies areas in shadow and non-shadow by Bayesian classification using manually-defined training areas. A simpler method, presented by Tsai [Tsa03], is based on recursive decomposition of an image by quad-trees until the intensity histograms of all regions are bimodal (the darker mode is assumed to correspond to shadow). We did not find it necessary to implement any shadow-compensating method since we are dealing with rural images where sharp shadows, usually caused by buildings, are uncommon.

Colour constancy and radiometry normalization

Colour information depends on sensor characteristics (but this is usually calibrated out using the α factors described above), material characteristics of the surface being viewed, characteris-

tics of the illuminating light and the geometry of the scene (illumination angles, for example). We are usually interested in the second factor only, surface characteristics, and we human beings are remarkably good at removing the third and fourth factor from the problem—for example, we usually know what colour an object would have under white light, even if we are seeing it under another light colour.

There has been some work at obtaining *colour constancy*, *illuminant invariance* or, in other words, colour measures which are invariant with respect to the illuminating colour.

Finlayson [Fin00] examines which perceived colours are compatible (*consistent*) with a particular light, and conversely, which illumination is consistent with a particular perceived colour, and goes on to define a 1-dimensional colour measure that is invariant to illumination changes, but which uses two parameters which depend on the spectral characteristics of the camera, which were not available to us.

Berens and Finlayson [BF00], citing previous work from Faugeras [Fau79] describe the *log-opponent* chromaticity coding of colour space, which is invariant under brightness changes, and which transforms other illumination changes into simple translations. They suggest a method to obtain complete illumination invariance by subtracting (or, for some applications, separating) the mean colour from the images.

Gevers and Stokman [GS03] propose a complex system for detecting photometric invariant regions in multi-spectral images which is robust to sensor noise and achieves brightness and colour illumination invariance (through white balancing), but works on heterogeneous materials only.

Vertan and Boujemaa [VB00] propose a separation of colour and intensity information, and a dimensionality-reducing colour transformation which achieves some colour constancy and has a strong perceptual foundation.

Manduchi [Man04] presents a method for compensating illuminant-dependent variation in colour images, which makes colour classification possible with just one training image.

Song and Woodcock [SW03] investigate the effects of external factors such as slope, orientation, sun angle, and atmosphere, on the spectral signature of trees, and evaluate several correcting methods. There exist colour and texture features which are invariant to these factors, for example, illuminant-independent colour measures.

There has also been some work on other normalizations. Palubinskas, Müller, and Reinartz [PMR03] give a simpler method to deal with radiometric distortion due to viewing angle at the edges of satellite images. Tanaka, Sugimura, and Hashiba [TSH03] try to compensate the effect on colours of taking images of rural areas at different dates.

2.1.2 Texture

If colour can be used to infer characteristics of the viewed object at the microscopic level, texture reveals macroscopic attributes of objects. Texture is present in almost all images, is easy to recognize, but hard to define [FP03]. Tentative definitions of texture abound: the presence of multiple instances of a base image element (the *texon*), optionally arranged in a particular way; periodic structure, detectable by frequential analysis tools such as the Fourier transform; fractal or fractal-like patterns; regions of a constant “complexity”, usually measured by the entropy of some image parameter; and many more, including combinations of these.

Texture can be used to differentiate between two otherwise equally-coloured regions. More importantly, because texture relates to physical properties of objects, we can identify, at least in part, the contents of a region from its texture. For example, some orchards can be distinguished from certain vines by the inter-row distance only.

There are many experimental comparisons of several texture extraction algorithms, for example [GPK00,SS02].

Orientation and frequency analysis

Orientation is a primary component of texture. In some cases, orientation must be extracted because it is significant for the application; some algorithms are proposed in [DCGB02]. In other cases, rotation or scale-invariant texture parameters are needed [Pun03,ZT02,ZT03,VdWSVD98].

Periodicity is also important in some applications. Orwell *et al.* [OBFW98] propose an iterative periodicity detector which works even for very noisy images.

Gabor filters [Mac91,BS98], like wavelets [Mal89,Mal97,HW89], are localized in space and frequency, and can be used to retrieve frequential properties of a texture. This in turn can be used for segmenting an image into texturally-homogeneous regions [ZTM02,NSK02,BJZ03] or to obtain image descriptors for use in classification [SL00] or retrieval [MM96]. Gabor filters make local orientation explicit, whereas it is only implicit in wavelets. Fourier-based techniques [Gar02,CRH02] also extract the local orientation and periodicity, but are not localized. Mota *et al.* [MASB04] present a local orientation estimator that handles textures with multiple orientations, more complex but with a more solid mathematical background than [Gar02]. Michelet *et al.* [MGBdC04] give another such estimator, implemented recursively to reduce the computing cost.

A more complex use of Fourier analysis for texture description is given by Hsu, Calway, and Wilson [HCW93], where the Multiresolution Fourier Transform is used for image segmentation and texture synthesis while taking into account the geometric warping which is typically found in images of natural textures.

Myint *et al.* [MLT04] present a wavelet-based texture feature for land use classification in high-resolution images of urban and suburban areas, and compare it to traditional features such as fractal-based descriptors and co-occurrence matrices, showing superior results.

Some methods do not only give the primary orientation and frequency, but also the remaining periodic components, which is useful for detecting certain seeding patterns. Zhou, Xin, and Zhang [ZXZ03] propose a very reliable system for obtaining the main orientation of a texture (but not its periodicity), and the scale of analysis at which this orientation is found.

South, Qi, and Lusch [SQL04] compare classification methods to classify crops into “not tilled” and “conventional tillage”. They use medium-resolution multispectral data to infer the tilling method; this is very surprising, because at this resolution the tilling patterns cannot be seen—unlike for the resolution used in this thesis.

Warner and Steinmaus [WS05] propose a method for distinguishing orchards, vineyards, and other kinds of fields using autocorrelograms. For each pixel, the autocorrelation is calculated in the horizontal, vertical, and two diagonal directions. If a consistent periodicity is detected in at least one of these directions, the pixel is deemed to belong to an orchard or vineyard. A hard threshold on the period is used to distinguish between orchards and vineyards. This partially inspires the method developed in this thesis for orientation and periodicity estimation, described in [TS05b] and, in more detail, in chapter 6.

Complexity

Complexity is another important component of textures. It is often used to distinguish between natural and artificial objects, because artificial objects are usually more ordered. Dubuc and

Zucker [DZ01a, DZ01b] give a study of visual complexity based on human perception. Most complexity indicators use estimators of the fractal dimension, or entropy measures.

The fractal dimension is often used as a measure of complexity [PNHA84, TLT00, TA95], but we have found it not to give good results for our application.

The grey level distribution of an image is used in [CMV02] to detect artificial objects in natural environments. Baillard [Bai97] uses the entropy of the histogram of gradient directions, and Guigues's variant [Gui00] uses a weighted sum of gradient directions based on Medioni's tensor voting; both take into account the amount of texture present, giving in effect three classes: flat texture (or untextured), unordered orientations (corresponding to natural textures) and ordered orientations (corresponding to artificial objects).

Gray-level co-occurrence matrices [HSD73] and various statistical indices extracted from them are also commonly used as texture indices. They indicate, among others, the self-similarity of an image at different scales and directions. Steinnocher, Kressler and Köstl [SKK03] and Steinnocher [Ste97] use them to derive a simple but useful detector of urban areas: They calculate the Inverse Differential Moment of the grey-level co-occurrence matrices over a region, and obtain the average, and the maximum difference, across orientations; both values are low for urban areas.

For texton-based textures, Grigorescu and Petkov [GP03] present a method using Rényi's generalized entropies. Standard entropy is also a common choice.

An example of the use of complexity measures is given by Shackelford and Davis [SD03], who present a land cover classification on urban area, using 4-channel, 4 m resolution multispectral Ikonos images, both raw and pan-sharpened with 1 m panchromatic data. A standard maximum-likelihood pixel-wise classifier gives 81% to 87% accuracy for a classification into "water", "road", "building", "grass", "tree", "bare soil" and "shadow" covers. The addition of the gray-level histogram entropy increases by 10% the accuracy for classes "grass" and "tree". The use of a feature based on the size of groups of pixels spectrally similar to the current pixel increases by 5% the accuracy for classes "road" and "building". These new features are used selectively only for the appropriate classes, bringing global accuracy to 93% and 96% (for the two tests sites). Of interest is the use of gray-level histogram entropy based measures for distinguishing between grass and trees. Zhang *et al.* [ZWGS03] show the use of similar textural features from panchromatic images for urban land cover classification. They use grey-level co-occurrence matrices, the number of different gray levels, and the density of image edges as textural features to classify pixels into "open space", "low-rise", "old multi-storey", "new multi-storey", "high-rise", "roads", "construction", and "agriculture", with 72% accuracy.

Other texture detectors

There are other texture descriptors which cannot be easily classified as orientation estimators or complexity estimators.

The dot-pattern selective cell operator [KP00] is a biologically motivated texture operator specialized in the detection (and characterization) of dotted textures. Another biologically motivated operator given in [KP99] detects gratings, that is, sets of close parallel lines.

Palmason, Benediktsson, and Arnason [PBA03] (and see also [CBP03]) use morphological transformations for land cover classification in urban areas. Different morphological structuring elements allows them to distinguish, for example, between wide roads and narrow streets. Hyenne [Hye03] uses mathematical morphology to detect individual image trees in high-resolution aerial and satellite images.

Combining rotational-invariance (or orientation detection, if needed) with multi-resolution analysis, Ojala, Pietikäinen and Mäenpää [OPM02] propose a texture classification method

using *local binary patterns*: These are a reduced number of local patterns whose statistical frequencies are shown to be sufficient for texture classification.

Santamaria, Bober, and Szajnowski [SBS04] present a texture descriptor based on statistics of the binary threshold of the one-dimensional signal obtained by scanning an image through a Peano curve. It gives good results for an approach bearing, at first sight, little promise. Hafiane *et al.* [HCSZ04] follow a similar approach.

Haindl and Havlíček [HH02] use causal autoregressive random field models to synthesise textures, but their method can also be used for analysis and indexation.

2.1.3 Joint use of colour and texture

Colour and texture are usually considered separately, and most texture extraction algorithms work on grey-level images only.

The interest for integrating colour and texture stems from the observation that considering texture as purely an intensity-based structure disregards, for instance, coloured texture primitives with constant intensity [Pal04]. However, we have found that in our context we do not encounter such constant-intensity textures, and therefore the interest of using colour-aware texture parameters is greatly reduced. In addition, Mäenpää and Matti Pietikäinen [MP04] compare several classification algorithms that use colour and texture either separately or jointly, and conclude that colour and texture are separate phenomena that should be treated separately. Nonetheless, there are many articles on using colour images for texture analysis.

Singh, Markou, and Singh [SMS02] review some texture parameters that take colour into account.

Nicolás, Yzuel, and Campos [NYC00] propose a way of using colour (or more generally, wavelength) as a third dimension in the Fourier transform, and use it to extend a correlation-based pattern recognition method. This trend continues in [Pal04], who extends co-occurrence matrices [HSD73] to deal with multi-channel images. Jakomulska and Stawiecka [JS02] and Zhang, Franklin, and Wulder [ZFW04] study the joint use of the raw multispectral bands and of variogram-derived textural features for land cover classification.

Taking the opposite approach, Mäenpää, Pietikäinen, and Viertola [MPV02] try to better separate colour and pattern from a coloured texture.

Another way of integrating colour and texture is that of Corso, Dewan, and Hager [CDH04], who present a segmentation method using colour and texture, but not both at the same time like many other articles, but rather at each area the one that is most appropriate.

Finally, another option is to use texture features computed not on raw intensity data but on one transformed radiometric channel. Greenhill *et al.* [GRH⁺03] discuss the use of two indicators, the weighted mean patch size (for vegetation patches) and the lacunarity, in analysis of high-resolution imagery, in this case NDVI data at 4m resolution obtained from Ikonos images. They suggest that their use could enable urban planners obtain a more sustainable city growth.

2.1.4 Shape

In addition to colour and texture, it is possible to use the *shape* of a region in a segmentation or classification process. For example, we could make the segmentation process tend to give as solutions regions of a given shape, or we could use the shape as an additional input for classification. We could use the available cadastre data to learn the “expected” shape of regions—since agricultural regions resemble in shape cadastre regions in the same area. To do this, we need a way to describe the shape or outline of a region.

Zhang and Lu [ZL04] and Lončarić [Lon98] give long reviews of shape description techniques, both contour-based descriptors and region-based descriptors, and ranging from simple descriptors such as area, circularity, or major axis orientation, to complex grammar-based structural descriptions. Comparison with a much older review by Pavlidis [Pav78] reveals that unfortunately no major breakthroughs seem to have occurred in shape description.

The need to complement pixel classification with shape models in order to obtain a better segmentation can already be found in some early papers. Fua and Hanson [FH85, FH87] derive shape hypotheses for an image segmentation; these hypotheses are used to predict the corrected outline for an object. They claim to use generic shape models, as opposed to application-specific, although in practice their models are tuned to the detection of man-made objects in aerial imagery. These models are defined using a grammar of image primitives (edges, pixels, . . .) and relations (inline, corner, T-junction, parallel, . . .). See also [Jib02] for a more recent approach of shape grammars for man-made object recognition in aerial images.

Still, since this description of shape will be used for classification and comparison, not for regenerating the shape from it, we can use the many *compact* shape descriptors which describe a shape with only a few values. Simple ones are the perimeter, area, centroid and higher-order moments, maximum and minimum distance to centroid, diameter, maximum chord, maximum arc length, major and minor axes, thickness, and topological descriptors [dFCC00], and compactness, rectangularity, and orientation. Peura and Iivarinen [PI97] examine the efficiency of some of simple shape descriptors (convexity, ratio of principal axes, compactness, circular variance and elliptic variance) and claim that there is, in general, no need to use more complex and computationally expensive shape descriptors.

2.2 Segmentation

To segment an image means to partition it into homogeneous regions. Ideally, this homogeneity should be semantic homogeneity, such that different objects (or parts of objects) define different segmentation regions. How to do this kind of segmentation is well beyond the state of the art, and in practice segmentation algorithms produce regions which are homogeneous in colour, texture, or other low-level visual attributes.

In this thesis, segmentation has two functions. First, if cadastre data is not available, it provides a partition of the image into regions which can be classified by per-region classifiers. Second, if cadastre data is available or the system is updating an old classification, it locales important image edges onto which the cadastre or old class map should be registered.

2.2.1 Single-scale segmentation

Most image segmentation methods take a parameter (usually a threshold for the dissimilitude between adjacent regions) and output a partition of the image. This usually translates into a single scale analysis of the image: small thresholds give segmentations with small regions and much detail, large thresholds give segmentations preserving only the most salient regions.

The literature is full with literally hundreds of different segmentation methods. See [PP93] for a review of early methods.

Simpler segmentation methods try to separate foreground and background using a single global threshold [Ots79] or locally adapted thresholds [Alt95]. Most methods, such as region growing or split-and-merge, operate directly on the image pixels; others, such as the snakes-like [SKS95], start with an edge extraction step which tries to capture discontinuity loci. Most methods are fully automatic —once their parameters have been tuned—, but some are inter-

active [LE02]. Most methods have an implicit model of what a segmented image should be; only a few explicit it, using for example the Mumford-Shah functional [MS89], extensions to it [LSLH02], or many others.

Furthermore, while most methods deal with grey-level images only, there is some work on colour image segmentation (see [CJSW01] for a study of this). Other recent examples include [MP00] for a direct segmentation of colour textures by clustering through multi-scale probabilistic relaxation [RHZ76,GG84,FY97], [HEMK98] for a multidimensional segmentation using watersheds, and [Wan98] for using probabilistic relaxation for image segmentation.

2.2.2 Edges and regions

There is an opposition, in single-scale segmentation methods, between edge-based and region-based approaches. Freixenet *et al.* [FMML04] propose a method integrating both which also uses, at the same time, colour and texture: colour and texture perceptual edges—which are badly localized because of texture’s non-locality—are used to place region seeds in homogeneous areas, that is, far away from those edges. These regions are then grown using also the colour and texture properties of its seed areas, providing closed and more precise region edges.

Mueller, Segl, and Kaufmann [MSK04] present a segmentation technique based on both edges and regions for extracting agricultural fields in high-resolution satellite imagery. This has some similarities to the technique used in this thesis, although in their case the application-specific part of the algorithm—looking for straight boundaries, homogeneous large regions—is mixed with the main segmentation algorithm, and in our case they are decoupled—region and shape energies, on the one hand, and the scale-sets climbing segmentation algorithm, on the other. They cite the program *eCognition* by Definiens Imaging as one of the reference commercial systems performing similar tasks, but note that *eCognition* does not take as much advantage of edge information for segmentation.

In [MFM04], several methods are proposed for detecting boundaries in an image. Boundaries are high-level features corresponding to separation between objects or parts of objects, in contrast to edges which are low-level features not related to the semantic interpretation of the image. Edges are usually computed as a first step towards boundary detection, but the authors argue that most current edge detectors are not appropriate for this task. In contrast to our application-specific features that we use to segment an image into agriculturally homogeneous areas, they present image features that are applicably in the general case.

Edge and line detection algorithms usually work by finding alignments of certain structures. In contrast, Newsam [New04] proposes to look for lines along with the distribution of pixel values is significantly different from the distribution in the whole image.

There is sometimes the need to incorporate a priori spatial grouping information in the segmentation. In this thesis, for example, the cadastre or old segmentation give clues of probable image regions. [YS04] propose a more general framework for including a priori grouping preferences into the segmentation process itself. The goal function which measures the goodness of grouping a set of pixels together—which exists, implicitly or explicitly in all segmentation algorithms—is extended to deal with the knowledge that certain pairs of pixels must belong to the same image segment.

2.2.3 Markovian techniques

Many segmentation algorithms use a Markovian approach. These are typically very slow algorithms that give very high quality results, supported by a solid mathematical foundation. Deng and Clausi present in [DC04] a classical segmentation method based on Markov Random

Fields (MRF), where the weight of the regularization term varies along the algorithm's iterations. Another MRF-based segmentation is [SPZ03], which uses not a flat but a tree-structured MRF for improved performance. This tree-structured MRF is used in [PSZ04] for a segmentation and classification application, reporting unimpressive global accuracies of 73.3% for a 3-channel SPOT image with 8 terrain classes, including 4 kinds of field, forests, urban areas, bare soil, and water.

2.2.4 Multi-scale segmentation

There is a problem with the single-scale segmentation methods reviewed above. As it has been said since the early times of image analysis (and also [Mar82]), meaningful structures and objects in a given image appear at different values of the controlling parameter (or in other words, at different scales). Sometimes, a given portion of an image has different meanings at different scales (roof tile, roof, house, ...). Also, even if in some cases there exists a single, correct scale of analysis, this scale can usually only be obtained from high-level analysis, which happens after, not before, the segmentation phase.

With that in mind, several authors have proposed *multi-scale* segmentation algorithms [JM92, KLM94, SG00]. Vanhamel *et al.* [VPS03, VKS03] propose a hierarchical segmentation algorithm which handles colour *and* texture and suggests which level in the hierarchy is the most significant. Chen *et al.* [CPMR04] present a segmentation system using a perceptual colour space and texture orientation information, with a multiscale analysis that handles competing orientations.

Guigues' algorithm [GLMC03b, GLMC03a, Gui03, Gui04] is particularly well adapted to our problem, because it makes both the segmentation criterion and the scale parameter explicit, and because of implementation issues. Because the segmentation criterion is explicit, we can easily incorporate application-specific texture, colour and shape attributes into the segmentation process. The scale parameter, which for classical segmentation methods is an input parameter, becomes an output in multi-scale methods. It is then up to the post-segmentation stages to either select the most appropriate scale, or to analyse the results as a whole.

A related approach was earlier proposed by Zhu, Lee, and Yuille [ZY95, ZLY95, ZY96]: a region growing and region competition algorithm is proposed which is based on energetic, Bayesian, or MDL criteria—like in Guigues' work [Gui04]—which find a balance between data fit and model complexity. This algorithm is dependent on the correct positioning of a number of initial seeds.

The work by Neelamani *et al.* [NRC⁺00] can also be seen as a precursor to Guigues' algorithm [Gui04]: they use the Minimum Description Length framework (MDL) to combine segmentation complexity (and thus take shape into account) with data fit (in this case, texture is considered). An iterative dynamic programming algorithm is used to obtain near optimal segmentations for fixed scales of analysis—dyadic downsamplings of the source image. Also from an MDL framework—although in a more general sense—Guigues manages to find true optimal segmentations, for all values of the scale parameter.

2.2.5 Range image segmentation

Although we will only use visible and near-infrared data in this thesis, it may be interesting, in further research, to use height data in the form of a digital terrain model and a digital surface model. The special nature of this kind of data calls for specific segmentation methods, for example [HŽ98, BHS98, JK98], regularization techniques [MKIZ98, Gho00] and fusion algorithms [HSIW96].

2.2.6 Input selection and evaluation

If several input image channels are available —as in this thesis, where many derived colour channels and many textural features are calculated for each image— the need arises to select the most appropriate ones as input to the segmentation algorithm.

Vansteenkiste *et al.* [VSGP04] test several colour spaces and texture features for terrain classification (actually, segmentation) in high resolution satellite images and conclude that texture does not add any significant advantage.

This input channel selection usually requires the definition of quality metrics for segmentations, that is, methods for evaluating the quality of a segmentation.

Segui Prieto and Allen [SPA03] compare several similarity metrics for edge images, which are often used to evaluate the quality of a segmentation, and propose their own metric intended to correct the drawbacks of the others. Chabrier *et al.* [CEL⁺04] study the performance of several criteria used for *unsupervised* evaluation of image segmentation —that is, they are used to determine the best segmentation without using a ground truth. Meas-Yedid *et al.* [MYGM⁺04] propose an improvement to Liu and Yang's criterion [LY94] for automatically evaluating the quality of an image segmentation, and use it for automatic colour space selection. We have not used these criteria because a ground truth is available in our case, and because these criteria apply to single-scale segmentations only.

2.3 Registration

The second processing step described in this thesis is the registration of the cadastre graph onto the image. With this we obtain large regions which are, in most cases, homogeneous in terrain type.

2.3.1 Graphs and graph matching

Both the cadastre and the image —or, actually, the image segmentation— can be represented as graphs. For the image segmentation, there will be a graph face for each segmentation region, and the edge between two faces will have a weight corresponding to the similitude between these two regions. We can view the problem of registering the cadastre onto the image as a graph matching problem.

In graph matching, starting with two graphs representing the same physical reality, the goal is to label —or otherwise match— the edges and nodes of one graph and those of the second graph, so that the elements referring to the same part of that physical reality match [HD98,GB03]. Walter [Wal98] uses a more informal approach. Variations include allowing unmatched nodes and edges, and varying degrees of latitude in the difference between the graphs. A related but more theoretical issue is that of *graph isomorphism*, where only topology is considered and unmatched elements are not allowed. See also the early work by Shapiro and Haralick [SH81] on graph matching by relational homomorphisms, where the problem is approached with a knowledge representation focus rather than with an image analysis one. Conte and others [CFSV03] give a review of graph matching applications in pattern recognition and image processing.

2.3.2 Rigid registration

In rigid registration problems, the goal is to find the rotation, translation, and, eventually, scaling factor required to transform a data set—a set of points, a set of lines, an image, . . .—so that it matches another. It is a simpler case of registration than the one used in this thesis.

Wi, Pan, Prinet, and Ma [YPP04] use frequential features obtained from a binary image of roads (obtained through an improvement of a binary threshold that selects only long straight objects) by Fourier techniques. Then they compare this frequency information to that obtained from GIS data in order to obtain the scale, rotation and shift required for registering both data.

Umeda, Godin, and Rioux [UGR04] give a method for registering a colour or intensity image and a range image, within the framework of rigid transformations, by using the intensity image that ranging sensors calculate along with the range image.

Boughorbel *et al.* [BKAA04] introduce a new energy function for the rigid registration of point sets, which allows for the use of standard optimization techniques without requiring an initialization close to the optimum. Leemans and Sijbers [LS04] do rigid registration of space curves. Chen and Varshney, with Arora [CVA03] and with Luo and Lin [mCVLhL04] use mutual information criteria for rigid registration of images. Chow, Tsui, and Lee [CTL04] use genetic algorithms to register two surfaces.

2.3.3 Non-rigid registration by deformation

Some instances of graph matching problems are actually non-rigid registration problems. In non-rigid registration problems, the goal is to find the non-rigid transformation (a deformation) in a transformation class (usually a subset of C^2 functions $\mathbb{R}^2 \rightarrow \mathbb{R}^2$) that best converts the working image to the reference image, according to some dissimilarity measure; matching is then trivial. See Cachier *et al.* [CBD⁺03] and Goshtasby *et al.* [GSST03] for some reviews of non-rigid image registration methods; as they say, most of the work in this domain focuses on medical imaging.

Transformations classes are usually a subset of C^2 functions $\mathbb{R}^2 \rightarrow \mathbb{R}^2$. For example, Viglino and Guigues [VG02] match cadastre graphs to terrain edges by finding the best global transformation from one to the other among the class of polynomial transformations of a certain degree.

Goshtasby [Gos88] wrote an early paper on non-rigid image registration. The author proposes the use of two surface splines (instead of the more usual polynomials) to represent a mapping function—to transform one image into the other.

Much work in this area deals with the registration of sets of points: Chui and Rangarajan [CR03] propose a non-rigid registration method for sparse sets of points which could be used in some instances of graph matching problems. Li, Meng, and Holstein [LMH04] show a simple and elegant method for registering two point sets of small cardinality, one of which is labelled; non-rigid deformations are supported but missing points—points which exist in only one set—are not. Luo and Hancock [LH03] propose a complex framework for alignment and correspondence of point sets, both using rigid transformations (rotation, translation, and scaling), and non-rigid transformations (arbitrary displacement for each point, not just a global continuous transformation); points are not attributed and edges are not considered, but points are grouped into landmarks.

Eugenio and Marquès [EM03b] perform automatic georeferencing of satellite images through the use of a registration procedure: after removing cloud-covered areas in an image, reliable edges are detected. These edges are matched to those in a geographical database for the imaged area using an iterative procedure which finds the optimal affine transformation.

Noblet *et al.* [NHHA04] present a topology-preserving non-rigid registration method. Here, as in most work on non-rigid registration it is assumed that the deformation must be continuous, which is not the case for the registration methods presented in this thesis.

2.4 Classification

The third processing step described in this thesis is classification. In a classification, each pixel or region in an image or other kind of dataset is assigned a *class*. Classification can be *supervised*, or *unsupervised*. In the first case, the classification system is given training data so that it can establish a link between classes, which have a precise meaning (such as *urban*, *water*, *deciduous forest*, ...) and the characteristics of data belonging to that class. In unsupervised classification similar data are grouped, and each such group is deemed to be a class; these classes have no semantic value, and there is usually the need for a post-processing stage to attribute meanings to each class. There is a partial overlap between classification and segmentation, in that some classification methods use spatial constraints [Pha02], and that some segmentation methods simply segment an image into the regions—possibly disconnected—defined by a classification algorithm [FP03].

2.4.1 Clustering

Many popular unsupervised classification methods involve *clustering* image pixels into a small number of spectrally homogeneous classes (without any *a priori* meaning), and then manually labelling each class. This is called *clustering* because one is actually looking for clusters of data points according to their spectral coordinates (as in a spectral scatter plot). One commonly used clustering method is *fuzzy c-means* [Bez81, DR01], but there are others [MMP00]. Clustering is a well-known and established technique. Berkhin [Ber02] gives a very broad review of clustering techniques.

Many segmentation methods use clustering techniques:

In an unsupervised land use classification method by Wu *et al.* [WPPM04] a new spatial mean shift procedure is used to obtain points in the middle of large statistically homogeneous regions, edge points, and points in small statistically homogeneous regions. Region growth is then used.

Virmajoki and Fränti [VF04] propose a divide-and-conquer approach for creating neighbourhood graphs that renders the computation time of agglomerative clustering techniques acceptable.

In the context of clustering, Fred and Leitão [FL03] observe that the dissimilarity increments between neighbouring patterns within a cluster have a smooth evolution, which follows an exponential probability law, whereas inter-cluster dissimilarities behave differently. Using this criterion they propose a hierarchical agglomerative clustering algorithm which can handle both dense and sparse clusters, unlike traditional hierarchical clustering algorithms. Additionally, the number of clusters is found automatically.

One common problem with clustering methods is that the number of classes must usually be specified beforehand. An alternative is to let the algorithm select the number of classes that maximizes the *goodness of fit*—that is, how well a model or a simplification of data reproduces the original data—, as determined by the distance from the data points to a canonical member selected for its class. But this is not enough, since maximizing the goodness of fit alone produces a clustering with one different class for each individual data point. To avoid this we have to add a penalty on the number of classes, that is, not only take into account the *goodness*

of *fit* of the representation, but also its *simplicity*. It is not trivial to take these two magnitudes into account at the same time.

There exist many theories that deal formally on how to compare goodness-of-fit and simplicity [WG83, OH94, OB94, GLMC, BCG98], including Minimum Description Length (MDL) or Minimum Message Length (MML), also known as Minimum Encoding. Evaluating the goodness-of-fit is a problem on its own. In [AS02] we find a comparison between different goodness-of-fit tests. Minimum Encoding is also used by Guigues [GLMC] for his hierarchical segmentation system. In [KDN⁺95] we find an MDL-based segmentation system.

Many clustering and classification schemes assume that each individual cluster follows a normal (Gaussian) distribution. However, this is not realistic for remotely sensed images, due to the high variability and mixture of reflectance found in most land cover types [SDS99]. Schöll and Schöll-Paschinger [SSP03] propose an alternative clustering method, based on restricted random walks, which does not assume normality; it returns a hierarchy of clusters, thereby also avoiding the problem of finding the optimal number of clusters. Pina and Barata [PB03] use mathematical morphology to obtain non-Gaussian clusters, with substantial improvements of classification rates. Fonseca [Fon03] studies the effect of applying normalization methods, that transform data so that clusters do have normal distributions. Brankov *et al.* [BGYW02] use normalized cross-correlation instead of the usual Euclidean distance to evaluate distances in the cluster space. Khotanzad and Chen [KC89] and Khotanzad and Orlando [KH03] detect maxima in histograms for their simple but elegant *peak climbing* clustering algorithm. Bachmann, Ainsworth, and Fusina [BAF05] obtain a manifold which tracks the hyperspectral pixel values better than the standard \mathbb{R}^n , and use as distance the distance on that manifold instead of the Euclidean distance on the whole hyperspectral radiometry space.

Another research area in clustering centres on the extraction of *community structure* in networks. Some networks (or graphs), such as social networks, telecommunication networks, or protein interaction graphs, exhibit the property that the vertices of these networks are often clustered into tight groups: there is a high density of edges joining vertices in the same group, and a low density of edges joining vertices in different groups. Newman [New03b, New03a], Newman and Girman [NG03, NG02, GN02] and other authors [RCC⁺03, MNL03, HH03] have studied these networks and proposed efficient algorithms that obtain this cluster structure in a hierarchical way. These algorithms can be applied to clustering problems where there is a weighted neighbourhood relation between data points—for example, because one data point is extracted for each face in a segmentation graph. Radicchi *et al.* [RCC⁺03] also defines a calculable notion of *community* which can be used in a hierarchical clustering process based on community structure to determine which nodes in the hierarchy actually correspond to communities and which are over-segmentations. Newman [New03b] gives a long review of clustering-like phenomena occurring in real networks.

2.4.2 Supervised methods

In supervised classification methods, a subset of pixels (the *training set*) is manually labelled, so that the class for each training pixel is known. Then, after a training or model estimation process, the algorithm is able to label the remaining pixels automatically.

A simple supervised method similar to clustering starts with the operator selecting one set of representative pixels for each class, and manually labelling them. The algorithm then labels each remaining pixel with the label of the nearest—in a spectral sense—manually-classified pixel, giving a supervised clustering. Other, more advanced, supervised classification methods include decision trees [Qui93], neural networks and support vector machines (SVM).

Supervised classification is in general very well understood, but it may give difficulties in some applications because it requires adequate amounts of training data, that is, images or image

subsets whose class has been previously decided by a human interpreter. There has been some work on techniques to reduce the required quantity of training data, for example [VKR03].

Supervised classification can be divided into discriminative and model-based. In the discriminative approach, the internal model created in the training phase is only good enough to distinguish between data points of different classes; for example, it may contain a set of appropriate thresholds. In the model-based (or generative) approach, the internal model describes training data well enough that it can actually be used to generate synthetic data. Milgram, Sabourin, and Cheriet [MSC04] give an interesting description of the difference between discriminative and model-based classification, and provides a hybrid approach that combines the advantages of both.

Probabilistic classification

In probabilistic supervised classification, the internal model is a probability model, and data points are classified according to probability theory. That is, the class assigned to a datum is the *most probable* class, in a clearly defined mathematical sense.

Smits, Dellepiane, and Schowengerdt [SDS99] state that maximum likelihood and Bayesian classification —two kinds of probabilistic classification— is computationally expensive and assumes normal distribution —actually, they probably mean that to avoid a high computational cost normal distribution is often assumed—; while this may have been true at the time the article was written, it is certainly no longer the case: in this thesis we use Bayesian classification with arbitrary distributions, both parametric and non-parametric, with acceptable computational costs.

Yuille and others [YC00,YCWZ01] study in detail a particular and interesting aspect of Bayesian inference and maximum a posteriori estimation, namely, for what kind of problems it is theoretically possible to obtain a correct classification, and for what kind of problems it is impossible to do so, independently of the actual algorithm used. They find that there is an order parameter K that describes the difficulty of the problem —for example, how much classes overlap each other— and that, in certain settings, the probability of correct classification shows a phase transition for a certain value of K : beyond a certain level of difficulty, it becomes completely impossible —in a probabilistic sense— to obtain a correct classification, irrespective of the algorithm used. They suggest that most performance measures can be reformulated to be equivalent to K , and apply their results to the task of detecting a road in noisy images.

Dempster-Shafer Theory of Evidence [Sha76] (or possibility theory), another mathematical framework for dealing with uncertainty, is sometimes used in multiple-classifier settings to merge the possibly conflicting results of several classifiers. Lein [Lei03] emphasizes that it can also be used to assign probabilities to sets of classes, instead of single classes.

2.4.3 Markov Random Fields

The output of classification systems is usually imperfect, with “noise” in the form of outlier pixels. A popular regularization method uses Markov Random Fields (MRFs) initialized with the result of the classification [DMZB96]. [PDZ00,MDZ96,YBG97] are good examples of their use for classification, and [FS02,WF02] for segmentation. Kato, Pong and Qiang [KPQ02] use MRFs to combine texture and colour for segmentation. Along with their related Markov Chains and Hidden Markov Models (HMMs) [Rab89] they are sometimes used for the classification itself. However, MRF-based methods are usually very slow.

Mottl, Dvoekno, and Kopylov [MDK04] give a variation on common Markov Random Fields (MRF) based techniques for classifying data with spatial relationships. In their setting, MRFs

are used only to model dependencies between the classes of adjacent pixels, and the classification of the whole image is, as usual, calculated at once.

2.4.4 Application-specific methods

Another, complementary approach, is to look for image features typical of a certain land cover, or to use external expert knowledge. In addition, for practical applications, it makes sense to use specialized versions of algorithms specially adapted to the kind of terrain classes found in each problem. This section gives some examples of such application-specific methods.

Zhang, Prinet, and Ma [ZPM03] use radiometric entropy to detect water bodies such as lakes, sea areas and large rivers (but not small rivers). Some authors [SKK03, Ste97, AZW00, LDZ98, Bai97] use texture to detect urban areas.

Methods for detecting trees in aerial images are given in [DR96] and [PKBP02], among many others. For example, Straub [Str03] presents a method for detecting trees using a Digital Elevation Model (DEM) and an image. A shape model is fitted onto the DEM at different scales, and tree hypothesis are then verified using shape features of the tree crown boundary and biological activity indicators obtained from the optical image.

Fayek and Wong [FW96] give an algorithm for extracting buildings from aerial topographic maps, akin to height or range images. Giada *et al.* [GdGES03] use morphological operations to detect and count tents in satellite imagery of a refugee camp. Turker and San [TS04c] use shadow detection to determine whether a building has collapsed or not, in aerial images taken after an earthquake. Miyoshi, Li, and Kaneda [MLK04] automatically locate buildings in scanned topographic maps by finding closed polygons with certain attributes. Roberts *et al.* [RDG⁺03] use hyperspectral data to evaluate fire hazard in shrub areas.

Robbez-Masson *et al.* [RMWAB01] use frequential information to classify among several vineyard types, and give textural descriptions that may allow discriminating between vines and orchards. The Vinident study [Aqu97, RANB98, MAR98] also gives detailed textural and radiometric descriptions of vineyards and also some identification methods. Wassenaar [Was01] gives a complete Fourier-based vineyard detection system as well as a vine growth state and vegetation cover estimation method. Ranchin *et al.* [RNA⁺99] use the local density of the edges given by an edge extraction algorithm to automatically detect vines.

Thomas, Hendrix, and Congalton [THC03] start with a pixel-wise maximum-likelihood classification of 4-channel (near infrared, red, green, blue), 1 m resolution images into “water”, “bare ground”, “vegetation”, “pavement”, and “rooftop” classes, which is then post-processed with hand-constructed rules using spatial relationships and ancillary data (such as digital elevation models, zoning data, or road centrelines) to bring the classification accuracy up to 79% from an initial 45%. This exposes the interest of complementing a statistical classification step with expert-produced classification rules for the most confused classes.

Simonneaux and François [SF03] use NDVI time series to discriminate between several kinds of crops. Lacroix *et al.* [LIH⁺04] uses Gabor filters and NDVI to determine built areas in high-resolution satellite images (Gabor filters detect structured and textured areas while NDVI filters out vegetation areas), to use in a change detection system. Gallo *et al.* [GEYR04] show how the NDVI and other vegetation indices can not only be used to estimate the vegetation cover density, but also for assessing the urban growth, although images depicting the night-time light emission are more useful for this task. Di Bella *et al.* [DBFR⁺04] use several vegetation indices (including NDVI) obtained from multispectral satellite images to obtain production data for pastures.

2.4.5 Dimensionality reduction

Most often, classification methods are used on hyper-spectral data, that is, on images with high spectral resolution (see Shippert [Shi04] for an informal presentation of the advantages of hyperspectral imagery, what kind of problems can be solved with it and general advice for manipulating such data). Hyperspectral imagery gives enough spectral information to distinguish between spectrally similar materials, and is also often used to distinguish between different crops and tree species.

However, high spectral resolution, while often needed to distinguish among some land cover types, also brings two problems: First, there is usually too much data, which causes higher computational costs. Second, and more important, the many classification methods which rely on inverting the data's covariance matrix fail because that matrix is often non-invertible. Although hyperspectral imagery is not used in this thesis, because many derived input channels are available some of its problems also apply to our work. The solution is to use *dimensionality-reduction* techniques to extract a discriminant subspace from the original sample space, for example Principal Component Analysis (PCA, or Karhunen-Loève transform) [DH73], Singular Value Decomposition (SVD) [Mar87], greedy search [MBB02], Kohonen's self-organizing maps (SOM) [Koh01] or techniques based on Mutual Information [AAD02]. Note that, in spite of PCA's popularity, it may not be appropriate for remotely sensed data [CB03]. Bruce *et al.* [BYKC03] gives a brief review of dimensionality reduction techniques.

These techniques generate reduced-dimensionality vectors by combining all the components of high-dimensionality input vectors. In our case, the actual calculation of all input features is computationally very costly, so it would be preferable to reduce dimensionality by, instead, selecting a few of these features. In remote sensing circles, this is known as *band selection*. These techniques can also be applied in our case, although our "bands" are texture features and derived colour channels: Bajcsy and Groves [BG04] present a large number of unsupervised and supervised methods for band selection in hyperspectral imaging for the purposes of dimensionality reduction for classification, and exhaustively tests combinations of these methods. Only band selection is performed, not band combination as in principal component analysis (PCA). Therefore, this procedure could be applied to the selection of texture features in our problem. We have however stuck with the simpler Stepwise Forward Selection (SFS) because of the complexity of implementing the described algorithm compared to the expected benefits, both for the actual results and for the scientific interest of this thesis. The Stepwise Forward Selection procedure starts with an empty set of features and adds at each iteration the feature that makes the cumulated set have the lowest classification error, until a stop criterion. See [UG04] for another application of SFS.

Miasnikov, Rome, and Haralick [MRH04] present a dimensionality reduction method for data sets in which clusters may have arbitrary shapes, based on the observation that, when projected on reduced-dimensionality spaces, clusters of arbitrary shapes usually become Gaussian. Fortuna and Capson [FC04] propose a classification method that iteratively combines a Principal Component Analysis (PCA) or Independent Component Analysis (ICA) of the data with a Support Vector Machine (SVM) classifier. Outliers are detected in the classification results and the PCA or ICA projection axes are modified in order to better separate the two classes. Chen and Zhu [CZ04] extend PCA by first partitioning the data set, and then computing the PCA for each partition separately, giving much better classification performances than standard PCA.

2.4.6 Evaluating the quality of a classification

In order to choose the best classification method for a given application, or to select the optimal parameter set, it is necessary to evaluate the quality of a classification.

Smits, Dellepiane, and Schowengerdt [SDS99] explore the problem of quality assessment of land cover classification algorithms. The authors conclude that evaluation methods based on confusion matrices and the KHAT statistic are the most suited for comparing classifiers. It is interesting to note that 90% accuracy is considered the goal for many remote sensing projects. In this thesis better figures are obtained.

Liu, Gopal, and Woodcock [LGW04] discuss the computation of confidence measures on classifications. They highlight that attempts to derive uncertainty information in remote sensing are still in their infancy, and propose a simple method for combining two classifiers which already provide confidence indicators. They do not discuss the issue of obtaining these per-classifier confidence indicators, although they cite two works by Foody *et al.* [FCTW92] and Canters [Can97] where the a posteriori probability of class membership in a maximum likelihood classifier is used as a measure of classification quality for each pixel, not unlike some of the confidence measures presented in this thesis.

2.4.7 Per-region classification

Tuominen and Pekkarinen [TP05], exploring several textural features and transformed radiometric channels to determine forest characteristics (such as tree height or volume) from high-resolution colour and near-infrared imagery, note that pixel-by-pixel analysis (using radiometry only) “is not applicable to high-resolution imagery because a single pixel is small in relation to the object of interest, i.e. a forest stand, and therefore it does not adequately represent the spectral properties of a stand. Additionally in aerial photographs, the spectral properties of the objects are dependent on their location in the image” (because of the hot spot effect, of mutual shadowing, and of the exposure fall-off effect). They show that most textural and radiometric features are, however, correlated.

Townshend *et al.* [THK⁺00] also argue against classical per-pixel classification, but for different reasons: Radiometry values for an image pixel are obtained from incident light from a terrain patch which is larger than the ground pixel. That is, terrain adjacent but not contained in a ground pixel is represented in the corresponding image pixel. This is typically described by means of a *modulation transfer function*, or its inverse Fourier transform, the *point spread function*. In estimating terrain contents, they show that a simple image downsampling increases the accuracy because the effect of adjacent pixels is reduced.

Although not mainstream yet, per-region classification (as opposed to per-pixel) has already been used in some articles:

De Wit and Clevers [DWC04] present a methodology used to produce a crop map of the Netherlands. Multi-temporal per-pixel NDVI profiles were extracted from imagery from several sensors (Landsat TM5, Landsat TM7, and IRS-LISS3). Field boundaries were extracted from a vector topographical database; although the authors had considered using image segmentation to obtain crop boundaries, they concluded that it was not currently feasible—in the system presented in this thesis, imprecise vector data from an external database is combined with image segmentation to obtain more precise crop boundaries—. Comparison of per-field NDVI averages and standard deviations against a set of thresholds is used to pre-classify each field into “evergreen”, “early crops”, “late crops” or “bare soil”. The authors also give a good introduction to per-field classification: They argue that there are two main problems with per-pixel classification: first, that within-field reflectance variability, due to variations in soil moisture conditions, nutrient limitations, or diseases, frequently causes part of the field to be classified incorrectly; second, that pixels at the boundary of two fields have a mixed radiometry which often results in it being classified not in the class of one of the adjacent fields, but in a third class—this latter problem does not affect our system due to the high spatial resolution of the source images—. The idea behind per-field (or per-object) classification is that the image

is divided into segments (objects) using knowledge of the “real-world” objects on the Earth’s surface; the class of the object is then decided based on the collection of pixels it contains. They notice two trends on per-field classification. In the first, per-field statistics are computed and used for classification. Typically, these are averages and standard deviations for each channel within a field. In the second, a per-pixel classification is performed, and then a per-field class is selected by majority voting. In this thesis a third approach will be introduced.

As explained in Richard’s excellent short review of remote sensing [Ric05], per-field classification has also been used on RADAR (SAR) images, motivated by the very low signal-to-noise ratio of these images, and the reduction in error rate obtained by applying averaging techniques over pixels in each field [FSS⁺03].

Guindon, Zhang, and Dillabaugh [GZD04] propose a method for land use classification in urban areas using Landsat imagery. A standard pixel-based classification is combined with a segment-based classification—the classification of whole regions obtained by stopping a region merging algorithm at a certain region homogeneity threshold.

Kumar, Loui, and Hebert [KLH03] expose the problem encountered in classification by generative models when, in the image to be classified (the input image), the radiometries (or other features) of the pixels for a certain class have a much smaller support than those in the training image (but are nevertheless contained in it, if the training image is representative). This leads to assigning an incorrect class membership probability to pixels in the input image. They solve the problem by spatially clustering pixels of similar radiometry in the input image, selecting a class for each cluster, constraining the probability model for the class to the cluster’s support, and obtaining the class membership probability for pixels in the cluster using this constrained class model. This is not unlike the problem described in section 5.3 which motivates the use of nested probability models in the classification step for the system described in this thesis, although we believe our solution to be much more flexible—although possibly more computationally expensive. Also, they use the constrained models to obtain the a posteriori probabilities of class membership for the selected class only, but the classification of each cluster is itself made using the unconstrained models, unlike our method, which uses random variable instances, equivalent to the constrained models of that article, also for class selection.

Another hint to the idea of nested models for classification of section 5.3 can be found in [MASB03]. They propose a model whereby each region in a segmentation follows a certain random law parametrized by a region-specific value. The underlying random law class is the same for all regions. Furthermore, regions are not defined a priori, but are the result of the segmentation. The segmentation is performed using Markov Random Field techniques. They suggest using this method for generic unsupervised segmentation. In contrast, our technique is geared towards classification, is supervised, the image partition is defined a priori, and each semantic class may use different random law classes. The class parameters also follow a class-dependent random law, unlike in [MASB03]. Furthermore, we show—through BIC values in model estimation—that we are using this to model radiometric and textural behaviours that actually follow our compound model; in [MASB03] they use it as a generic segmentation tool for images which do not necessarily behave in this way, and the nestedness is used only as a tool to handle heavy noise in images.

2.4.8 Classifier and model selection

If several classification methods are available, it is necessary to select the best one. In some cases, this selection can be made by evaluating the quality of the result of these methods. In other cases, such as when selecting the best model corresponding to a certain training dataset, other techniques have to be used.

Weakliem [Wea99] presents a critique of the Bayes Information Criterion (BIC), a criterion which balances how well a model fits the data with how complex the model is, and which can be used for model selection. In his opinion, the prior beliefs implicit in the calculation of BIC values do not always correspond to reality.

Bernadó Mansilla and Tin [BMT04] give a formal answer to the question of which classifier to use for a given problem. Depending on the shape of the discrimination frontiers, some classifiers are better than others. [DPT04] answers the same question by comparing classifiers not by their correct classification rate, but by whether they give similar classifications—irrespective of whether these are right or wrong. [JTLB04] presents a taxonomy of clustering algorithms and a method for comparing their performances.

Model selection is also discussed in [ZWM97, ZWM98]. They propose a much more complex method for model selection in the case of texture modelling: given a set of image features, they are combined to obtain maximum entropy; given a set of possible models, the one with minimum entropy is selected.

2.4.9 Sample applications and miscellaneous techniques

A great many number of papers have been published on classification techniques and applications. In this section a small selection is given, based on relevance and similarity to this thesis, with the goal not of providing a comprehensive survey, but of giving a taste of the techniques, applications, and performances found in the literature. The study by Wilkinson [Wil05] of articles on classification applications published in between 1989 and 2003 is in this respect very interesting: it shows that the average classification accuracy has stayed at around 80% during all that period. It is extremely surprising that accuracy does not show an improving trend—and the analysis of this fact given in the article is slightly shallow—but it nonetheless sets an interesting benchmark against which to compare this thesis' results. In addition, he shows that accuracy does not depend on the number of features (input channels) used, nor on the pixel resolution (the author hypothesizes that the advantage gained from the higher spatial resolution is offset by the lower spectral resolution), the number of classes, and the area of the experimental site. Much touted neural-network based methods do not show any significant advantage—nor disadvantage—compared to other methods.

Han, Champeaux, and Roujean [HCR04] present a land cover classification product with a coarse classification of non-urban areas of France at 1 km resolution using SPOT4/VEGETATION data. Classification starts with a k -means clustering of the principal component analysis transformation of four spectral bands, and is performed separately pixel by pixel. Evaluation is difficult to interpret, among other reasons because the land classes provided do not match those of the reference database or other similar products, but unfortunately only a 65% accuracy seems to be obtained.

Haapanen *et al.* [HEBF04] present a study on a coarse land cover classification on northern USA. Using 18 low resolution multispectral bands (6 bands obtained from the Landsat 7 ETM+ sensor at three different dates, at 30 m resolution), and a k -nearest neighbours classifier, they obtain accuracies of 88% for a pixel-wise classification into “non-forest”, “forest” and “water”. Of interest is that this figure is considered a good result in the remote sensing community. By using much higher resolution data and more advanced models and classifiers, we obtain higher accuracies, despite the fact that fewer input bands are available.

Viola and Jones [VJ01b, VJ01a] present a system for fast robust object detection (such as faces) based on an efficient image representation and a sequence of filters based on binary masks learnt automatically. Similar methods could be used to detect trees or vine stocks.

Carleer and Wolff [CW04] present a classification of trees into seven different species, with

distinction of two age classes for one class. They use orthorectified Ikonos images, both panchromatic (at 1 m resolution) and multispectral (red, green, blue and near-infrared channels at 4 m resolution), and taken at summer and winter, for the same study areas (making 10 raw image channels). Classification is a standard supervised maximum-likelihood, pixel-wise, using the raw 10 channels, a Normalized Difference Vegetation Index (NDVI) channel, and Principal Component Analysis (PCA) channels, all smoothed by a 3×3 mean filter. Three to twelve pure and homogeneous parcels were chosen for each class for training and evaluation. A 3×3 modal filter was applied to the classification for regularization. An accuracy of 86% is claimed, which is impressive. Not using the mean filter lowers accuracy to 79% but should give more accurate edges. We believe that the fact that both summer and winter images are available is crucial in obtaining such high accuracies for tree species classification.

Benediktsson, Pesaresi, and Arnason [BPA03] show the use of the morphological profile for land cover classification in urban areas. Morphological profiles are calculated for high-resolution panchromatic images (at 5 m and 1 m resolution), dimensionality reduction procedures are used to select some components from the morphological profile, and finally a neural network is used for classification. Accuracies reach 78% when 6 profile components are used (the maximum shown in the article). Classes include "large buildings", "small buildings", "streets", "open areas", "residential lawns" and "shadows", which would be impossible to detect in panchromatic images using traditional remote-sensing techniques only, and for which a morphological approach gives much better results. These ideas are used in a more recent paper by Benediktsson, Palmason and Sveinsson [BPS05] where hyperspectral imagery is used to obtain excellent accuracies ranging from 92.8% to 97.2% with 9 classes.

Chen and Stow [CS03] compare three methods of combining imagery at multiple spatial resolution for land-use and land-cover classification. The first method consists in joining all spectral bands at multiple spatial resolutions together. All layers are brought to the resolution of the finest layer through oversampling, and then a single classifier (a maximum likelihood classifier in their tests) is used for a per-pixel classification. In the second method, a separate per-pixel maximum-likelihood classification is performed on the layers for each resolution (one classification per spatial resolution). For each pixel, the class with the highest posterior probability at all resolutions is selected. In the third method, the coarsest images are used for a first classification; only for pixels with a posterior probability under a certain threshold are the next finer images used for classification, and so on. This third method is shown to give the best results, and the first method the worst.

Schmidt *et al.* [SSK⁺04] present a study on vegetation classification in a coastal area, for 15 vegetation and 4 non-vegetation classes. Previous manual interpretation of the same area had accuracies around 43%. An automatic classification using spectral angle mapping of hyperspectral aerial imagery at 3.5 m resolution (spectral resolution is not given) gives an accuracy of 40%. They describe the post-processing of this per-pixel classification by an expert system which takes into account height, slope, aspect, and terrain position, which brings accuracy up to 66%.

Abou El-Magd and Tanton [AEMT03] present a multi-stage maximum-likelihood classification method and show improvements from 85% to 94% accuracy in crop classification from 6 bands of 30 m resolution Landsat 7 ETM+ images. In each stage, only a subset of the classes are classified. The results from certain stages are used to produce the final land cover map. Murtagh, Barreto, and Marcello [MBM03] use the Bayes Information Criterion (BIC) to select the optimal model among a set—which includes a Markov Random Field prior to express spatial relations—and then perform maximum a posteriori classification using this model, for the case of detecting cloud pixels in satellite images. Topchy *et al.* [TMBJP04] show how to combine several partitions to obtain a better partition, and an adaptive sampling method that concentrates on difficult-to-classify samples. Plaza *et al.* [PMPP04] present an extension

of mathematical morphology operators to multidimensional-valued 2D images, such as hyperspectral imagery, which they use to obtain morphological profiles of these images. They use these profiles in a classification method for hyperspectral images which handles mixed pixels and takes into account simultaneously the spectral and the spatial information. Xiao, Ustin, and McPherson [XUM04] use hyperspectral AVIRIS imagery at 3.5m resolution and 224 channels to classify terrain into several species of trees and also a few non-tree classes, obtaining an average 94% accuracy for tree identification.

2.5 Data fusion, decision

Since we want to have very reliable results, but not necessarily complete coverage, we need a way to evaluate the local reliability of our system's output.

One way to do that—which we have not used in this thesis—, and also to improve the overall performance, is to use multiple data sources for deciding. For example, to decide that a region is a forest we could use simultaneously NDVI values, texture analysis, and expert knowledge based on local slope and on the distance to the nearest town. The combination of these sources can at the same time improve the correctness of our results or, when the sources disagree substantially, make this fact known to the end user so she can then process that area manually. The methods for combining multiple data sources are known as *data fusion* techniques.

Possibility theory or Dempster-Shafer theory [Sha76] provides a framework for dealing with uncertainty and imprecision simultaneously, something that probability theory does not. Fusion of uncertain and imprecise data sources can be done with Dempster's combination rule. Foucher, Boucher, and Bénié [FBB03] use it to combine certain but geometrically imprecise classification from denoised images with uncertain but geometrically precise classification from the raw images. The related techniques of fuzzy sets [Zad65] and fuzzy logic can also be used [Chi03]. Fauvel, Chanussot, and Benediktsson [FCB05] use fuzzy techniques to merge the results of several classifiers, taking into account the different performance of each classifier for each class.

Methods from probability theory usually assume independence of the data sources, which is usually false in this context. There are methods which go around the independence assumption and render probability theory usable for data fusion [CF99, Jou02]. See [PC03] for an application to landslide hazard assessment, and [SFW03] for a detection of urban areas from coarse, mis-registered sources.

Data can also be combined *after* a decision has been taken [AK99]. That is, multiple classification systems ("experts") make decisions independently, and these decisions are combined. The classical *majority vote* fusion method—which selects as global decision the decision that the most experts have given— can be extended in several ways, such as weighting votes according to an optimization algorithm [DSDCM02]. Duin [Dui02] uses an additional classifier to merge the results of the multiple first-level classifiers, and discusses the issue of training this merging classifier. Altınçay and Demirekler [AD02] discuss the problem of normalizing the outputs of the first-level classifiers to make them comparable. Schiele [Sch02] discusses the quantitative effects over performance of using an increasing number of first-level classifiers. Bovino *et al.* [BDI⁺03] describe and compare several classical decision fusion methods (majority vote, Dempster-Shafer, and Behaviour Knowledge Space) and Tax *et al.* [TvBDK00] study whether results should be averaged or multiplied. Benediktsson and Sveinsson [BS03] compare several neural network based methods for combining decisions from several experts, focusing on the extra redundancy given by the fact that multiple experts are deciding on the same object.

2.6 Optimization methods

Many processing steps in the system described here are highly combinatorial and cannot be solved exactly in a reasonable amount of time. There exists a set of techniques, sometimes called *meta-heuristic optimization methods*, which try to give near-optimal solutions to this kind of problems in reasonable time.

Relaxation labelling (RL), also known as probabilistic relaxation [RHZ76, FB81, HZ83, GG84, FY97] is a class of methods for assigning labels to a set of *sites* (pixels, regions, ...) with contextual constraints. It is widely used for regularization and smoothing of raw images or classifications. Several underlying optimization algorithms can be used, such as Iterative Conditional Modes (ICM) [Bes86] or the better performing, but costlier, simulated annealing [KGV83, Ing93, Bus]. Simulated Annealing tries to reach the global optimum by combining a local gradient descent with random perturbations of decreasing variance, in an analogy to the physical phenomena occurring in real-world annealing.

Genetic algorithms [Mic95] are a well known approach inspired from natural evolution. They start with a mapping from tentative solutions to strings of bits, and a number of random initial solutions. Their fitnesses (qualities, or distance to the optimal solution) are evaluated. From the best current solutions new solutions are created by a mechanism known as *cross-over* which mimics sexual reproduction. The worst current solutions are deleted. Solutions may be also randomly modified by *mutations*. This evaluation-crossover-deletion cycle is repeated. At the end, the best solution found so far is given as the result.

Extremal optimization [Boe00, BP01b, BP01a] is a relatively new optimization method based on self-organized criticality, a physical concept related to non-equilibrium processes and phase transitions. It significantly outperforms simulated annealing and genetic algorithms (in execution time and quality of the solution) but can only be applied to problems with *separable cost functions*, that is, problems where the solution is defined by a set of variables, and where it is possible and practical to determine which variable is contributing the most cost to the solution.

2.7 Complete systems

Although we have not found any reference to research covering a similar breadth of topics as this thesis, there are two complete systems worth mentioning.

In his Ph.D. thesis, Flack [Fla95] describes a complete land use classification system with separate segmentation, classification —by maximum likelihood and decision trees— and data fusion steps. For data fusion, he explores the use of Dempster-Shafer's possibility theory and other techniques. He also explores the combination of remotely sensed data with data in a Geographical Information System (which would correspond to the cadastre graph in our case), but is a bit shallow in that respect. The system gives acceptable results, and is able to discriminate among a dozen vegetation classes.

Definiens Imaging GmbH' computer program, eCognition, among many other features proposes automatic and semi-automatic segmentation of an image into agricultural regions, using colour and texture information, with very satisfying results. However, to the best of my knowledge, it does not take cadastre information into account, nor does it give confidence measures on the result.

3 | Image segmentation

The processing of an image starts by segmenting it into regions. With this we accomplish two goals. First, if no cadastre data is available, we obtain a partition of the image into small homogeneous regions which can then be processed by the per-region classifiers of chapter 5. Second, we obtain information about image edges and their saliency —how strong they look—, which we will use in chapter 4 to register the cadastre graph onto the image in order to obtain a partition into larger homogeneous regions.

Figure 3.1 shows as a flowchart the part of the complete process described in this chapter.

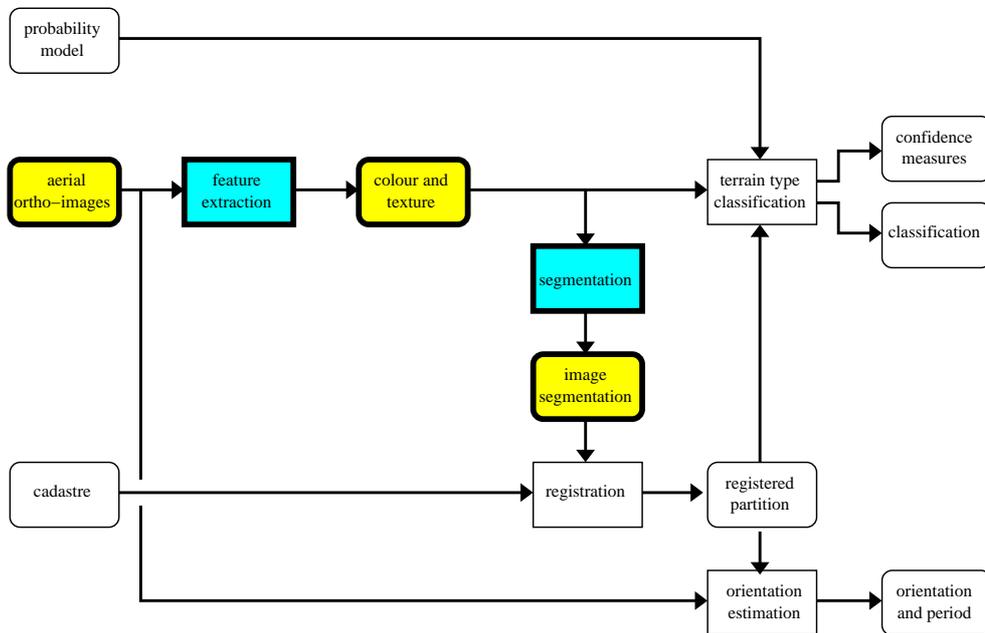


Figure 3.1: Flowchart depicting the processing steps (in blue rectangles) and intermediate results (in yellow rounded boxes) involved in image segmentation, in the context of the complete system.

Image segmentation is usually performed as a preprocessing step for many image under-

standing applications, for example in some land-cover and land-use classification systems. A segmentation algorithm is used with the expectation that it will divide the image into semantically significant regions, or objects, to be recognized by further processing steps. It is however well known that semantically significant regions are found in an image at different scales of analysis. For a high-resolution aerial image, for example, at coarse scales we may find fields, while at finer scales we may find individual trees or plants. Parameters and thresholds in a typical single-scale segmentation algorithm must be tuned to the correct scale of analysis. However, it is often not possible to determine the correct scale of analysis in advance, because different kinds of images require different scales of analysis, and furthermore in many cases significant objects appear at different scales of analysis in the same image.

In an attempt to overcome this problem, in recent years there has been a trend toward multi-scale or hierarchical segmentation algorithms [GLMC03b,SG00]. These analyse the image at several different scales of analysis at the same time. Their output is not a single partition, but a hierarchy of regions, or some other data structure that captures different partitions for different scales of analysis.

In this chapter, we will first review, in section 3.1, general concepts about hierarchical segmentation algorithms, and we will study Guigues' algorithm [GLMC03b,GTSC⁺06] in more detail. We will see that instead of the raw radiometry other kinds of data can be used for segmentation—and, in fact, for classification—perhaps giving better results; section 3.2 describes how transformed colour spaces and texture descriptors can be used instead of the original radiometry channels—the transformed colour spaces and texture descriptors themselves, developed by other authors, are described in appendix A—. That section also studies other aspects of the segmentation algorithm that can be modified or improved to yield better results. Since we then need to determine which of these color spaces, texture descriptors, and other parameters actually performs best, in section 3.3 I present a measure for evaluating the quality of a multiscale segmentation. Finally, section 3.4 describes the experiments that I run to find the best parameters.

3.1 Hierarchical segmentation

Most image segmentation methods take a parameter (usually a threshold for the dissimilarity between adjacent regions) and output a partition of the image. This usually translates into a single scale analysis of the image: small thresholds give segmentations with small regions and much detail, large thresholds give segmentations preserving only the most salient regions.

The problem is that, as it has been said since the early times of image analysis, meaningful structures and objects in a given image appear at different values of the controlling parameter (or in other words, at different scales). Sometimes, a given portion of an image has different meanings at different scales (roof tile, roof, house, . . .). Also, even if in some cases there exists a single, correct scale of analysis, this scale can usually only be obtained from high-level analysis, which happens after, not before, the segmentation phase.

With that in mind, several authors have proposed *multi-scale* segmentation algorithms [JM92, KLM94,SG00].

We decided to use Guigues' algorithm [GLMC03b,GLMC03a] because it makes both the segmentation criterion and the scale parameter explicit, and because of implementation issues. The scale parameter, which for classical segmentation methods is an input parameter, becomes an output in multi-scale methods. It is then up to the post-segmentation stages to either select the most appropriate scale, or to analyse the results as a whole.

3.1.1 Partitions

A *partition* P over a domain \mathcal{D} is a division of \mathcal{D} into separate pieces, or *cells*, p_i . Formally, $P = \{p_i\}_i$, $p_i \in 2^{\mathcal{D}}$ such that the whole domain is covered and the cells are disjoint:

$$\bigcup_i p_i = \mathcal{D}, \text{ and} \quad (3.1)$$

$$\text{for all } i \text{ and } j, i \neq j \Rightarrow p_i \cap p_j = \emptyset. \quad (3.2)$$

We name $\text{part}(\mathcal{D})$ the set of all possible partitions of \mathcal{D} . For an $x \in \mathcal{D}$, let $P(x)$ denote the cell of P which contains x .

We say that a partition $P = \{p_i\}_i$ is *coarser than* another partition $Q = \{q_j\}_j$ (both over the same domain \mathcal{D}), and we write $P \geq Q$, if each cell in Q is contained in a single cell in P ,

$$\text{for all } x \in \mathcal{D}, P(x) \supseteq Q(x). \quad (3.3)$$

Conversely, Q is *finer than* P , $Q \leq P$.

3.1.2 Geometrical graphs

In this section we define a mathematical object, the *geometrical graph*, which combines the topological aspects of a graph with geometrical information. We will use them to describe graphs which are located in an image and, ultimately, on the ground.

Let $G = (V, E)$ be a directed graph, with V the set of vertices and $E \subset V^2$ the set of edges.

Let S be the support space for the geometry part, for example an Euclidean space or \mathbb{Z}^2 with 4-connectivity. Let $f_V : V \rightarrow S$ be a function mapping each vertex to a point in S , and $f_E : E \rightarrow 2^S$ a function mapping each edge to a directed connected curve in S , with the additional condition that for the edge $e = (v_1, v_2)$, the starting point of $f_E(e)$ is $f_V(v_1)$ and the ending point of $f_E(e)$ is $f_V(v_2)$. The trace of a vertex v is $f_V(v)$, the trace of an edge e is $f_E(e)$. For simplicity, let's call trc the function returning the trace of an edge or vertex. The trace of an edge can be viewed as a parametric curve $\text{trc}(e) : [0, 1] \rightarrow S$.

We call the tuple $G' = (V, E, \text{trc}, S)$ a *geometrical graph*, that is, a graph where its elements have a geometrical correspondence. We call trc the *trace function* and S the *geometrical domain* of G' . This notion can be trivially extended to undirected graphs and to weighted or attributed graphs. In the context of geometrical graphs, we also call standard graphs *topographical graphs*.

A geometrical graph is *geometrically planar* (and we may call the standard planarity *topographical planarity*) if no two vertex traces are equal, all intersections of an edge trace and a vertex trace happen at the edge's endpoints, and no two edge traces cut each other:

$$\begin{aligned} &\text{for all } v, v' \in V, e, e' \in E, v \neq v', e \neq e' : \\ &\quad \text{trc } v \neq \text{trc } v', \\ &\quad \text{trc } v \cap \text{trc } e \setminus p(e) = \emptyset, \text{ and} \\ &\quad \text{trc } e \cap \text{trc } e' \setminus (p(e) \cup p(e')) = \emptyset \end{aligned} \quad (3.4)$$

where $p(e) := \text{trc } e(0) \cup \text{trc } e(1)$ are the endpoints of edge e .

3.1.3 Guigues' hierarchical segmentation

We will now describe, borrowing from [GLMC03b], the segmentation algorithm we used.

Let's use as domain \mathcal{D} a finite subset of \mathbb{Z}^2 of size N , the support for our image. To speed up calculations, instead of using a \mathcal{D} whose elements are pixels, we can use a fine partition of the image, such as a watershed segmentation, as the domain for the hierarchical segmentation. Consider a partitioning algorithm applied to an image, A_I , whose result depends on a one-dimensional parameter λ , the *scale parameter*

$$A_I : \mathbb{R} \rightarrow \text{part}(\mathcal{D}), \quad (3.5)$$

and which is *causal*, that is,

$$\lambda_2 \geq \lambda_1 \Rightarrow A_I(\lambda_2) \geq A_I(\lambda_1). \quad (3.6)$$

(recall the definition of \geq for partitions in equation (3.3)).

Consider the set $R_I = \{A_I(\lambda)\}_{\lambda \in \mathbb{R}}$ of the partitions obtained for all values of λ . Notice that, since \mathcal{D} is finite, $\text{part}(\mathcal{D})$ is also finite and so is R_I . Since A_I is causal, there is a total ordering among the elements of R_I , and R_I has size N , $R_I = \{r_1 \leq r_2 \leq r_3 \leq \dots \leq r_N\}$. Note that $r_1 = \bigwedge \text{part}(\mathcal{D})$ is a partition with N cells, the elements of \mathcal{D} as singletons, and $r_N = \bigvee \text{part}(\mathcal{D})$ is a partition with a single cell, \mathcal{D} .

Also note that the transition between r_i and r_{i+1} consists in merging some cells in r_i into a single larger cell in r_{i+1} . If we only consider these merging events, we can then represent the results of A_I for all values of λ simultaneously, as a tree or *hierarchy* H_I . This tree has partition cells as nodes (a single cell $x \subset \mathcal{D}$ which is the result of $A_I(\lambda)$ for many different values of λ is still considered *one* cell). A merging of cells c_1, \dots, c_n into a cell $p = \bigcup_i c_i$ corresponds in the tree with the node p being the parent of the nodes c_i . This tree has the individual elements of \mathcal{D} at the base, and the whole \mathcal{D} at the root.

For any cell $x \in H_I$, we consider $\Lambda(x)$ the set of values of λ for which the cell is part of the optimal partition at that scale,

$$\Lambda(x) = \{\lambda : x \in A_I(\lambda)\}. \quad (3.7)$$

Since A_I is causal, $\Lambda(x)$ is an interval which can be written as

$$\Lambda(x) = [\lambda^+(x), \lambda^-(x)]. \quad (3.8)$$

We call $\lambda^+(x)$ the *scale of appearance* of x and $\lambda^-(x)$ its *scale of disappearance*. We call $\Lambda(x)$ the *interval of persistence* of x . Note that we can reconstruct R_I from (H_I, λ^+) . We call $S_I = (H_I, \lambda^+)$ a *scale-sets* representation of the results of A_I .

Figure 3.2 depicts these concepts graphically.

Finding the partition of \mathcal{D} that maximises the *goodness-of-fit* to the data would just give the trivial partition $\bigwedge \text{part}(\mathcal{D})$. Theories of model inference (variational [MS89], Bayesian [GG84], minimum encoding [Lec89]) indicate the need for a *regularisation* that takes into account the size or complexity of the solution. Usually, the end goal is finding a partition P_λ which minimizes an energy such as

$$E(P, \lambda) = D(P) + \lambda C(P) \quad (3.9)$$

where $D(P)$ measures how well P represents the original image and $C(P)$ measures the complexity of P . Depending on the framework, D is a goodness-of-fit term, a likelihood potential or a deviation encoding cost, and C is a regulariser, a prior potential, or a model encoding cost. The calculation of the optimum $P_\lambda^* = \text{argmin}_{P \in \text{part}(\mathcal{D})} E(P, \lambda)$ is usually intractable.

However, for certain kinds of energies, Guigues proposed a practical algorithm [GLMC03b] for finding P_λ^* for all values of λ , giving the result in scale-sets representation. For this algorithm to work, the energy must be separable, that is, decomposable as

$$E(P, \lambda) = D(P) + \lambda C(P) = \sum_{x \in P} (\bar{D}(x) + \bar{C}(x)), \quad (3.10)$$

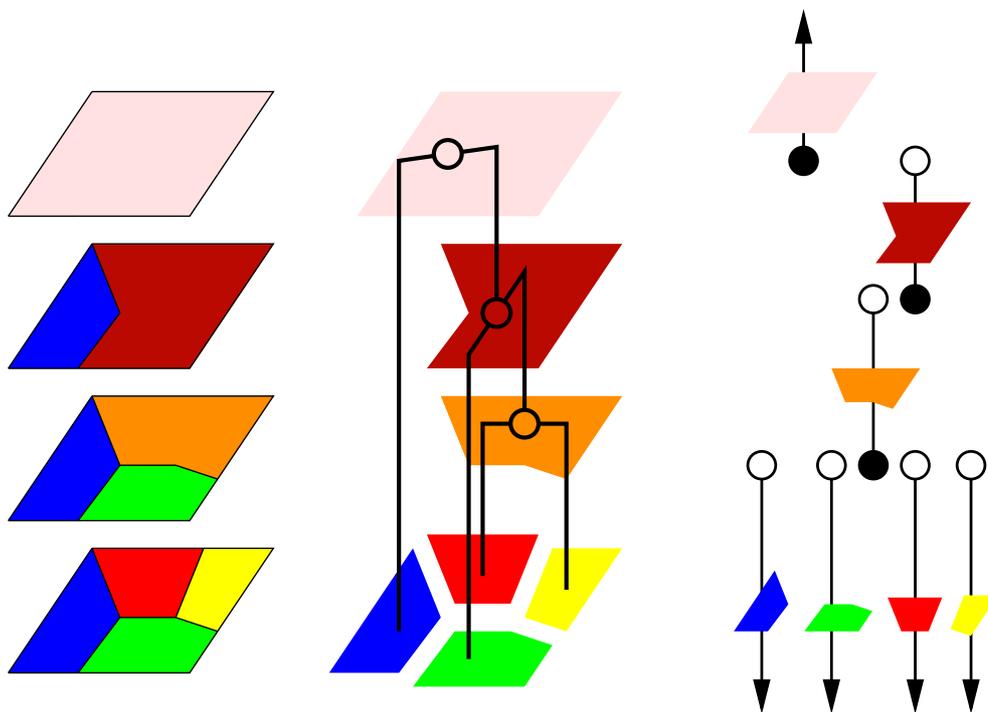


Figure 3.2: Graphical depiction of concepts related to hierarchical segmentation. The diagram on the left shows partitions of an image —not shown— at four different scales λ . The partition at the top has the highest λ and is therefore the coarsest (it is the trivial partition containing only one cell for the whole image); the partition at the bottom is the finest. The diagram in the middle shows the cells of the corresponding hierarchy H , with the links indicating merging events. On the right we have, for each cell in H , its interval of persistence (solid and hollow circles indicate closed and open interval ends, respectively, and arrows indicate intervals to infinity).

and \bar{C} must be decreasing in $2^{\mathcal{D}}$, that is, for all $P, Q \in \text{part}(\mathcal{D})$,

$$P > Q \Rightarrow \bar{C}(P) \leq \bar{C}(Q) \quad (3.11)$$

(i.e. coarser partitions have less regularization energy).

The algorithm works as follows: We start with an initial over-segmentation P_0 . At a given step i , hypothetically merging the cells x and y of the partition P_{i-1} would add the region $x \cup y$ to the hierarchy H . We can compute the scale of appearance $\lambda^+(x \cup y)$ for the energy $\bar{E}(x \cup y) = \bar{D}(x \cup y) + \bar{C}(x \cup y)$. Then we actually merge the two cells whose union would have the smallest scale of appearance. This amounts to choosing the merging which demands the least regularisation.

Many commonly-used energies are separable; for example, the Mumford-Shah functional [MS89], Markov random fields potentials involving a Potts prior [GG84] and many Minimum Description Length criteria [Lec89].

3.1.4 Advantages and drawbacks of a hierarchical segmentation

As stated before, the main advantage of using a hierarchical segmentation is that (at least in the scale-sets framework) there are no image-dependent parameters to set beforehand.

There are however two main problems: region spill-over and the relativity of λ .

Region spill-over

Although the scale-sets representation is presented as a region-oriented approach, and implicitly suggests that any interesting region will appear in the region hierarchy at a certain scale, this is not the case. As shown in figure 3.3 sometimes the merging process “spills over” the desired region before actually including it completely.

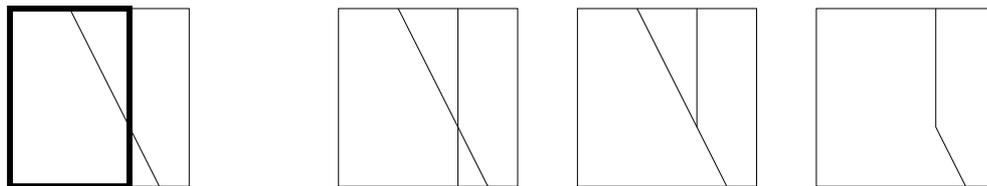


Figure 3.3: Region spill-over. Left: a semantically-significant region (thick line) and other edges in an initial fine segmentation $r_1 = \bigwedge \text{part}(\mathcal{D})$. Three right-most images: successive fusion steps, corresponding to different scales λ ; the desired region of the left image is not a cell in any $A_I(\lambda)$.

We find that, however, the scale of apparition of a cell *is* meaningful: large structures with regular shapes but slightly heterogeneous contents appear higher in the hierarchy, while smaller but more homogeneous structures appear lower in the hierarchy. To solve this problem, we transform the scale-set representation in such a way that the region nesting structure is lost but the scale information is preserved, by *flattening* the hierarchy: We construct a planar, geometrical graph $G = (V, E, t, S)$ whose edges are all the boundaries of the elements of \mathcal{D} (the geometrical domain of this graph need not be \mathcal{D} ; for example, when \mathcal{D} contains pixels or groups of pixels, the geometrical domain contains edges between 2 pixels and points between 4 pixels). We further attribute each edge e with the highest scale of apparition of all the cells in the hierarchy whose boundary contains e .

We obtain, then, a geometrical graph corresponding to the edges of the initial image over-segmentation, with its edges attributed with the highest scale at which this edge exists in the segmentation.

The relativity of λ

As shown by [GLMC03b], the scale parameter has an invariance property, whereby if $S = (H, \lambda^+)$ is the scale-set representation associated with the energy $E(P, \lambda) = D(P) + \lambda C(P)$, then the scale-set representation associated with the energy $E_{\mu, \nu}(P, \lambda) = \mu D(P) + \lambda \nu C(P)$ is $S_{\mu, \nu} = (H, \frac{\mu}{\nu} \lambda^+)$. Different energies can result in very different dynamic behaviour of λ .

Because of that, we decided not to directly use λ^+ , but a transformation of these values, which we call *apparition weight*: We sort the scales of apparition λ^+ of the $|E|$ edges of the geometrical graph in decreasing order (duplicate values are allowed, and are sorted arbitrarily). We then assign to the k -th value of λ^+ the a *weight* of value $e^{-\alpha k}$. For duplicate values of λ^+ , all values are assigned the weight of the first value. We set α such that the last value gets a certain constant weight, independent of $|E|$.

To reduce computational load, we finally delete all edges in G which have an apparition weight under a certain threshold.

3.2 Segmentation energy

As stated in section 3.1.3, the segmentation process is steered by a energy of the form

$$E = D + \lambda C \quad (3.12)$$

where D is a measure of the goodness-of-fit (how well the segmentation fits to the original image, better fits give lower values of D) and C is a measure of the complexity of the segmentation (less complex solutions give lower values of C). The parameter λ balances between a perfect fit to the original data, consisting of one segmentation region for each pixel in the original image, and the simplest segmentation, consisting of a single region containing the whole image.

Guigues' basic segmenter uses Mumford and Shah's piecewise-constant model model [MS89] (also known as *cartoon model*). An image I of support Ω and values in the RGB (red, green and blue) colour space is modelled as a piecewise-constant function. Each constant region is a region in the corresponding image segmentation. If region i has constant value v_i and support ω_i , Mumford and Shah's energy is

$$E = \sum_i \int_{z \in \omega_i} \|I(z) - v_i\|^2 dz + \lambda \sum_i \int_{s \in \partial \omega_i} ds \quad (3.13)$$

where ∂a is the boundary of a . The first term is a goodness-of-fit measure, and the second term a complexity measure (in this case, the total boundary length). By $\int_{s \in A} ds$ we mean the length of the curve A .

While this model is good for the general case, we can use models better suited to our application:

1. Fields, forests and other vegetation regions are not necessarily constant in RGB space. At first sight, it is texture and color hue that are more or less constant. We can evaluate the goodness of fit not as the Euclidean distance in RGB space but in another space which includes transformed colour coordinates and texture parameters.

- For a given scale, the regulariser tends to produce minimal regions (honeycomb-like structures like those in figure 3.4). However, fields and cultivated forests have a very distinct shape, with long straight edges and right angles. Natural forests also tend to have these shapes, since their boundaries are mostly man-made. We can use another regulariser which promotes field-like shapes.

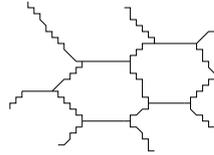


Figure 3.4: Honeycomb structures of minimal complexity.

- We could replace the piecewise-constant model with more complex models, such as a piecewise-smooth one. However vegetation regions are actually piecewise constant in an adequate colour and texture space, and unnecessarily complicating the model would yield worse results.

In RGB colour space, vegetation regions have non-constant radiometry in several situations: shadows, and slopes of non-constant gradient give variations in the colour intensity. Different crop growth states within a single field can give arbitrary variations; we choose to segment zones of different crop growth state as different vegetation regions, and eventually to merge them back together at a later processing stage.

- We could take into account the fact that close vegetation regions tend to have similar shapes and crops, by using, for example, Markov models. I have not researched into this and, in particular, I have not studied whether this modification would affect energy separability.

In this section we will study some of this alternate models for input data and region shape.

We will start by studying transformed colour spaces and texture descriptors, which, when used as input instead of the raw radiometry channels, may give better segmentations. These alternate input channels may also give better results for the classifications of chapter 5. The proposed colour spaces and texture descriptors, developed by other researchers and which I adapted in minor ways to this specific application, are themselves described in appendix A.

3.2.1 Colour spaces

We would like to find distance functions for colour values such that there is a large distance between different colours and a small distance between similar colours. The problem is that what we mean by “similar” and “different” is application-specific. We want vegetation regions which “look” different to be well separated using this distance.

Of course, the question is inherently ill-posed, and we will avoid giving a clear answer to it. Instead we will evaluate many different distance functions against a ground truth. The problem of defining colour similarity is then translated to the problem of creating a ground truth. However, the basic idea remains that we want distances that reproduce subjective human responses (distances with *perceptual relevance*), and which reflect the real land use and crop growth state while being more or less independent of external factors such as lighting or weather.

We must note that, in general, it is equivalent to define a new distance d' for colours in the red-green-blue space $C = [0, 1]^3$ (or equivalently for our purposes, $\{0, \dots, 255\}^3$), and to define a transformation F from C to another space C' (not necessarily a linear transformation), and use the Euclidean distance there. For $x, y \in C$,

$$d'(x, y) = d_{\text{Eucl}}(Fx, Fy). \quad (3.14)$$

The space resulting from the transformation of C by F is called a *transformed colour space*.

In this section we will describe the colour spaces we investigated for use in our system, as well as some additional transformations. Note that we do not need to use all the components of a single transformed colour space: we may choose to use some components from one colour space and some from another one. See [CJSW01] for a review of colour spaces in the context of image segmentation.

We have not found any reference to colour spaces taking the near-infrared channel into account. This was to be expected: colour space transformation is mostly aimed at obtaining perceptually relevant distances; since we humans do not perceive near-infrared light, perceptual relevance is meaningless for that frequency band. We will however consider a transformation involving the near-infrared and red channels, the *normalized difference vegetation index*.

Figures 3.5 and 3.6 show the red, green, and blue channels, composited into a single image, of a sample fragment of the Toulouse data set, and of the St. Léger data set, respectively. In the remainder of these chapter, several images will be shown depicting different colour and texture channels. The original imagery in the Toulouse data set contains red, green, blue, and near-infrared channels at 20 cm ground pixel size. Texture parameters are calculated at 50 cm resolution, and the images shown in this chapter are downsampled to 2 m. For the St. Léger data set, only the red, green, and blue channels are available; their original resolution is 80 cm, but they have been resampled to 50 cm—and the original data is not available. Texture parameters are calculated at 50 cm resolution, and images shown here are downsampled to 2 m.

Apart from the basic red-green-blue space, the transformed colour spaces that seemed interesting for this application were the hue-saturation-intensity space, the Karhunen-Loève colour space, the CIE XYZ, Lab, and Luv spaces, and log-opponent chromaticity spaces. We also explored a simple technique for haze removal, and briefly investigated one-dimensional colour constancy. Since the main object of interest for this application is vegetation, we also studied several vegetation indices. These colour spaces were developed by other researchers, and I only had to do some minor modifications to some of them to adapt them to the framework of this thesis. They are described in detail in appendix A.

3.2.2 Texture parameters

We would also like to find other kinds of parameters that mostly have constant values across a vegetation region and present discontinuities at its edges. Vegetation is heavily textured (for example, we can distinguish between a forest, a field, and a vineyard from texture alone, with gray-scale images), so texture parameters can be useful.

We have concentrated on two types of parameters: Parameters that measure the orientation of a texture and parameters that measure the complexity of a texture. Note that these same parameters may allow us to roughly classify a given vegetation area. In a way, by using them at the segmentation stage (before the actual classification) we let semantic information be used for low-level processing, without having to take a hard classification decision at such early stage.

Most texture description operators that we have found in the literature operate on whole images



Figure 3.5: Red, green, and blue composite image for the G003002 section of the Toulouse data set.



Figure 3.6: Red, green, and blue composite image for the G002003 section of the St. Léger data set.

(that is, they give a single, perhaps multi-dimensional, descriptor for the whole image). We need operators that describe each region in a partitioned image, or that give local descriptions of each pixel and its neighbourhood. Therefore, we have had to adapt some of these operators.

Texture is typically a local property, but not a property of single pixels. Colour, on the other hand, is a pixel property. The regions over which we calculate texture should already be homogeneous in texture. How can we use texture parameters as input to the segmentation algorithm, if we do not know yet over what regions we should calculate texture parameters? There are two solutions.

The first solution assumes that the actual shape of the region we use for calculating texture is mostly irrelevant. For each pixel in the image, we calculate texture parameters over a square neighbourhood centred on it. We can then use the results as a pixel-level input to the segmentation algorithm. At the boundaries of vegetation regions, the analysis windows will actually cover two different textures, so there will not be sharp discontinuities in texture, as we would like to have, but smooth transitions. With this solution we are assuming that this smoothing is not a problem. We call these “pixel based” and are designated in section 3.4 with names starting with “pix-”.

In the second solution, we first run the segmentation algorithm—which operates by iteratively merging regions—using only colour and pixel-based texture features as input, until a certain condition on region size is met. We then calculate the texture parameters for each of these regions, and restart the merging at the point we had left it, using also texture this time. In this solution, we assume that when we first stop the algorithm, the merged regions will be big enough so that texture calculation in them will be possible, and at the same time small enough that segmentation on colour alone is enough to produce only homogeneous regions. We call these “region based” and are designated in section 3.4 with names starting with “reg-”. Not all textural features can be calculated in this way.

We do not actually calculate texture parameters for each pixel, because of computational cost. For one pixel in every, for example, four pixels, we calculate texture parameters using *all* pixels in a square neighbourhood around it. So the texture *features* are those obtained from the full-resolution images and describe textures found at that resolution, but the *resolution* of the texture parameter is reduced and the positional accuracy of texture boundaries is lower.

These operators work on single-channel images only. We use the image intensity (from the HSI colour space) as input. We have not found it necessary to use texture operators capable of handling colour textures.

Two-phase segmentation

Guigues’ segmentation algorithm is implemented as a region-merge segmentation, that is, after an initialisation step in which the image is partitioned into a large number of small regions—for example, using a fine watershed segmentation, or even putting each pixel into a region of its own—at each iteration two neighbouring regions are selected and merged into a larger one. In a classical “one phase” set-up, the segmentation ends when the whole image has been merged into a single region. Guigues’ algorithm differs from other region-merge algorithms in that the obtained hierarchy—and more precisely the scale parameter corresponding to each fusion—is defined by the partition minimizing a certain energy involving data-fit terms and partition complexity terms, but not involving any explicit notion of region fusion. That is, in Guigues’ algorithm, step-by-step region fusion simply turns out to be a possible method for globally minimizing an energy.

This single-phase segmentation has two drawbacks. First, computation time is high. Second, and more important, texture parameters must be computed using an arbitrarily-shaped

neighbourhood. We use “two-phase segmentation” to overcome these problems.

In two-phase segmentation the segmentation procedure is altered. Two size thresholds are defined, t_{low} and t_{high} , and the ordering of region fusions is modified so that certain fusions are not executed. In particular, a candidate region pair, with regions of areas a_1 and a_2 must satisfy

$$\max(a_1, a_2) < t_{\text{high}} \text{ and } \min(a_1, a_2) < t_{\text{low}}. \quad (3.15)$$

in order to be considered for fusion. This segmentation ends when the only remaining region pairs are ineligible for fusion.

At this point, the image has not yet been merged into a single large region, but is still partitioned into smaller regions. The edges of this partition will be the final segmentation edges for the two-phase segmentation. Region-based texture parameters are then computed, using these regions as neighbourhoods. Because these regions were obtained using image data, they may more accurately follow meaningful object boundaries.

Afterwards, a new multiscale segmentation is executed, but in this case the final image partition of the previous segmentation is used as initial image partition, instead of a fine watershed segmentation or a partition into pixels, as would usually be done. Thanks to this, there are much fewer initial image regions, and the segmentation is done much more quickly. This second segmentation is run until the whole image is merged into a single large region, and it is the scales of analysis obtained in this second segmentation that are used to attribute the final segmentation edges.

After reviewing the literature, the texture descriptors that seemed interesting for this application were Gabor filters, Local Binary Patterns, the dot-pattern detector, and scale-orientation histograms. We also studied indicators of fractal dimension, of structure complexity, and a Fourier-based orientation estimator, and we investigated the use of the intensity average as a “texture descriptor”. As with transformed colour channels, these texture descriptors were developed by other researchers. However they were mostly developed for image database retrieval applications, so the modifications I had to do to adapt them to the framework of this thesis are more significant, although uninteresting from a research point of view. The descriptors, and these modifications, are described in detail in appendix A.

3.2.3 Shape regularisers

In Guigues’ hierarchical segmentation set-up, and in other energetical formulations such as Mumford and Shah’s segmentation energy, a complexity term $C(P)$ balances the data-fit term $D(P)$ and acts as a regulariser, preventing the trivial solution of partitioning an image into its individual pixels from being taken.

Guigues proposes several complexity terms in his thesis work [Gui04]: a constant cost K , the polygonalisation length L_σ , the number of segments T_σ in the polygonalisation, a polar encoding length P_σ , a Gaussian polar encoding length P_σ^N , and a concavity energy C_σ . Polar encoding lengths are the number of bits needed to transmit a polygonalisation encoded in polar form, using Rissanen’s universal encoding for the segment lengths. For P_σ , angles are supposed equiprobable, and for P_σ^N they are assumed to follow a Gaussian distribution.

Let’s polygonalise a connected component of a region’s boundary to precision σ . This polygonalisation contains p segments, and can be described, up to a translation of the connected component, as a sequence of p vectors $v_i, i \in \{1, \dots, p\}$ (one per segment), and \hat{p} angles θ_i , where θ_i is the angle between v_i and v_{i+1} ($\hat{p} = p - 1$ for open boundaries, or $\hat{p} = p$ for closed boundaries). Let $\Sigma_\theta^2 := \frac{1}{\hat{p}} \sum_{i=1}^{\hat{p}} \theta_i^2$. Let q be an angle quantization step, which Guigues arbitrarily takes

as $q = \pi/256$. Guigues' geometric costs are then defined as follows:

$$K = 1, \quad (3.16)$$

$$L_\sigma = \sum_{i=1}^p \|v_i\|, \quad (3.17)$$

$$T_\sigma = p, \quad (3.18)$$

$$P_\sigma = 5p + \sum_{i=1}^p \log \|v_i\|^2, \quad (3.19)$$

$$P_\sigma^N = 2p + \sum_{i=1}^p \log \|v_i\| + \frac{p}{2} \log(2\pi e \Sigma_\theta^2 / q^2), \quad (3.20)$$

$$C_\sigma = \sum_{i=1}^{\hat{p}} |\theta_i|. \quad (3.21)$$

3.2.4 Gradient-based data fit

Guigues also suggests adding a gradient-based data-fit term to $C(P)$. In particular, he proposes modifying the geometry complexity costs so that it is less expensive to have segmentation boundaries along regions of the image with a high gradient module. This is expected to make segments follow high-contrast lines. For a geometry energy function E defined for a region with boundary ∂R , he derives a gradient-following geometric energy function E^f as

$$E^f = E \cdot \frac{1}{|\partial R|} \sum_{x \in \partial R} f(\|\nabla I(x)\|) \quad (3.22)$$

where $\nabla I(x)$ is the image gradient at point $x \in \mathbb{Z}^2$. He proposes the following f s:

$$f(G) = \frac{1}{k + G^a}, \quad (3.23)$$

$$f(G) = e^{-kG^a}. \quad (3.24)$$

3.2.5 Fit to local cadastre orientation

So far, the geometry complexity energies described try to implement general ideas about what "simple" shapes are. For example, the length energy L_σ favours compact shapes, while the concavity energy C_σ favours convex shapes. We would like to modify these shape regularization terms to take into account the shapes found in the particular domain of this thesis work, namely, land plots: plots are usually of rectangular shape, are convex, and have straight edges. Furthermore, we can adapt these geometric energies to take into account not only the general shape of plots of land, but also the specific shape expected in a given analysis area.

To do that, we use the observation that land-use regions have similar shapes *and orientations* to those of cadastre regions. For each pixel we obtain the local cadastre orientation, as follows:

Let $C = \{c_i\}_i$ be the set of cadastre edges, each edge being $c_i = (\mathbf{a}_i, \mathbf{v}_i)$ a segment defined by an endpoint \mathbf{a}_i and a vector $\mathbf{v}_i = (\rho_i \cos \theta_i, \rho_i \sin \theta_i)$. Its trace is

$$\text{trc } c_i = \{\mathbf{a}_i + s\mathbf{v}_i : 0 \leq s \leq 1\}. \quad (3.25)$$

Let's quantize angles (modulo π) into q histogram bins, each bin h_b corresponding to the angle range

$$h_b = [b\pi/q, (b+1)\pi/q) \cup [\pi + b\pi/q, \pi + (b+1)\pi/q). \quad (3.26)$$

We write $\theta_1 \equiv \theta_2$ if the angles θ_1 and θ_2 belong to the same histogram bin. We define a *local cadastre orientation*, $\Theta : \mathbb{R}^2 \times \mathbb{R} \rightarrow \mathbb{R}$ as

$$\Theta(z, \theta) = \sum_{i: \theta_i \equiv \theta} (\text{trc } c_i * G)(z), \quad (3.27)$$

and a *local directional cost*, $W : \mathbb{R}^2 \times \mathbb{R} \rightarrow \mathbb{R}$ as

$$W(z, \theta) = \alpha(z) \frac{1}{m + \Theta(z, \theta)}, \quad (3.28)$$

with $\alpha : \mathbb{R}^2 \rightarrow \mathbb{R}$ having values such that $\int_0^{2\pi} W(z, \theta) d\theta = 2\pi$, with m a small non-zero value to ensure that no orientation has an infinite cost, and where G is a very wide smoothing kernel, for example a Gaussian kernel with a large σ . In these equations, $*$ is the convolution operator.

The function W defines an anisotropy in the \mathbb{R}^2 (or \mathbb{Z}^2) image space. That is, some directions are more costly than others. We can then modify the basic geometric energies given by Guigues to take this anisotropy into account.

Lengths

For a segment with endpoints a_i and $a_i + v_i$, which has an isotropic, Euclidean length of $\|v_i\|$, we define an anisotropic length, using the W anisotropy function, as

$$\|v_i\|_W := \|v_i\| \int_0^1 W(a_i + sv_i, \arg v_i) ds. \quad (3.29)$$

For slowly-varying functions W such as the ones we use, we can approximate equation (3.29) by

$$\|v_i\|_W \simeq \|v_i\| W(a_i + v_i/2, \arg v_i). \quad (3.30)$$

Angles

The P_σ^N and C_σ energies use the concept of angle. This concept can also be extended to anisotropic spaces.

Angle measures can be defined through the scalar product. The angle between two vectors a and b measures α radians, where

$$\alpha = \arccos \frac{a \cdot b}{\|a\| \|b\|}. \quad (3.31)$$

For our purposes, there is another definition of angle measures which is more useful. It is tied to that of length through the following: The angle defined by two vectors v_1 and v_2 (which we write $\text{ang}(v_1, v_2)$), expressed in radians, is the length of the curve that lies on the unit circle $\{z \in \mathbb{R}^2 : \|z\| = 1\}$, starts at $v_1/\|v_1\|$ and ends at $v_2/\|v_2\|$ (rotating counter-clockwise from the start to the end points), that is, trivially,

$$\text{ang}(v_1, v_2) = \int_{\arg v_1}^{\arg v_2} d\theta. \quad (3.32)$$

Note that a similar definition is involved in the measure of solid angles.

By transposing the anisotropic length used in equation (3.29), we might define the anisotropic angle spanned by vectors v_1 and v_2 with center in z , $\text{ang}_{\int W, z}(v_1, v_2)$ as

$$\text{ang}_{\int W, z}(v_1, v_2) = \int_{\arg v_1}^{\arg v_2} W(z, \theta) d\theta. \quad (3.33)$$

See figure 3.7. Note that a full angle also has an anisotropic measure of 2π radians, and that the measure of two juxtaposed angles is the sum of their measures.

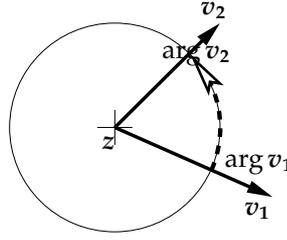


Figure 3.7: Diagram describing the variables involved in the length-based definition of angular measures of 3.33.

However nice these two geometrical properties of $\text{ang}_{\int W, z}$ may be, this definition has a fundamental problem that prevents its use as an anisotropic replacement for angle measures in segmentation geometric energies. Whereas the anisotropic length measure is, as we want, smaller for segments which are oriented like cadastre edges in the vicinity than that of other segments, the anisotropic angle measure is *not* smaller for angles which are oriented like angles in the neighbouring cadastre compared to other angles:

Assume that, locally, the cadastre graph consists of a grid of more-or-less vertical and horizontal lines, so that $\Theta(z, \theta) = \cos(2\theta)^4$. Then, $W(z, \theta) = \frac{\alpha}{m + \cos(2\theta)^4}$. Let's take $m = 0.1$.

With such a cadastre graph, horizontal and vertical lines are less costly than diagonal lines, as expected:

$$\left\| \begin{pmatrix} 1 \\ 0 \end{pmatrix} \right\|_W = \int_0^1 W(\cdot, 0) ds = \frac{\alpha}{m + 1} = \frac{10}{11}\alpha \approx 0.9\alpha, \quad (3.34)$$

$$\left\| \begin{pmatrix} 0 \\ 1 \end{pmatrix} \right\|_W = \int_0^1 W(\cdot, \pi/2) ds = \frac{\alpha}{m + 1} = \frac{10}{11}\alpha \approx 0.9\alpha, \quad (3.35)$$

$$\left\| \begin{pmatrix} 1/\sqrt{2} \\ 1/\sqrt{2} \end{pmatrix} \right\|_W = \int_0^1 W(\cdot, \pi/4) ds = \frac{\alpha}{m} = 10\alpha. \quad (3.36)$$

However, the angle spanned by $(1, 0)^T$ and $(0, 1)^T$ has the same anisotropic measure as the angle spanned by $(-1, -1)^T$ and $(1, -1)^T$,

$$\text{ang}_{\int W, z}((1, 0)^T, (0, 1)^T) = \int_0^{\pi/2} W(z, \theta) d\theta = \pi/2, \quad (3.37)$$

$$\text{ang}_{\int W, z}((-1, -1)^T, (1, -1)^T) = \int_{-\pi/4}^{\pi/4} W(z, \theta) d\theta = \pi/2, \quad (3.38)$$

despite the fact that the former is oriented like angles in the cadastre and the latter is not. See figure 3.8.

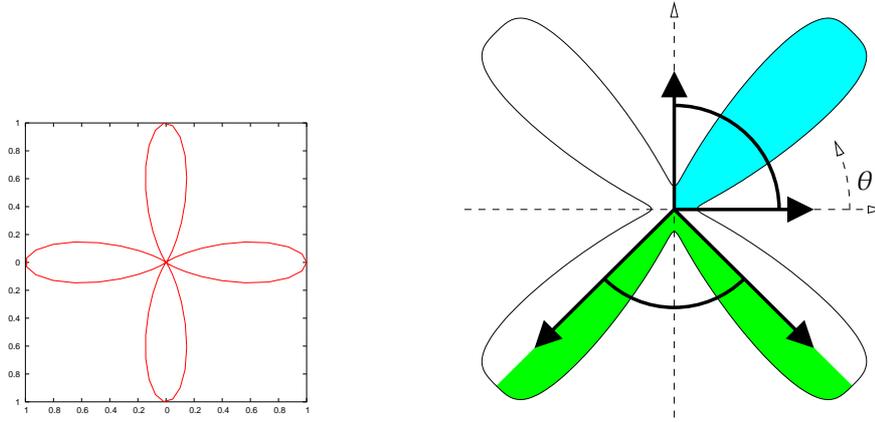


Figure 3.8: Left: $\Theta(z, \theta)$. Right: diagram showing that $\text{ang}_{\int W, z}$ gives the same anisotropic angular measure for an angle oriented like those found in the cadastre—in blue—and to one unlike cadastre angles—in green.

In the geometric energies proposed by Guigues, angles are used in such a way that larger angle measures imply higher geometric energies or costs. I therefore propose the following definition for an anisotropic angle measure:

The anisotropic angle defined by vectors v_1 and v_2 with center in z , $\text{ang}_{W, z}(v_1, v_2)$ can be defined as

$$\text{ang}_{W, z}(v_1, v_2) = (W(z, \arg v_1) + W(z, \arg v_2)) \cdot \text{ang}(v_1, v_2). \quad (3.39)$$

This measure does not have the interesting properties of $\text{ang}_{\int W, z}$, but it is much more appropriate to our needs, since angles oriented like those in the cadastre have smaller measures than angles which are not. For example,

$$\text{ang}_{W, z}((1, 0)^T, (0, 1)^T) = 2 \frac{10}{11} \alpha \frac{\pi}{2} \simeq 0.9 \alpha \pi, \quad (3.40)$$

$$\text{ang}_{W, z}((1, -1)^T, (1, 1)^T) = 2 \cdot 10 \alpha \frac{\pi}{2} \simeq 10 \alpha \pi, \quad (3.41)$$

$$\text{ang}_{W, z}((1, 0)^T, (1, 1)^T) = (10/11 + 10) \alpha \frac{\pi}{4} \simeq 2.7 \alpha \pi, \quad (3.42)$$

$$(3.43)$$

Anisotropic geometric energies

With these correspondences for lengths and angles, we can obtain anisotropic versions of the geometry complexity energies proposed by Guigues.

To the values p , v_i , \hat{p} , and θ_i that define a polygonalisation of a connected component of a region's boundary (as shown in section 3.2.3), we add z_i , the mid-point of the i -th segment, \hat{z}_i , the position of the i -th vertex, and θ_i^W , the anisotropic angle between v_i and v_{i+1} . If \hat{z}_0 is the

starting point of the polygonalisation,

$$\hat{z}_i = \hat{z}_0 + \sum_{j=1}^i v_j \quad (3.44)$$

$$z_i = \hat{z}_{i-1} + \frac{1}{2}v_i \quad (3.45)$$

$$\theta_i^W = \min\{\text{ang}_{W, \hat{z}_i}(-v_i, v_{(i+1) \bmod p}), \text{ang}_{W, \hat{z}_i}(v_{(i+1) \bmod p}, -v_i)\}. \quad (3.46)$$

See figure 3.9 for a diagram showing these points, vectors, and angles.

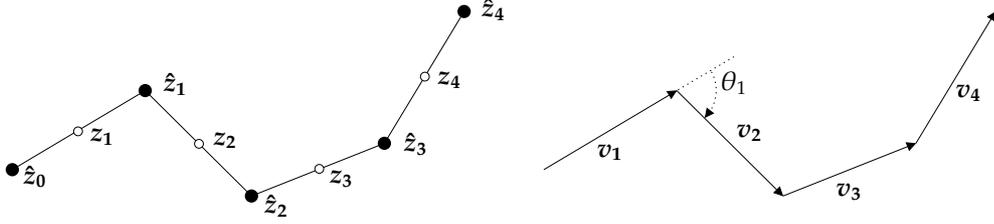


Figure 3.9: Diagram showing the points, vectors, and angles of a polygonalised segmentation boundary.

The anisotropic polygonalisation length is, using the approximation from equation (3.30),

$$L_\sigma^W = \sum_{i=1}^p \|v_i\|_W = \sum_{i=1}^p \|v_i\| W(z_i, \arg v_i). \quad (3.47)$$

The “number of segments” cost can be similarly adapted, if we take that $T_\sigma = p = \sum_{i=1}^p 1$ and by analogy with L_σ ,

$$T_\sigma^W = \sum_{i=1}^p W(z_i, \arg v_i). \quad (3.48)$$

The polar encoding length becomes

$$P_\sigma^W = 5p + \sum_{i=1}^p \log \|v_i\|_W^2 = 5p + \sum_{i=1}^p 2 \log(\|v_i\| W(z_i, \arg v_i)). \quad (3.49)$$

The Gaussian polar encoding length becomes

$$P_\sigma^{W, \mathcal{N}} = 2p + \sum_{i=1}^p \log(\|v_i\| W(z_i, \arg v_i)) + \frac{p}{2} \log(2\pi e \Sigma_{\theta, W}^2 / q^2), \quad (3.50)$$

$$\Sigma_{\theta, W}^2 = \frac{1}{\hat{p}} \sum_{i=1}^{\hat{p}} (\theta_i^W)^2. \quad (3.51)$$

Finally, the anisotropic concavity energy is

$$C_\sigma^W = \sum_{i=1}^{\hat{p}} |\theta_i^W|. \quad (3.52)$$

The gradient-based data-fit term can be incorporated to anisotropic geometrical energies in the same way as Guigues did for isotropic energies (equation 3.22), that is, for an anisotropic geometric energy function E^W defined for a region with boundary ∂R , we can derive a gradient-following geometric energy function $E^{f,W}$ as

$$E^{f,W} = E^W \cdot \frac{1}{|\partial R|} \sum_{x \in \partial R} f(\|\nabla I(x)\|). \quad (3.53)$$

3.3 Evaluating the quality of a multiscale segmentation

As with classical, single-scale, segmentation algorithms, the need arises to evaluate the quality of a multi-scale segmentation against a reference, in order to compare different algorithms, and to select for an algorithm the parameters which are optimal for a given application. Most current segmentation evaluation methods [SPA03, LY94] handle only single-scale segmentations, that is, partitions of an image. They usually work by finding correspondences between points in the reference and points in the edges of the regions given by the segmentation. However, because multi-scale algorithms can deliver arbitrarily fine segmentations —at the finer end of the scale range— the concepts of “correspondence between reference points and segmentation edge points” and of “distance between segmentation edge and reference edge” cannot be easily transposed to that case —in effect, at the right scale, the segmentation is so fine that all reference points are arbitrarily close to a segmentation edge.

In this section I present a method for evaluating the quality of a multi-scale or hierarchical image segmentation against a reference. The reference segmentation is given as a set of edges of two kinds, *compulsory* and *optional*. The method computes two measures, a *false detection* (or commission) measure, and a *missed detection* (or omission) measure, by considering that a falsely detected segmentation edge is a bigger error if it appears at a coarse scale of analysis; conversely, the missed detection measure takes into account the fact that a reference edge can be completely missed —if there is no corresponding segmentation edge— but also “nearly missed” if only fine-scale corresponding segmentation edges are found. This assumes that the most visually salient edges are found at the coarsest scales, which is the case. These two measures can be minimized jointly, or an aggregate such as their average can be calculated. I have found that they can be used to compare two different multi-scale segmentations of the same image.

This method was presented, in summarized form, in [TS05a]. In the following sections, I will use these quality measures to determine the most appropriate input data to use for the segmentation, and the optimal settings for the several segmentation parameters.

3.3.1 Quality measures

The procedure to compute the segmentation quality measures operates by searching for pixels belonging to segmentation edges and for reference pixels. The first step is therefore to convert the inputs into a suitable discrete (pixel based) form.

The segmentation reference is given as two sets of segments, one for the compulsory edges and one for the optional edges. A rasterisation algorithm —such as Bresenham’s line drawing method— is applied to convert these sets of edges into two sets of pixels. Note that information about what edge each pixel belongs to is lost. Let $D \subset \mathbb{Z}^2$ be the usually rectangular image domain. Pixel coordinates are elements of D . Let $R_c \subset D$ be the set of pixels given by the compulsory reference edges. Let $R_o \subset D$ be the set of pixels given by the optional reference edges, and $R = R_c \cup R_o$.

The hierarchical or multi-scale segmentation must also be converted into a flat, discrete, representation. Guigues [GLMC03b] and others suggest that multi-scale segmentations should be *causal*, i.e., segmentation edges found at coarser scales must have corresponding edges at finer scales. When corresponding finer edges have exactly the same position as the coarser edges, as is the case in Guigues' algorithm, the flattening technique described in [TS04a] can be used: Each edge $e \in E$ can be found in segmentations at a given coarsest scale $\lambda_E^+(e)$ and at all finer scales. The edge e is rasterized and converted to a set of pixels, $p(e) \subset D$. Let $S = \cup_{e \in E} p(e)$. For each pixel s in S , its coarsest scale is computed as

$$\lambda_S^+(s) = \max\{\lambda_E^+(e) : s \in p(e), e \in E\}. \quad (3.54)$$

However, the behaviour of $\lambda_E^+(e)$ is highly dependent on the specific segmentation technique and parameter set used. To avoid this, we use not $\lambda_S^+(s)$ but this transformation: All values of λ_S^+ are sorted in decreasing order, keeping repeated values. Then each is replaced by the exponential of its rank—the rank being 0 for the largest value of λ_S^+ and $|S| - 1$ for the smallest; repeated values are given the rank of their first occurrence—as

$$w^+(s) = e^{-\alpha \cdot \text{rank}(\lambda_S^+(s))}, \quad (3.55)$$

where α is a user-defined parameter.

In other causal algorithms where the finer corresponding edge can have a different position from that of the coarser edge, each edge e can be seen to keep its position between scales $\lambda_E^-(e)$ and $\lambda_E^+(e)$. We can then correspondingly define λ_S^- , w^- , and, in the following discussion, use $w = w^+ - w^-$ instead of w^+ .

Computing the missed detection and false detection measures involves searching in small circular neighbourhoods of pixels, of fixed radius ϵ . The neighbourhood radius ϵ is used as a distance threshold when determining if a segmentation pixel and a reference pixel could correspond and, therefore, ϵ must be set taking into account positional accuracy in the reference and acceptable positional errors in the segmentation edges, and must keep the same value for all tests in a comparison of segmentations.

Let $B_{x,\epsilon}$ be the ball in D centred on x with radius ϵ ,

$$B_{x,\epsilon} = \{\mathbf{y} \in D : \|\mathbf{y} - \mathbf{x}\| \leq \epsilon\}. \quad (3.56)$$

Given the reference sets R and R_c , the segmentation set S , the weight map w^+ , and the neighbourhood radius ϵ , the missed detection penalty for a single pixel $x \in R_c$ is defined as

$$p_m(\mathbf{x}) = \begin{cases} 1 - \max_{\mathbf{y} \in B_{x,\epsilon} \cap S} w^+(\mathbf{y}) & \text{if } B_{x,\epsilon} \cap S \neq \emptyset, \\ 1 & \text{if } B_{x,\epsilon} \cap S = \emptyset, \end{cases} \quad (3.57)$$

and the false detection penalty for a single pixel $x \in S$ as

$$p_f(\mathbf{x}) = \begin{cases} 0 & \text{if } B_{x,\epsilon} \cap R \neq \emptyset, \\ w^+(\mathbf{x}) & \text{if } B_{x,\epsilon} \cap R = \emptyset. \end{cases} \quad (3.58)$$

The missed detection quality measure for the whole segmentation is

$$M = \frac{\sum_{\mathbf{x} \in R_c} p_m(\mathbf{x})}{\text{card } R_c}, \quad (3.59)$$

and the false detection quality measure for the whole segmentation is

$$F = \frac{\sum_{x \in S} p_f(x)}{\sum_{x \in S} w^+(x)}. \quad (3.60)$$

The missed detection—or omission—penalty is computed for all pixels corresponding to compulsory ground truth segmentation edges. The penalty is maximum, as in single-scale segmentations, when no corresponding segmentation edge pixel is found. However, a penalty is also given when a corresponding pixel is found, and is lower the higher the pixel scale is. Therefore, if the only segmentation pixel corresponding to a ground truth edge is found at very fine scales of analysis, it will be counted as an “almost-missed detection”. On the contrary, if the corresponding pixel is very salient and appears at coarse scales of analysis, the penalty will be minimal. Conversely, the false detection or commission penalty is computed for all pixels corresponding to detected segmentation edges, that is, edges present in the segmentation algorithm’s output. No penalty is given if a corresponding ground truth edge is found. If none is found, the pixel’s scale is used for the penalty. Therefore, very salient detected edges that do not correspond to the ground truth have a high penalty, while incorrectly detected edges at fine scales of analysis are better tolerated.

This provides a two-dimensional measure of the quality of a segmentation compared to a ground truth, $(M, F) \in [0, 1] \times [0, 1]$. The closer M and F are to zero, the higher the quality. These measures can be used to compare different segmentation algorithms or parameter sets, but should not be taken to be absolute quality measures—that is, a value of $F = 0.5$ does not mean that half the detected edges are false detections. In order to find the optimal parameter set these two measures can be minimized jointly, using the partial order

$$(M_1, F_1) \leq (M_2, F_2) \iff M_1 \leq M_2 \cap F_1 \leq F_2, \quad (3.61)$$

or an appropriate aggregate such as their average can be calculated and that scalar value minimized.

3.4 Choosing the best segmentation parameters

The results of the segmentation phase will be inputs to the following processing stages, which have to decide where vegetation regions are, and classify these regions. The segmentation phase need not itself produce a partition of the image into these vegetation regions. Instead, it should simply either find relevant edges in the image, to be used in the registration phase, or find small homogeneous regions that will be classified if cadastre data is not available. Therefore, we can afford to tolerate a certain amount of error, be it bad localisation, missing some limits, and of course, a substantial amount of overdetection.

That said, we would still like to have the best possible results at that point. However, it is not trivial to choose one “best” segmentation, for two reasons.

First, we do not know how the quality of the segmentation relates to the quality of the final result—in terms of classification accuracy. Unfortunately, it would be computationally too expensive to optimize for the final classification result directly. We will have to content ourselves with an optimization of the quality measures of section 3.3.

Second, the parameter space for the segmentation algorithm is too large. To sidestep this problem, I will assume that some parameters are mutually independent and optimize them separately, and I will use Stepwise Forward Selection for some of these parameters.

3.4.1 Pareto optimality

Because the segmentation quality measures are two-dimensional, it is not trivial to compare parameter sets to find the optimum.

Let's say that the quality measure of the parameter set p is $Q(p) = (\mu_p, \phi_p)$ where μ_p is the missed-detection measure and ϕ_p is the false-detection measure. If we have two parameter sets a and b with $Q(a) = (\mu_a, \phi_a)$ and $Q(b) = (\mu_b, \phi_b)$, should we say that a is *better than* b if $\mu_a\phi_a < \mu_b\phi_b$? Or should we say it if $\mu_a + \phi_a < \mu_b + \phi_b$? Or something else?

Instead we will use *Pareto optimality* [Par06], [Gaw70, Allan Schick, p. 32]. Given two n -dimensional vectors $\mathbf{a} = (a_0, a_1, \dots, a_{n-1})$ and $\mathbf{b} = (b_0, b_1, \dots, b_{n-1})$ we say that, in the sense of Pareto,

$$\begin{aligned} \mathbf{a} < \mathbf{b} &\iff \forall i \in \{0..n-1\} . a_i < b_i, \\ \mathbf{a} > \mathbf{b} &\iff \forall i \in \{0..n-1\} . a_i > b_i, \\ \mathbf{a} = \mathbf{b} &\iff \forall i \in \{0..n-1\} . a_i = b_i, \\ \mathbf{a} \geq \mathbf{b} &\iff \text{neither } \mathbf{a} < \mathbf{b}, \mathbf{a} > \mathbf{b}, \text{ nor } \mathbf{a} = \mathbf{b}; \end{aligned} \tag{3.62}$$

in the latter case, $\mathbf{a} \geq \mathbf{b}$, we say that \mathbf{a} and \mathbf{b} are not comparable.

Given a set of n -dimensional vectors S , the minima set (in the sense of Pareto) of S , $\min S$, is the set

$$\min S = \{a \in S : \forall b \in S . \neg(a > b)\} \tag{3.63}$$

where \neg is the logical "not". We can define the maxima set, $\max S$, in a similar manner.

3.4.2 Methodology

I assume that a higher segmentation quality, as defined above, translates, approximately, into a higher overall system performance. That is, if the remaining processing stages stay constant, using segmentation parameters giving higher segmentation performances will yield higher performances for the whole system.

However, that being just an educated guess, we will not only select the Pareto-optimal parameter sets, but instead we will in some cases keep a few more, which have qualities close to the qualities of these optima, and test the whole system will all of them.

There is an additional problem: the parameter space, taking into account colour transformations, texture parameters, shape regularisers, and so on, is too big to be explored systematically. I will assume that some parameters are mutually independent and optimize them separately.

The parameters to be optimized are the following:

1. Which radiometry, derived radiometry, and pixel-based texture channels should be used as segmentation input as the first segmentation phase—that which may be used to obtain regions for region-based texture features. In addition, some features, such as *hue*, have an angular nature; how this is integrated with other features must also be taken into account.
2. Whether the segmentation should be divided into two phases (sections 3.2.2 and 3.4.4), and in that case which stop condition for the first segmentation phase to use.
3. Which radiometry, derived radiometry, pixel-based texture channels, and region-based texture channels should be used as segmentation input as the second segmentation phase, and how angular features should be handled.

4. Which data-fit energy term D to use. However, Guigues' implementation of his hierarchical segmentation algorithm only provides one complexity term. This parameter will therefore not be studied.
5. Which geometry energy term C to use.
6. Whether the "gradient-based data fit" of equations (3.22) should be used, and the values of its factor (k) and exponent (a) parameters.
7. Whether the "fit to local cadastre orientation" of equations (3.27) to (3.53) should be used, and the value of the m parameter in equation (3.28).

Because only for the Saint Léger test site we have a reliable boundary ground truth, the tests described in this chapter have been run only for this site and not for the Toulouse site.

Figure 3.10 shows a flowchart of steps involved in testing and evaluating a segmentation configuration.

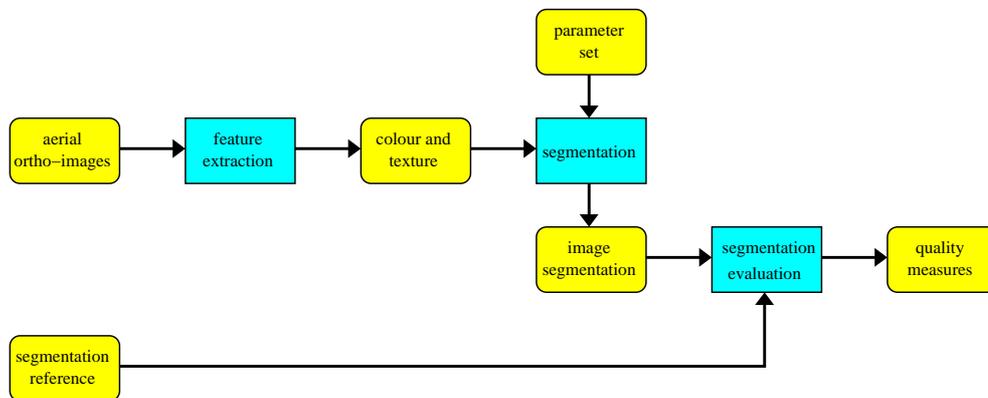


Figure 3.10: Flowchart depicting the processing steps (in blue rectangles) and intermediate results (in yellow rounded boxes) for the evaluation of a parameter set for the segmentation subsystem.

3.4.3 Choice of input channels

The first parameter that we optimize is the set of channels to be used in the first and second phase of the segmentation, whether the segmentation should actually have two phases (sections 3.2.2 and 3.4.4), and the treatment of polar channels.

Because of the enormous size of the parameter space, we cannot do any kind of exhaustive search. Furthermore, the parameter being non-numerical, typical gradient descent optimization techniques are not applicable. It is unclear whether heuristic methods such as genetic algorithms would be of any use.

For all these reasons, I have chosen to optimize the choice of channels using Stepwise Forward Selection (SFS), as in [DK82, MK97]. This technique proceeds as follows: in the first step, each channel is tested separately; that giving the best results are retained for the second step. In the second step, we test each combination of the channel retained in the first step and another channel. The two channels giving the best result (one of which will be the channel obtained in the first step) is retained for the next step. Similarly, in the third step these two channels are combined with each of the remaining ones, and so on. The procedure may be stopped at a

fixed step, or when the best result from one step is worse than that of the previous step. In the end, the channels giving the best overall result are selected.

I have modified the standard SFS to suit two needs: First, the distinction between the first and second segmentation phases must be taken into account. Second, polar channels must be properly handled. In addition, if several results at one step look promising, we may keep all of them instead of only the best.

A *polar* channel, one that has an angular nature (with, for the purpose of this explanation, values in $[0, 2\pi)$), cannot be directly used as an input to the segmentation because the segmenter would consider values close to 0 as very different from values close to 2π , when they are in fact close. Let ϕ be a channel with angle values. Let f be a channel with non-angular values (a *linear* channel), which we will also assume to have values in $[0, 1]$. Then, instead of ϕ , we can use as segmentation inputs one of the following:

1. The channels $\cos(\phi)$ and $\sin(\phi)$,
2. the channel $\cos(\phi)$,
3. the channel $\sin(\phi)$,
4. the channels $f \cos(\phi)$ and $f \sin(\phi)$,
5. the channel $f \cos(\phi)$,
6. the channel $f \sin(\phi)$,
7. the channels $(1 - f) \cos(\phi)$ and $(1 - f) \sin(\phi)$,
8. the channel $(1 - f) \cos(\phi)$, or
9. the channel $(1 - f) \sin(\phi)$.

In particular, the polar channels are: *hue*, *pix-gabor-a*, *pix-soh-th*, *pix-soh-m*, *pix-aurelie-d1*, *pix-aurelie-d2*, *sathue*, *reg-soh-th*, *reg-soh-m*, *reg-aurelie-d1*, and *reg-aurelie-d2*.

The central part of the SFS algorithm is deciding which tests to perform given the results of the previous step. Let the best result of one step be described by a tuple $(R_1, R_2, U_1, U_2, C_p, C_r)$, where:

1. R_1 is the set of segmentation inputs for the first segmentation phase, after transformation of polar channels (that is, it would include inputs such as $\cos(\text{hue})$ instead of *hue*); this will be empty in the first step,
2. R_2 is the set of segmentation inputs for the second segmentation phase; this will be empty if the segmentation is to have only one phase,
3. U_1 is the set of channels, before transformation of polar channels, used in R_1 ,
4. U_2 is the set of channels, before transformation of polar channels, used in R_2 ,
5. C_p is the set of pixel-based colour and texture channels not in $U_1 \cup U_2$,
6. C_r is the set of region-based colour and texture channels not in U_2 ,

for example, $R_1 = \{\text{pix-ent-e}, \cos(\text{hue})\}$, $U_1 = \{\text{pix-ent-e}, \text{hue}\}$, $R_2 = \emptyset$, $U_2 = \emptyset$. Let's note by $T \leftarrow m$ the addition of the test described by the tuple m to the list of tests T to be performed in the next step. Then the list of tests to be performed in the next step can be obtained, in the adapted SFS algorithm, as described in figure 3.11.

For each unused channel $c \in C_p \cup C_r$:

1. If $R_2 = \emptyset$ and $c \in C_p$:

If c is a linear channel: $T \leftarrow (R_1 \cup \{c\}, R_2, U_1 \cup \{c\}, U_2, C_p \setminus \{c\}, C_r)$.

Else, is c is a polar channel:

1. $T \leftarrow (R_1 \cup \{\cos c, \sin c\}, R_2, U_1 \cup \{c\}, U_2, C_p \setminus \{c\}, C_r)$.

2. $T \leftarrow (R_1 \cup \{\cos c\}, R_2, U_1 \cup \{c\}, U_2, C_p \setminus \{c\}, C_r)$.

3. $T \leftarrow (R_1 \cup \{\sin c\}, R_2, U_1 \cup \{c\}, U_2, C_p \setminus \{c\}, C_r)$.

4. For each *linear* channel d in U_1 :

For each S in $\{d \cos c, d \sin c, \{d \cos c\}, \{d \sin c\}, \{(1-d) \cos c, (1-d) \sin c\}, \{(1-d) \cos c\}, \{(1-d) \sin c\}\}$,

$T \leftarrow (R_1 \cup S, R_2, U_1 \cup \{c\}, U_2, C_p \setminus \{c\}, C_r)$.

2. If $R_1 \neq \emptyset$:

If c is a linear channel: $T \leftarrow (R_1, R_2 \cup \{c\}, U_1, U_2 \cup \{c\}, C_p \setminus \{c\}, C_r \setminus \{c\})$.

Else, is c is a polar channel:

1. $T \leftarrow (R_1, R_2 \cup \{\cos c, \sin c\}, U_1, U_2 \cup \{c\}, C_p \setminus \{c\}, C_r \setminus \{c\})$.

2. $T \leftarrow (R_1, R_2 \cup \{\cos c\}, U_1, U_2 \cup \{c\}, C_p \setminus \{c\}, C_r \setminus \{c\})$.

3. $T \leftarrow (R_1, R_2 \cup \{\sin c\}, U_1, U_2 \cup \{c\}, C_p \setminus \{c\}, C_r \setminus \{c\})$.

4. For each *linear* channel d in U_2 :

For each S in $\{d \cos c, d \sin c, \{d \cos c\}, \{d \sin c\}, \{(1-d) \cos c, (1-d) \sin c\}, \{(1-d) \cos c\}, \{(1-d) \sin c\}\}$,

$T \leftarrow (R_1, R_2 \cup S, U_1, U_2 \cup \{c\}, C_p \setminus \{c\}, C_r \setminus \{c\})$.

Figure 3.11: Main step of the modified SFS algorithm.

The algorithm is first run with a state tuple $(\emptyset, \emptyset, \emptyset, \emptyset, C_p, C_r)$ with C_p the set of pixel-based channels and C_r the set of region-based channels.

We have not run a Principal Component Analysis on colour and texture input, because—for texture—calculating all of them for each image would be computationally prohibitive. Therefore, we are not interested in selecting the few linear combinations of colour and texture inputs which are most discriminant, but in selecting a few untransformed, discriminant channels.

These tests were performed using the following default values for the remaining segmentation parameters:

1. Low region size threshold for two-phase segmentation: $t_{\text{low}} = 200 \text{ m}^2$,
2. high region size threshold for two-phase segmentation: $t_{\text{high}} = 400 \text{ m}^2$,
3. complexity term for region boundaries: T_σ of equation (3.18),
4. gradient-based data fit of equation (3.22): not used,
5. fit to local cadastre orientation of equations (3.27) to (3.53): not used.

The choice of ϵ affects the actual quality measures, but has no effect on the quality itself: higher values of ϵ give lower values to the quality measures (and thus show higher quality), but this apparent improvement is, of course, caused by a higher tolerance of the quality measurement. In the tables and plots shown in this chapter, we have chosen $\epsilon = 4$.

The results of the first step of the SFS algorithm are given in table B.1 (in the appendices) and figures 3.12 and 3.13. Each line in table B.1 describes a segmentation test; we give the channel sets R_1 and R_2 that were used in the test, and the quality measures M and F of the resulting segmentation. This has been run for a hand-made reference on the St. Léger test site. The Pareto-optimal parameter sets are marked in green. Parameter sets on white background in a green frame are Pareto-optimal but will not be retained for the next step because they are too far away from the origin. In addition, the close-to-optimal parameter sets marked in yellow will also be retained for the next step. In this first step, because only one input channel was used, two-phase segmentation was not performed: only a single segmentation phase was done.

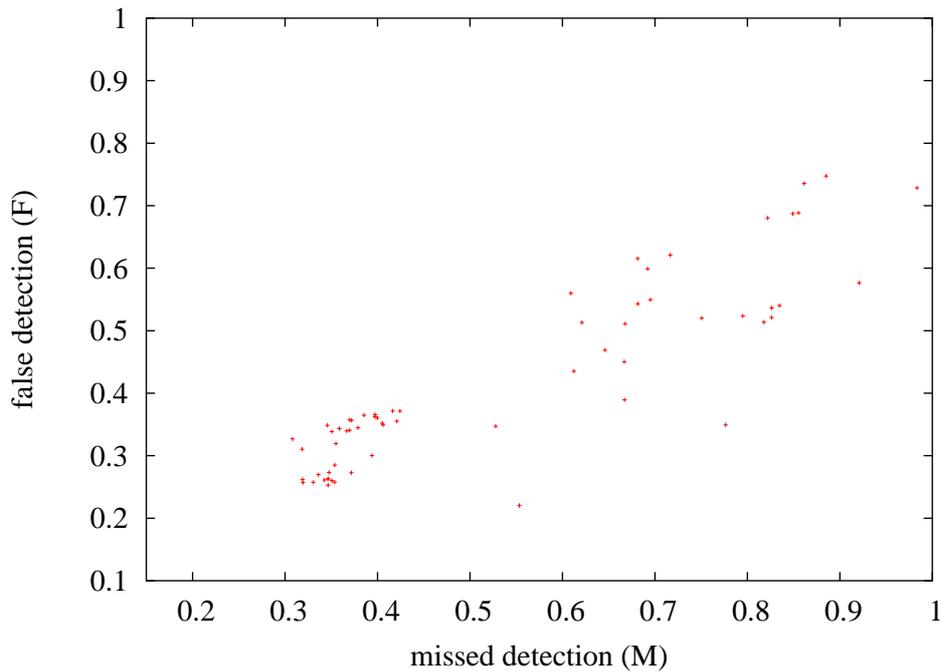


Figure 3.12: Scatter plot of the segmentation quality measures for the first step of the Stepwise Forward Selection for the selection of the best parameter sets.

The parameter sets marked in green and yellow background in table B.1 are used as “best results” to obtain the list of tests to perform in step 2. In this second step, the following little-promising channels (those giving the worst results in the first step) were excluded from the available channel set $C_p \cup C_r$: *kl3*, *ciea*, *log-rg*, *log-gs*, *pix-gabor-m*, *pix-gabor-f*, *pix-gabor-a*, *pix-fractal*, *pix-lbp-riu2m*, *pix-lbp-riu2f*, *pix-ent-e*, *pix-ent-c*, *pix-ent-lg-d*, *pix-ent-lg-h*, *pix-dot-i*, *pix-dot-s*, *pix-soh-th*, *pix-soh-g*, *pix-soh-m*, *pix-soh-v*, *pix-aurelie-d1*, *pix-aurelie-p1*, *pix-aurelie-d2*, *pix-aurelie-p2*, *sathue*, *tgdoi*, and *wi*.

The results of the second step of the SFS algorithm are given in table B.2 (in the appendices) and figures 3.14 and 3.15. For table B.2 we indicate, for each channel set R_1 and R_2 used, the two segmentation quality measures; color coding is the same as in table B.1. In this second step two input channels were used; some tests use both of them in a single segmentation phase (then $R_2 = \emptyset$), and some use one in the first and one in the second segmentation phase.

The parameter sets marked in green background in table B.2 are used as “best results” to obtain the list of tests to perform in step 3. In this third step, the following little-promising channels (those giving the worst results in the second step) were excluded from the available channel set $C_p \cup C_r$, in addition to those already excluded in the second step: *kl1*, *reg-lbp-riu2m*, *reg-lbp-*

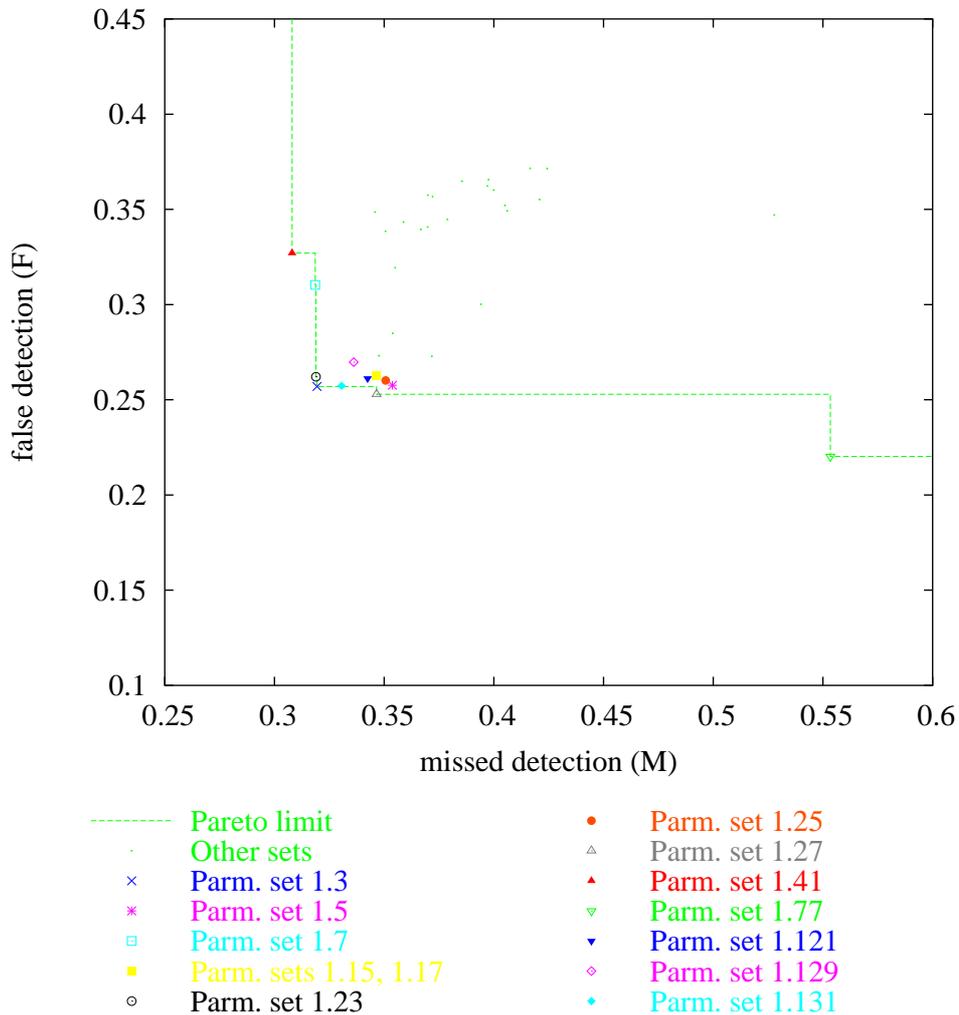


Figure 3.13: Close-up of the scatter plot of the segmentation quality measures for the first step of the SFS, showing the Pareto-optimal parameter sets, some additional parameter sets that are retained for the next step, and the Pareto limit (all points in the upper-right side of this limit are worse than some Pareto-optimal parameter). Parameter set numbers correspond to those in table B.1. The remaining parameter sets are also shown as smaller dots.

riu2f, reg-ent-e, reg-value-i, reg-soh-th, reg-soh-m, reg-soh-v, reg-fractal, reg-aurelie-d1, reg-aurelie-p1, reg-aurelie-d2, and reg-aurelie-p2.

The results of the third step of the SFS algorithm are given in table B.3 (in the appendices) and figures 3.16 and 3.17. For table B.3 we indicate, for each channel set R_1 and R_2 used, the two segmentation quality measures; colour coding is the same as in table B.1. Notice that the Pareto limit contains, in this case, some points from the second SFS step as well as the third SFS step. These optima coming from the second SFS step should not be used as “best results” to initialize the fourth SFS step, since these would simply repeat some of the tests of the third step.

The SFS algorithm does not select the combination of *red*, *green*, and *blue* channels for testing.

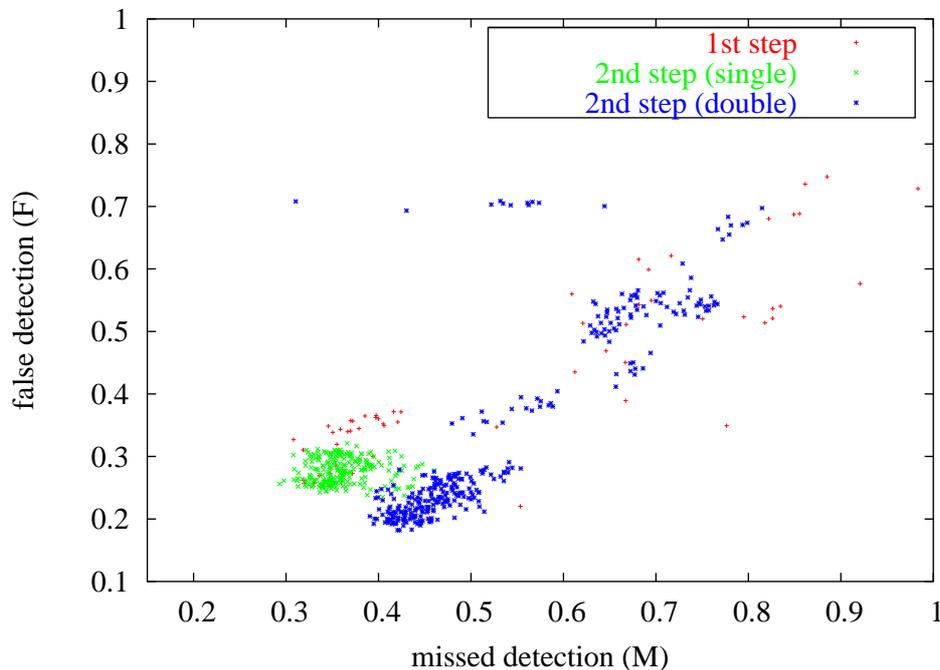


Figure 3.14: Scatter plot of the segmentation quality measures for the second step of the Stepwise Forward Selection for the selection of the best parameter sets. Measures using a single segmentation phase are shown in green, and those using two segmentation phases are shown in blue. The results of the previous SFS step are also shown, in red.

We have however manually tested them for completeness. The results for $R_1 = \{red, green, blue\}$ and $R_2 = \emptyset$ are $M = 0.3722$ and $F = 0.2569$.

The parameter sets marked in green and yellow background in table B.3 are used as “best results” to obtain the list of tests to perform in step 4. No additional channels were excluded.

The results of the fourth step of the SFS algorithm are given in table B.4 (in the appendices) and figures 3.18 and 3.19. For table B.4 we indicate, for each channel set R_1 and R_2 used, the two segmentation quality measures; colour coding is the same as in table B.1. Notice that the Pareto limit contains, in this case, some points from the previous SFS steps.

Since the fourth SFS step shows little improvement compared to the third step, we shall stop iterating the SFS algorithm at this point. Taking into account the results at all steps, the best choice of channels is given in table 3.1 and figure 3.20.

It is interesting to compare these results to those of Vansteenkiste *et al.* [VSGP04]. In that article, colour channels and texture features are tested for a terrain segmentation in high resolution satellite images. The authors conclude that texture does not add any significant advantage. In this section we have obtained similar results, after a much more thorough evaluation: we have tested many more texture features and colour spaces, and have performed tests over a larger area. We found that, indeed, texture does not seem to be useful to obtain good segmentations: no texture channel is present in any of the optimal parameter sets. However, we do see the interest of using derived colour channels, or polar transformations. Of the 12 optimal parameter sets, only one, number 2.435, does not contain any (non-trivially) derived or transformed colour channel. This is shown graphically in figure 3.21, which compares test sets which use at least one texture feature with those that use none for the fourth step of SFS

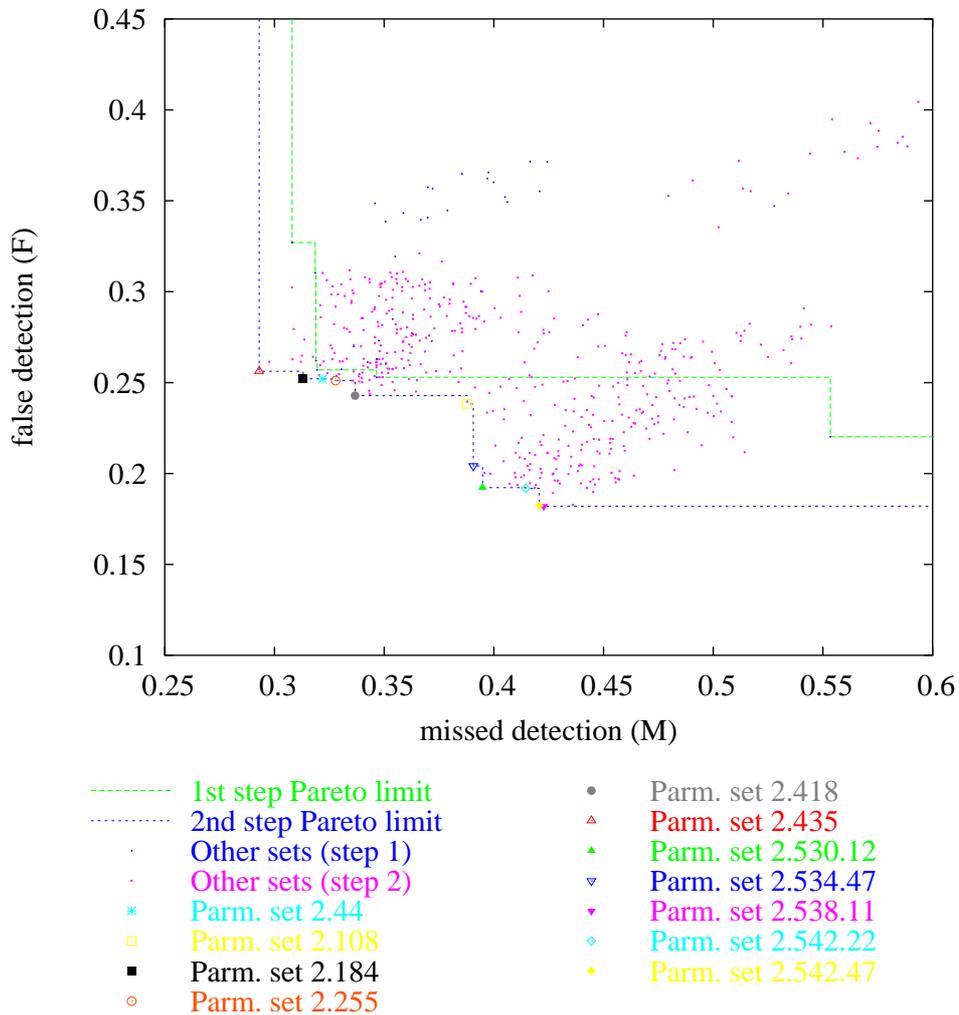


Figure 3.15: Close-up of the scatter plot of the segmentation quality measures for the second step of the SFS, showing the Pareto-optimal parameter sets, some additional parameter sets that are retained for the next step, and the Pareto limit (all points in the upper-right side of this limit are worse than some Pareto-optimal parameter). Parameter set numbers correspond to those in table B.2. The Pareto limit for the previous SFS step is also shown. The remaining parameter sets, for the first and second SFS steps, are also shown as smaller dots.

(other steps have qualitatively similar results); note that sets without texture are clearly better performing.

It may seem counter-intuitive at first to notice that textural features offer no improvement over radiometry for segmentation purposes, although the fact that Vansteenkiste *et al.* obtain similar results is reassuring. It may be argued that, since textural features are inherently defined on neighbourhoods and not individual pixels, their positional accuracy is less than for radiometric data. Although, as we will see, texture does play a role in terrain classification, it may be that radiometry is enough for segmentation—as long as it is not necessary to determine the terrain types being separated—and that texture’s inferior positional accuracy makes it less useful for

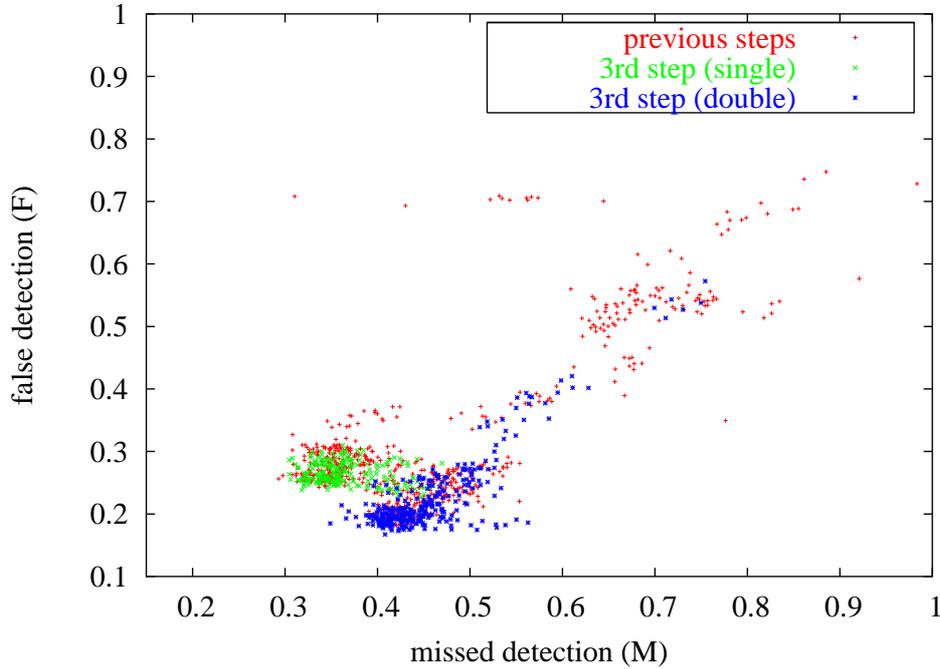


Figure 3.16: Scatter plot of the segmentation quality measures for the third step of the Stepwise Forward Selection for the selection of the best parameter sets. Measures using a single segmentation phase are shown in green, and those using two segmentation phases are shown in blue. The results of the previous SFS steps are also shown, in red.

Number	Input channels (R_1) / Input channels (R_2)	Missed det. (M)	False det. (F)
2.435	<i>green-z, blue</i> / \emptyset	0.2931	0.2562
3.425	<i>green-z, blue, sat</i> / \emptyset	0.3061	0.2551
4.154	<i>green-z, blue, log-rs, (1 - log-rs) · sin(hue)</i> / \emptyset	0.3069	0.2531
4.177	<i>green-z, blue, log-rs, chr-rg</i> / \emptyset	0.3123	0.2520
4.146	<i>green-z, blue, log-rs, log-rs · cos(hue)</i> / \emptyset	0.3170	0.2409
3.447	<i>green-z, blue, log-rs</i> / \emptyset	0.3182	0.2383
3.597.26	<i>satint</i> / <i>ciez, log-bg</i>	0.3490	0.1848
3.595.56	<i>ciex</i> / <i>blue-z, chr-rg</i>	0.3798	0.1808
4.464.131	<i>ciex</i> / <i>blue-z, green-z, kl2</i>	0.3818	0.1745
4.464.72	<i>ciex</i> / <i>satint, cie, green-z</i>	0.3884	0.1722
4.464.36	<i>ciex</i> / <i>satint, kl2, green-z</i>	0.3902	0.1689
4.464.26	<i>ciex</i> / <i>satint, kl2, cie</i>	0.4042	0.1615

Table 3.1: Best segmentation parameter sets after four iterations of SFS.

segmentation.

For the following experiments we will select, as optimal parameter set, set number 3.597.26, with *satint* as input channel for the first segmentation phase, and *ciez* and *log-bg* as input channels for the second segmentation phase. For this parameter set, $M = 0.3490$ and $F = 0.1848$.

For illustration, figures 3.22, 3.23, and 3.24 show some image segmentations obtained in the previous tests. The source image is shown in figure 3.22. Figure 3.23 shows the corresponding

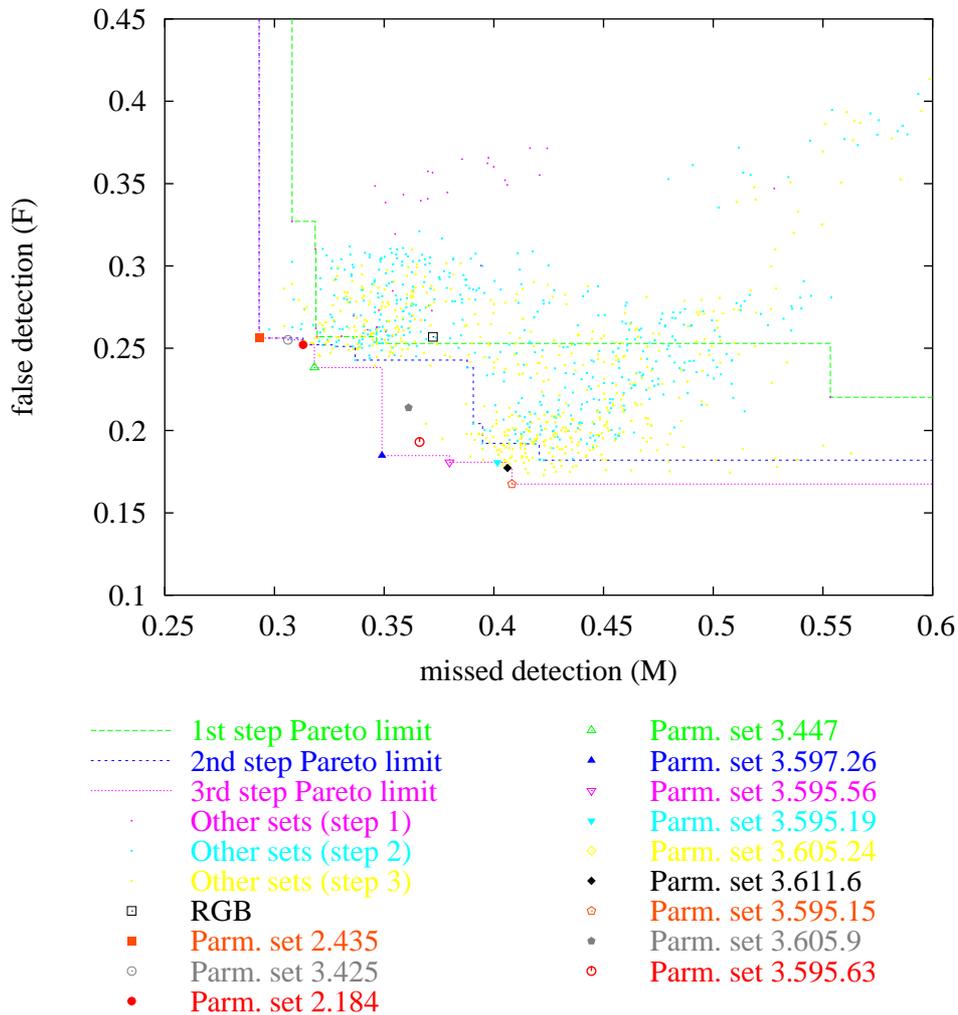


Figure 3.17: Close-up of the scatter plot of the segmentation quality measures for the third step of the SFS, showing the Pareto-optimal parameter sets, some additional parameter sets that are retained for the next step, and the Pareto limit (all points in the upper-right side of this limit are worse than some Pareto-optimal parameter). Parameter set numbers correspond to those in table B.3 and B.2. The Pareto limit for the previous SFS steps is also shown. The remaining parameter sets also shown as smaller dots.

segmentation with set number 3.597.26, the selected optimal set, with quality measures $M = 0.3490$ and $F = 0.1848$. Darker edges appear at coarser analysis scales in the multiscale segmentation, and are therefore supposed to correspond—if the parameter set is well chosen—to more salient edges in the source image. Figure 3.24 shows the segmentation corresponding to an average badly-performing parameter set, set number 2.544.35, with quality measures $M = 0.7721$ and $F = 0.6471$. Note that, whereas darker edges in figure 3.23 do correspond to more salient edges in the source image, edge darkness in figure 3.24 is less correlated with edge saliency—hence the worse quality measures.

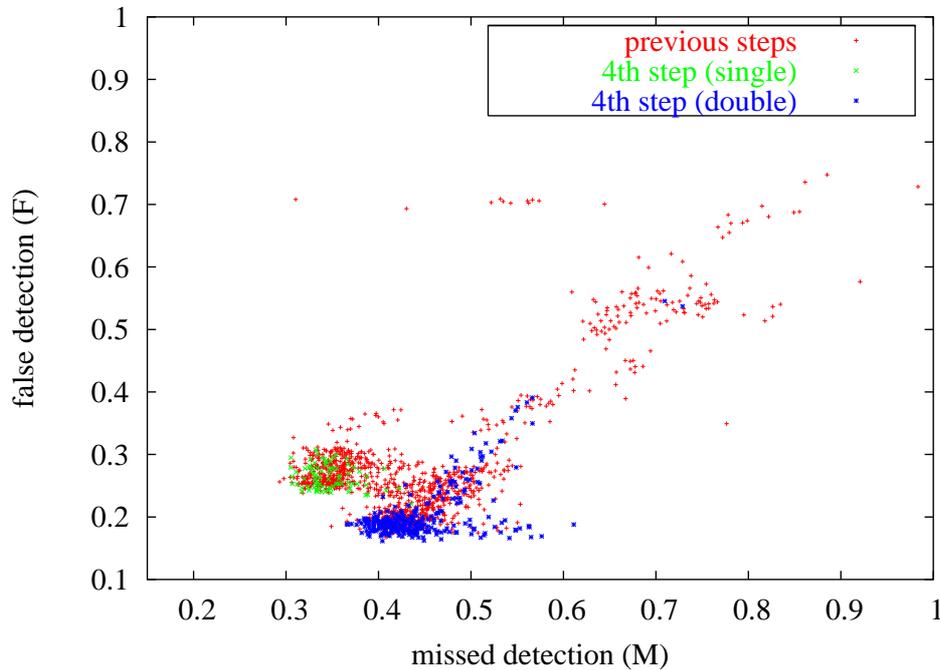


Figure 3.18: Scatter plot of the segmentation quality measures for the fourth step of the Stepwise Forward Selection for the selection of the best parameter sets. Measures using a single segmentation phase are shown in green, and those using two segmentation phases are shown in blue. The results of the previous SFS steps are also shown, in red.

3.4.4 Two-phase segmentation threshold condition

In section 3.2.2 I described a two-phase segmentation procedure for which two thresholds, t_{high} and t_{low} , must be defined. For the tests of the previous section, we took $t_{\text{low}} = 200 \text{ m}^2$ and $t_{\text{high}} = 400 \text{ m}^2$. In this section we explore the optimal value of these thresholds while keeping the remaining parameters as given by the optimal parameter set selected in the previous section, set 3.597.26. Note, however, that since no region-based texture parameter was selected in the optimal parameter sets of the previous section, two-phase segmentation has less interest, now being reduced to a device for obtaining simpler, more manageable, and more quickly computed segmentations.

Figure 3.26 and table B.5 (in the appendices) show the resulting quality measures for several segmentations using parameter set 3.597.26 and different values of t_{low} and t_{high} . Point marks in the figure indicate the value of t_{low} , and points with equal value of $t_{\text{high}}/t_{\text{low}}$ are joined by lines. Also shown in the figure is the Pareto limit of the previous section. Since t_{low} and t_{high} are continuous, I expected the curves of figure 3.26 to be smooth, looking like figure 3.25. They are not smooth at all —although of course $t_{\text{high}}/t_{\text{low}}$ was not scanned in a continuous way— which can indicate either an implementation error, or that the algorithm depends non-linearly on these thresholds. I believe that we can discard the former possibility, since, although the lines are not smooth, a certain pattern does emerge. The latter is indeed unfortunate: with a non-linear dependence, it becomes harder to select thresholds that will give good results for a range of input images.

We notice that, as expected, higher size thresholds correspond to worse missed detection measures and better false detection measures, simply because there are less edge boundaries

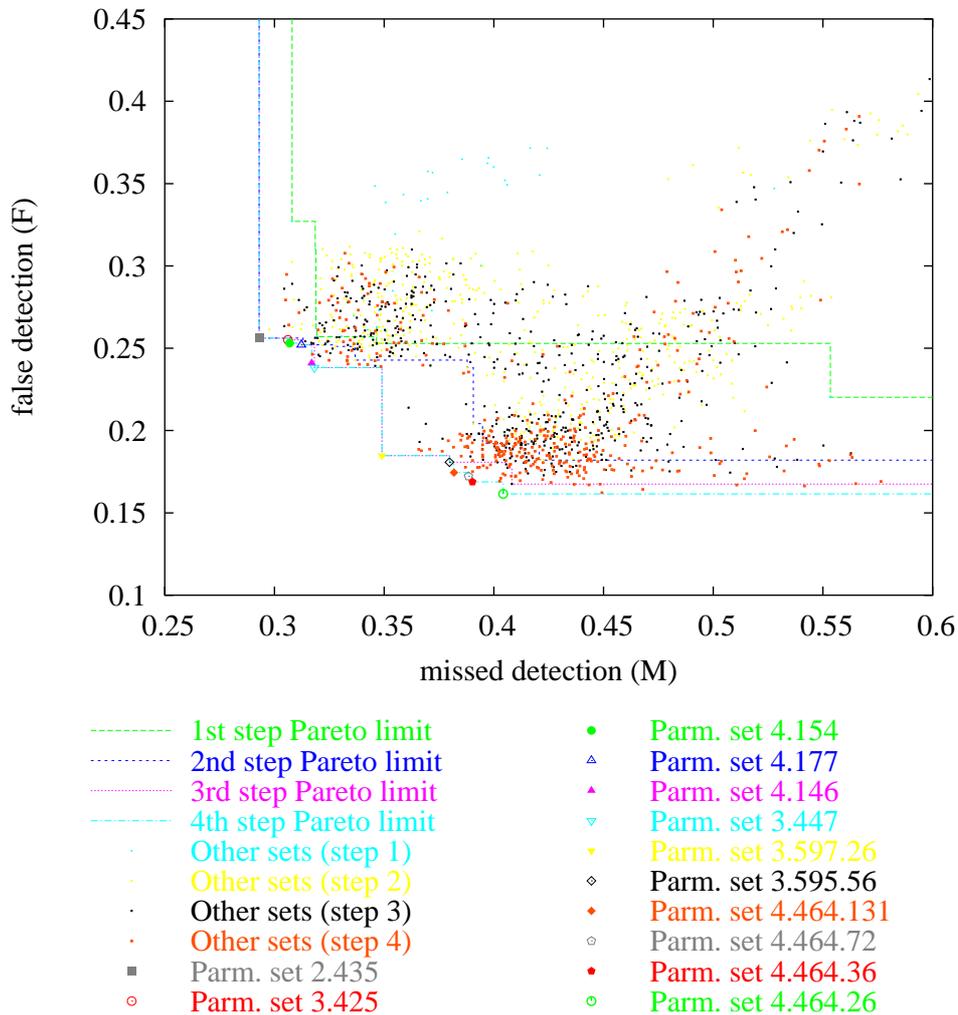


Figure 3.19: Close-up of the scatter plot of the segmentation quality measures for the fourth step of the SFS, showing the Pareto-optimal parameter sets, and the Pareto limit (all points in the upper-right side of this limit are worse than some Pareto-optimal parameter). Parameter set numbers correspond to those in tables B.4, B.3, and B.2. The Pareto limit for the previous SFS steps is also shown. The remaining parameter sets also shown as smaller dots.

in the segmentation. However, the values $t_{\text{low}} = 200 \text{ m}^2$ and $t_{\text{high}} = 400 \text{ m}^2$ seem to provide a good compromise between these two quality measures, and they will be retained for the following sections.

For illustration, figures 3.27, 3.28, and 3.29 show some image segmentations with different values of the t_{low} and t_{high} parameters. The source image is that of figure 3.22. Figure 3.27 is with $t_{\text{low}} = t_{\text{high}} = 10 \text{ m}^2$. Figure 3.28 is with $t_{\text{low}} = 200 \text{ m}^2$ and $t_{\text{high}} = 400 \text{ m}^2$; these are the thresholds selected for the following sections. Finally, figure 3.29 is with $t_{\text{low}} = 1000 \text{ m}^2$ and $t_{\text{high}} = 3000 \text{ m}^2$. Darker edges appear at coarser analysis scales in the multiscale segmentation, and are therefore supposed to correspond —if the parameter set is well chosen— to more salient edges in the source image.

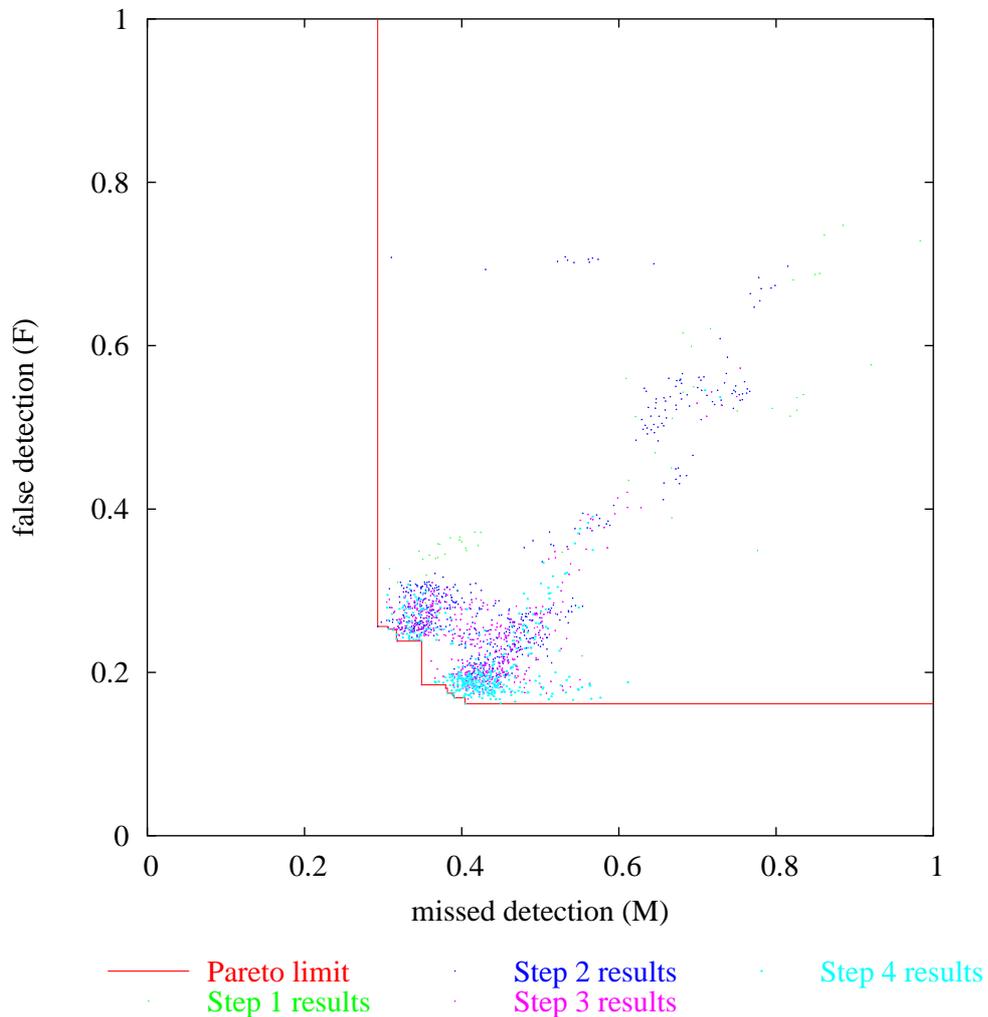


Figure 3.20: Scatter plot of the segmentation quality measures for the parameter sets evaluated in each SFS step, and final Pareto limit.

3.4.5 Complexity term for region boundaries

In section 3.2.3 we recall the shape regularisers proposed by Guigues in [Gui04]. These regularisers, the $C(P)$ term in equation (3.9), are defined as energies depending on the shape of the boundaries of a certain image partition. For the tests of the previous sections, we used the T_σ energy of equation (3.18). In this section we explore the performance of the other proposed energies.

Figure 3.30 and table 3.2 show the resulting quality measures for several segmentations using parameter set 3.597.26, $t_{\text{low}} = 200 \text{ m}^2$ and $t_{\text{high}} = 400 \text{ m}^2$ as thresholds for the two-phase segmentation stop criterion, and different complexity energies. Also shown in the figure is the Pareto limit of section 3.4.3. Notice the wide variation in performance among the several energies. It is clear that the T_σ energy of equation (3.18) performs better than the rest. It will be retained for the following sections.

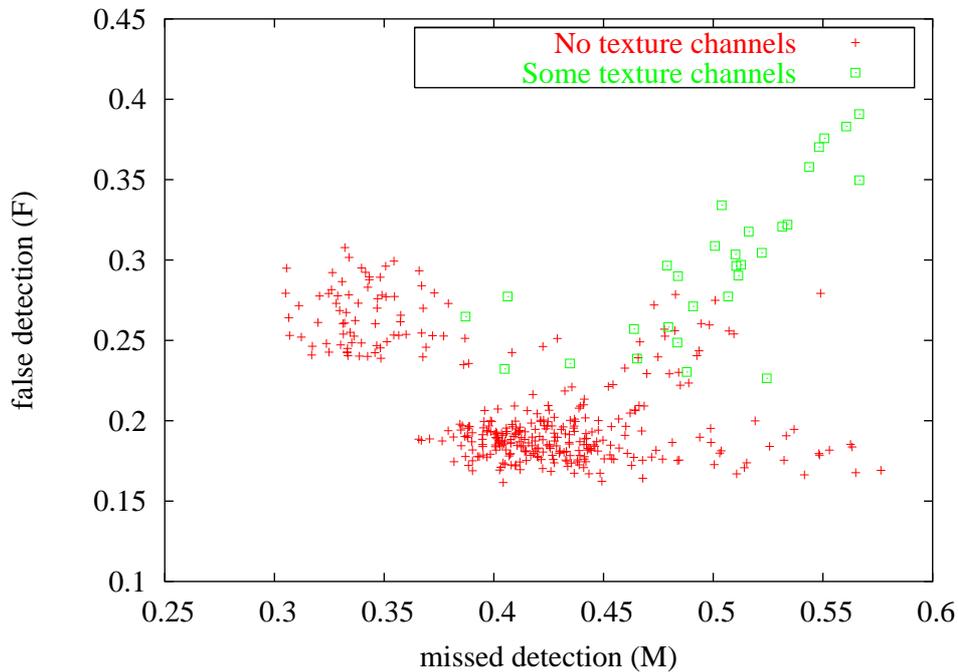


Figure 3.21: Scatter plot of the segmentation quality measures after the fourth step of the SFS, separating test sets which use at least on texture feature and those that use none.

Shape energy $C(P)$	Missed detection (M)	False detection (F)
K (constant)	0.3814	0.2663
L_σ (length)	0.4949	0.2792
T_σ (number)	0.3489	0.1847
P_σ (polar)	0.5012	0.2862
P_σ^N (Gaussian polar)	0.4694	0.2797
C_σ (concavity)	0.7038	0.5194

Table 3.2: Segmentation performance of different shape regularizers.

3.4.6 Gradient-based data fit

In section 3.2.4 we describe a suggestion made by Guigues in [Gui04] on how to modify the complexity term in order to include the image gradient under the segmentation boundaries. In this section we explore the effect of using the two suggested modifications, equations (3.23) and (3.24), for different values of the parameters k and a used in these expressions.

Figure 3.31 and table B.6 (in the appendices) show the resulting quality measures for several segmentations using parameter set $3.597.26$, $t_{\text{low}} = 200 \text{ m}^2$, $t_{\text{high}} = 400 \text{ m}^2$, $C(P) = T_\sigma$, and different values of k and a for the gradient-based data fit $f(G) = 1/(k + G^a)$ of equations (3.22) and (3.23). Similarly, figure 3.32 and table B.7 (in the appendices) show results for different values of k and a for $f(G) = \exp(-kG^a)$ of equations (3.22) and (3.24). Also shown in the figure is the Pareto limit of section 3.4.3, and the results without the gradient-based data fit modification (that is, with $f(G) = 1$).

We can see that this modification does not have the expected beneficial effect, neither for



Figure 3.22: Part of the source image whose segmentation is shown in figures 3.23 and 3.24. North is to the left of the page.

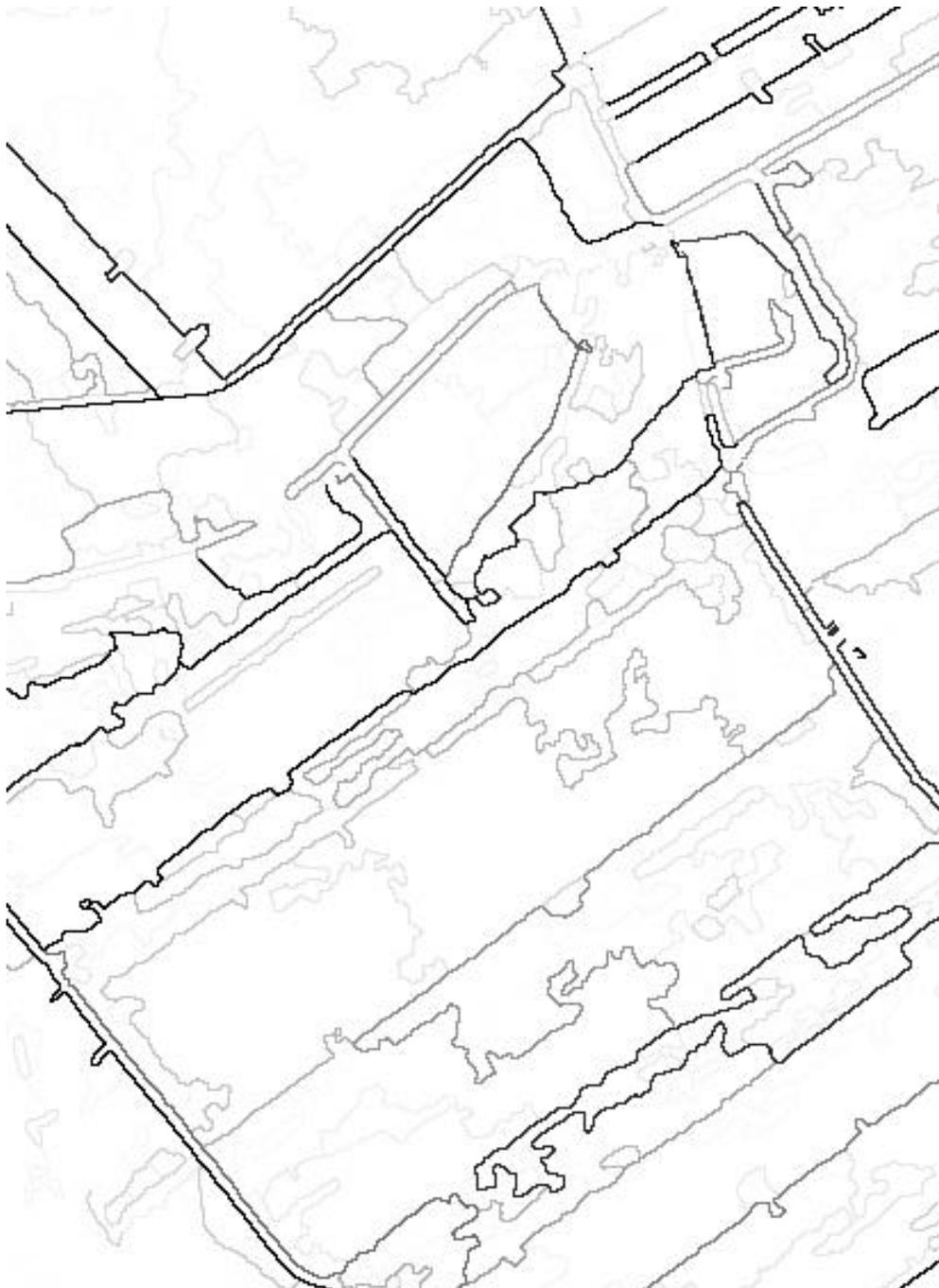


Figure 3.23: Segmentation of the image shown in figure 3.22 using parameter set 3.597.26, one of the optimal sets, with quality measures $M = 0.3490$ and $F = 0.1848$. Darker edges are supposed to correspond to more salient edges in the source image, and in this case they do, hence the good quality measures of this set. North is to the left of the page.

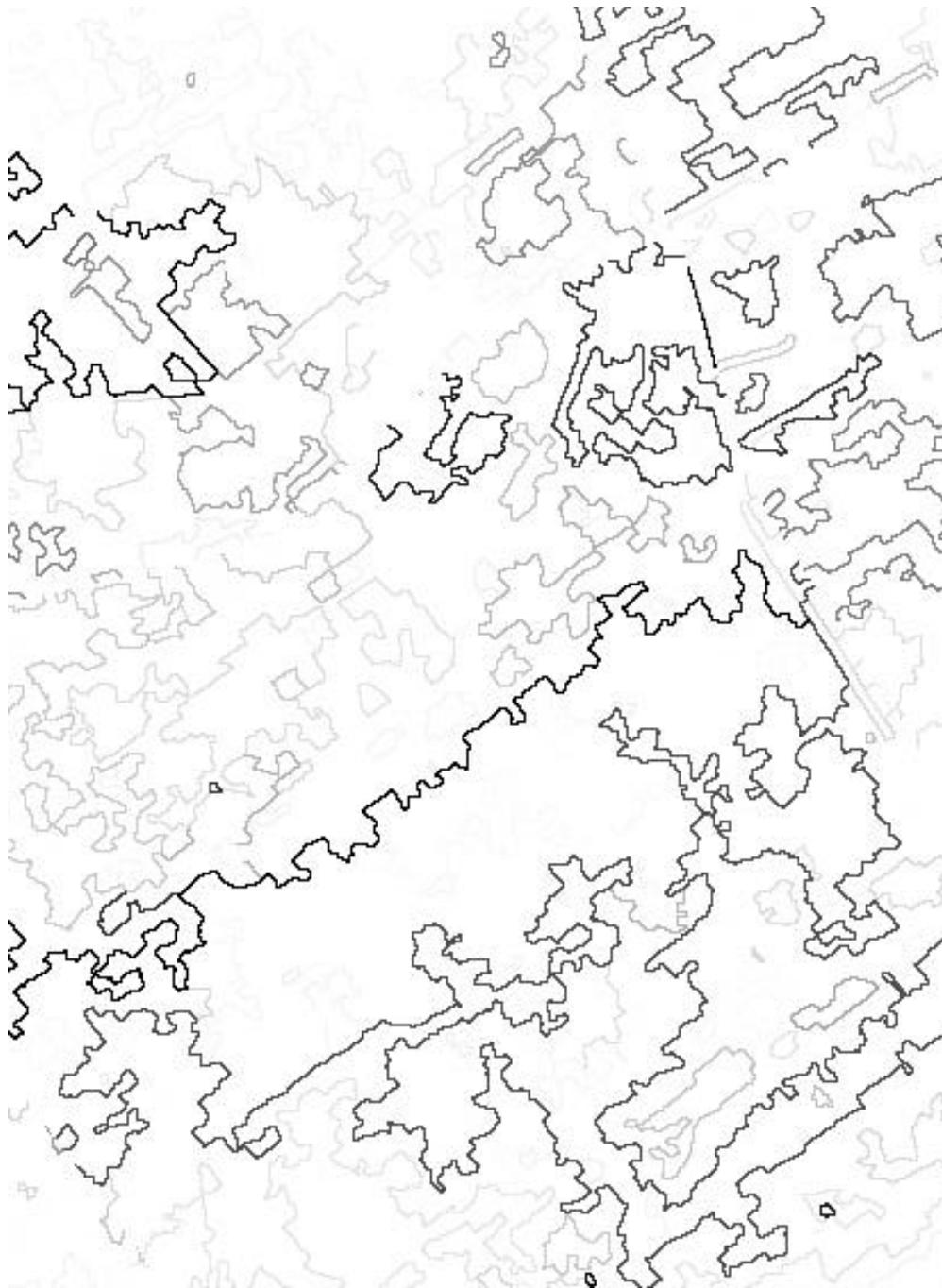


Figure 3.24: Segmentation of the image shown in figure 3.22 using parameter set 2.544.35, an average badly-performing set, with quality measures $M = 0.7721$ and $F = 0.6471$. Darker edges are supposed to correspond to more salient edges in the source image, but in this case the correlation is weak, hence the bad quality measures of this set. North is to the left of the page.

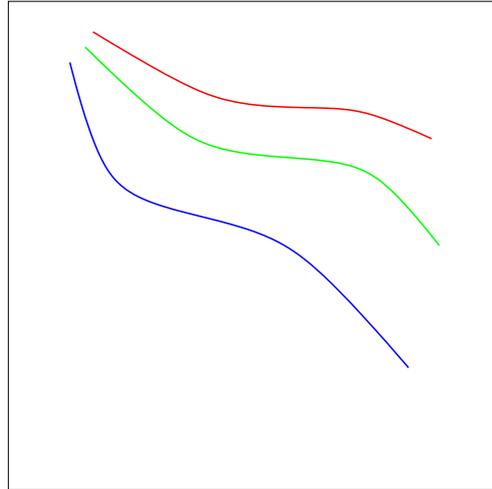


Figure 3.25: Expected behaviour of the segmentation quality measures for varying t_{low} and $t_{\text{high}}/t_{\text{low}}$. The actual behaviour is given in figure 3.26.

$f(G) = 1/(k + G^a)$ nor for $f(G) = \exp(-kG^a)$. Indeed, for all but the highest values of a , the effects seem to be mostly random—notice how the graph points are clustered in a small region, and that the lines joining points of equal a do not seem to follow any order. Even for the highest values of a , where some trend can be seen, it gives much better results to simply use $f(G) = 1$, that is, not to use the gradient-based data fit modification to the complexity energy. Therefore, for the next and final section, we shall continue to use $f(G) = 1$.

3.4.7 Fit to local cadastre orientation

In section 3.2.5 I proposed anisotropic versions of Guigues' geometric energies, in an attempt to take into account a priori knowledge of predominant field edge orientations which can be obtained from cadastre data. In particular, I proposed modifications to the L_σ , T_σ , P_σ , P_σ^N , and C_σ energies in which the terms of *edge length* and *angle at vertex* are modified so that edges oriented along a predominant orientation are taken as shorter—and so less expensive, therefore biasing the solution towards such edges—and pairs of coincident edges at predominant angles are also taken as having a “less expensive” angle, in terms of geometry energy.

In this section, I have tested the performance of the segmentation algorithm for these anisotropic geometric energies: L_σ^W of equation (3.47), T_σ^W of equation (3.48), P_σ^W of equation (3.49), $P_\sigma^{W,N}$ of equation (3.50), and C_σ^W of equation (3.52). For the $P_\sigma^{W,N}$ and C_σ^W energies, which take angles into account, I have tested both the integral definition of anisotropic angles, $\text{ang}_{f_{W,z}}$ of equation (3.33), and the end-angles definition, $\text{ang}_{W,z}$ of equation (3.39), as well as the standard, isotropic angles. I have also tested several values of m for equation (3.28). As we will see, the results are dissatisfying, since not only we obtain better results with the isotropic variants, but the anisotropic variants seem to behave erratically in some cases. The latter might indicate a programming error, but the fact that some of the variants behave non-erratically—albeit with worse performance than the isotropic variants—seem to suggest that there is actually a problem in the formulation of the anisotropic lengths and angles themselves. Unfortunately, I could not conduct further research into this topic.

I did not attempt to evaluate the anisotropic variant of the gradient-based data-fit term $E^{f,W}$ of equation (3.53).

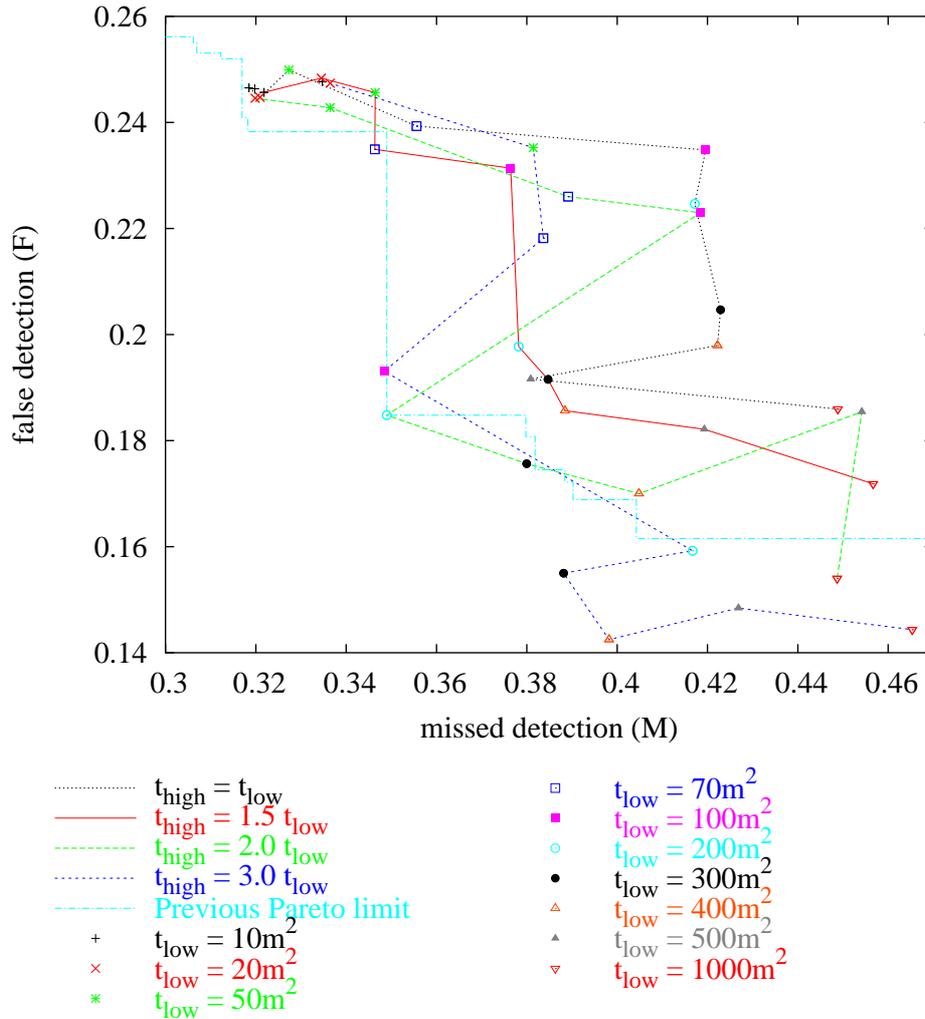


Figure 3.26: Quality measures for several segmentations using parameter set 3.597.26 and different values of t_{low} and t_{high} . Point marks indicate the value of t_{low} , and points with equal value of $t_{\text{high}}/t_{\text{low}}$ are joined by lines. Also shown is the Pareto limit of the previous section.

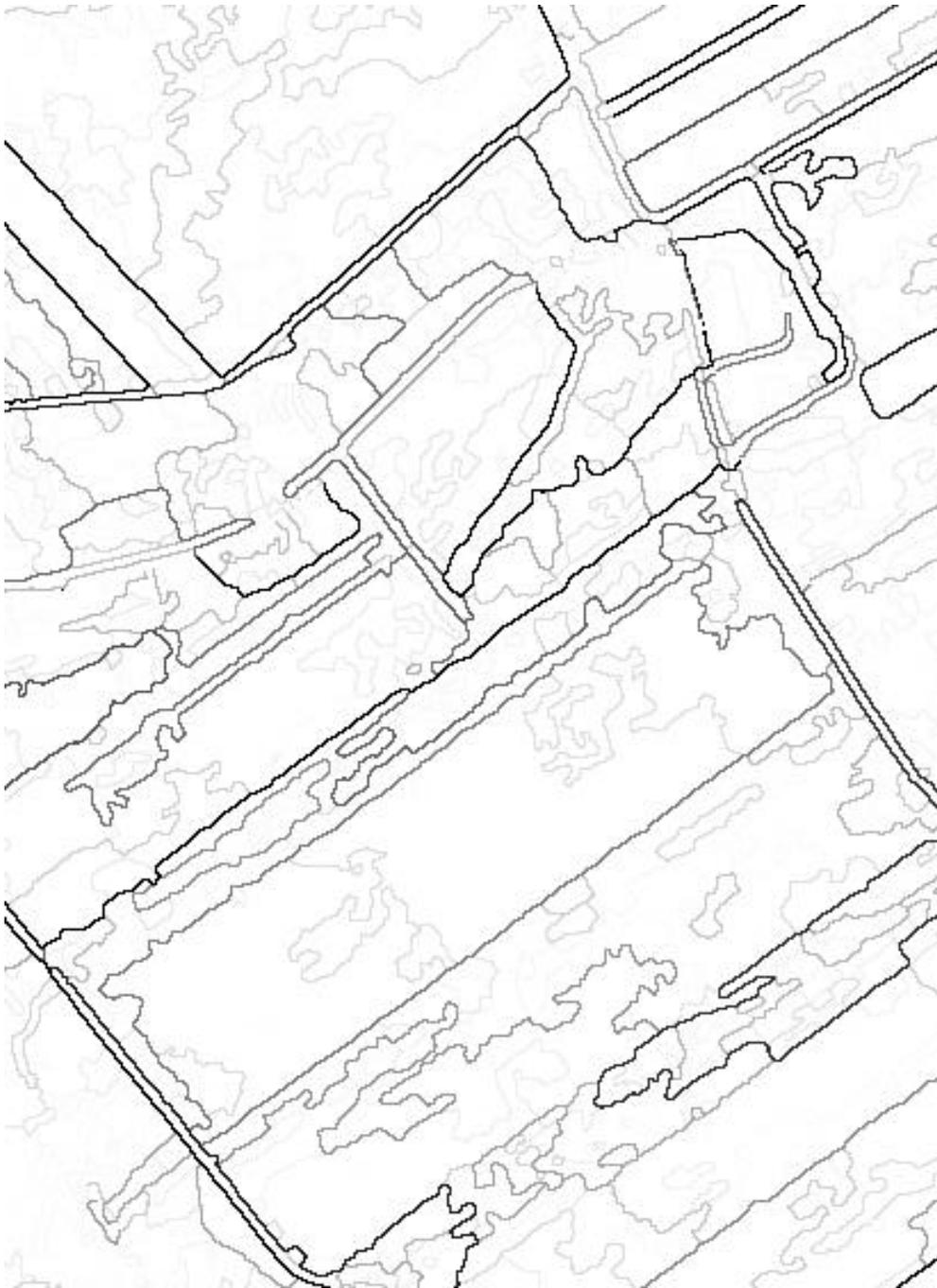


Figure 3.27: Segmentation of the image shown in figure 3.22 using $t_{low} = 10 \text{ m}^2$ and $t_{high} = 10 \text{ m}^2$. Darker edges correspond to coarser scales of analysis. North is to the left of the page.

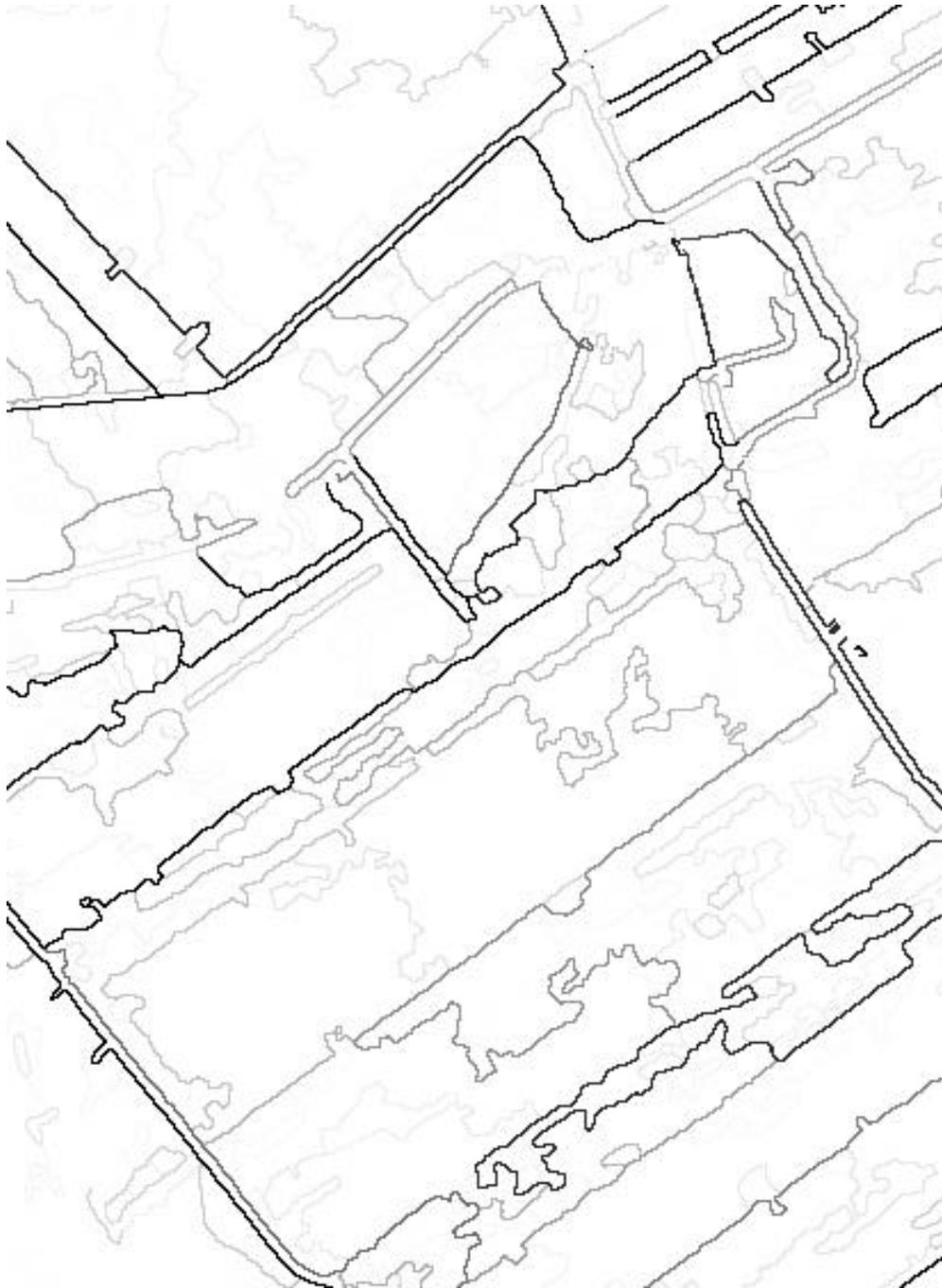


Figure 3.28: Segmentation of the image shown in figure 3.22 using $t_{\text{low}} = 200 \text{ m}^2$ and $t_{\text{high}} = 400 \text{ m}^2$. Darker edges correspond to coarser scales of analysis. North is to the left of the page.

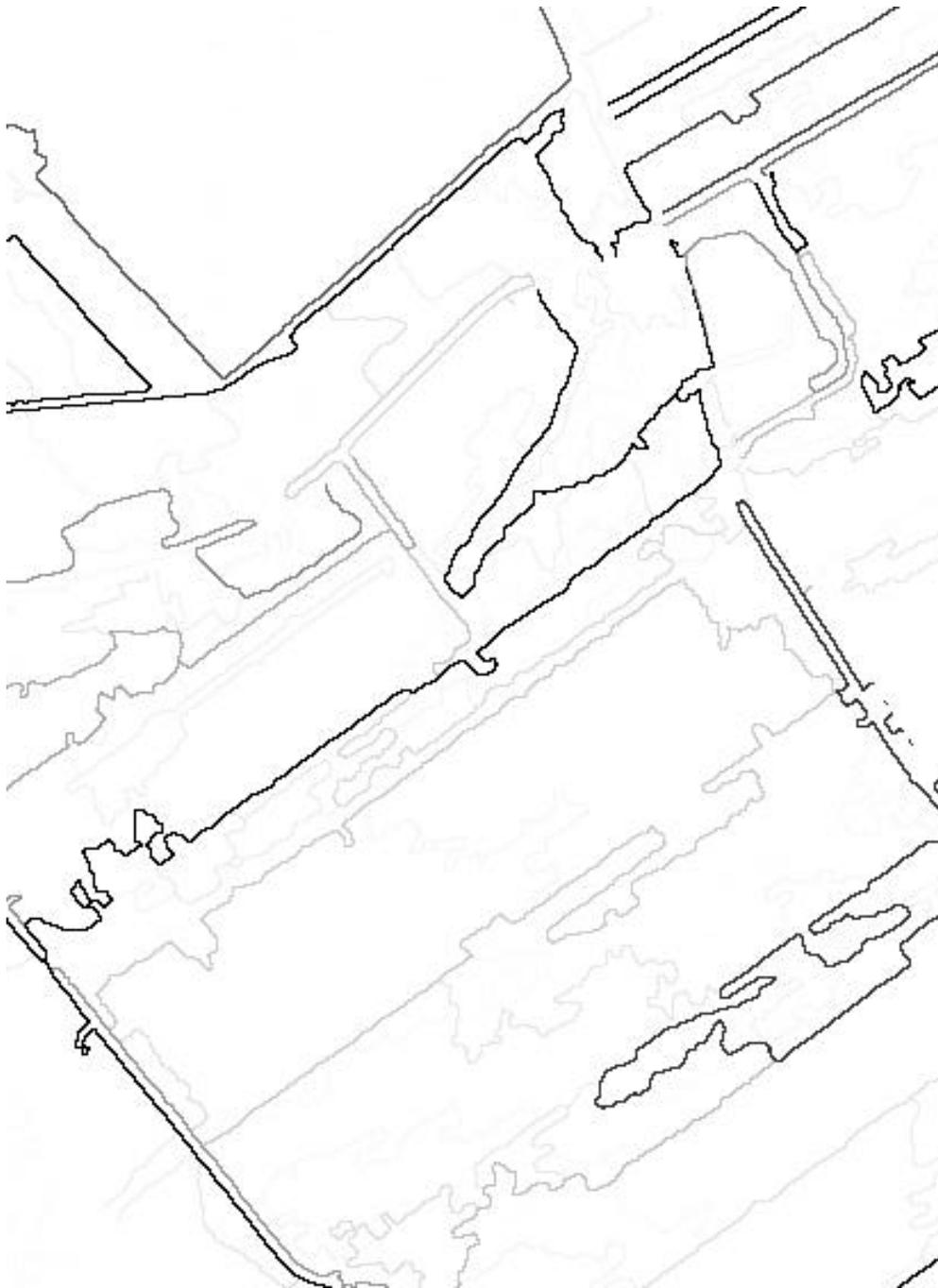


Figure 3.29: Segmentation of the image shown in figure 3.22 using $t_{\text{low}} = 1000 \text{ m}^2$ and $t_{\text{high}} = 3000 \text{ m}^2$. Darker edges correspond to coarser scales of analysis. North is to the left of the page.

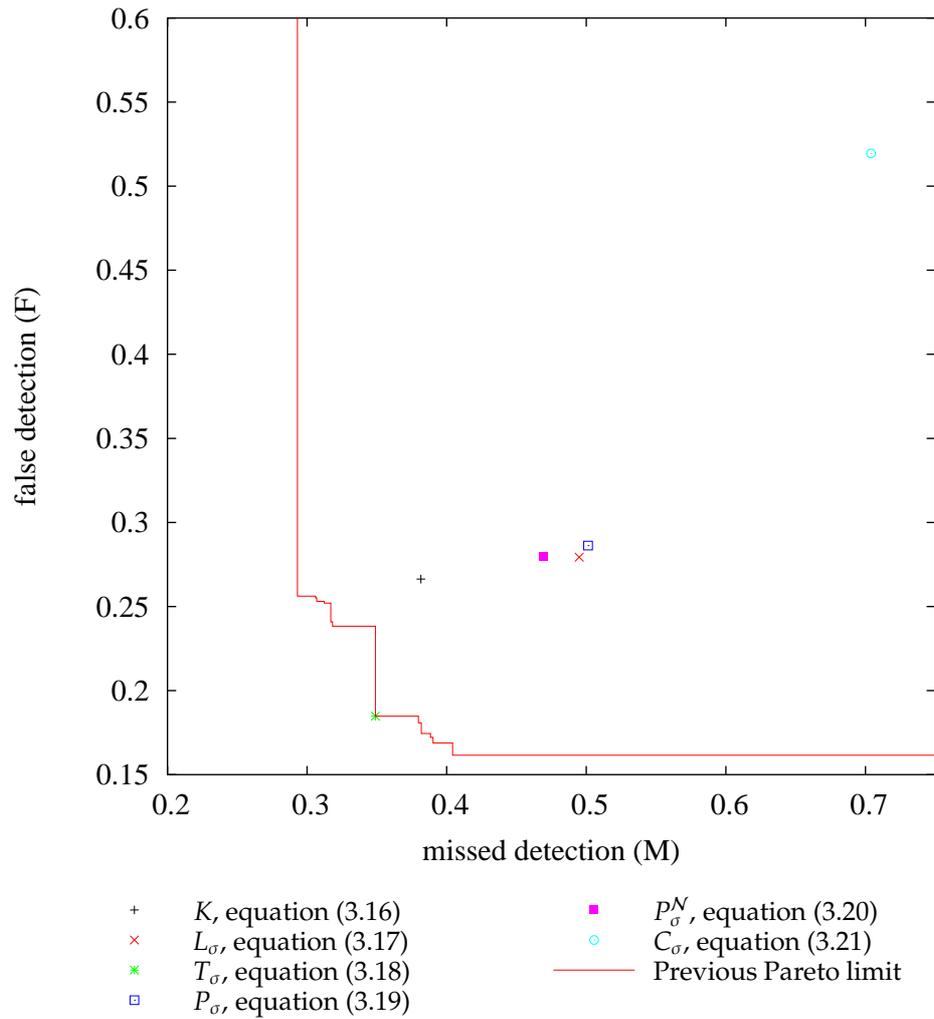


Figure 3.30: Quality measures for several segmentations using parameter set 3.597.26, $t_{\text{low}} = 200 \text{ m}^2$, $t_{\text{high}} = 400 \text{ m}^2$ and several complexity energies $C(P)$ in equation (3.9). Also shown is the Pareto limit of the previous sections.

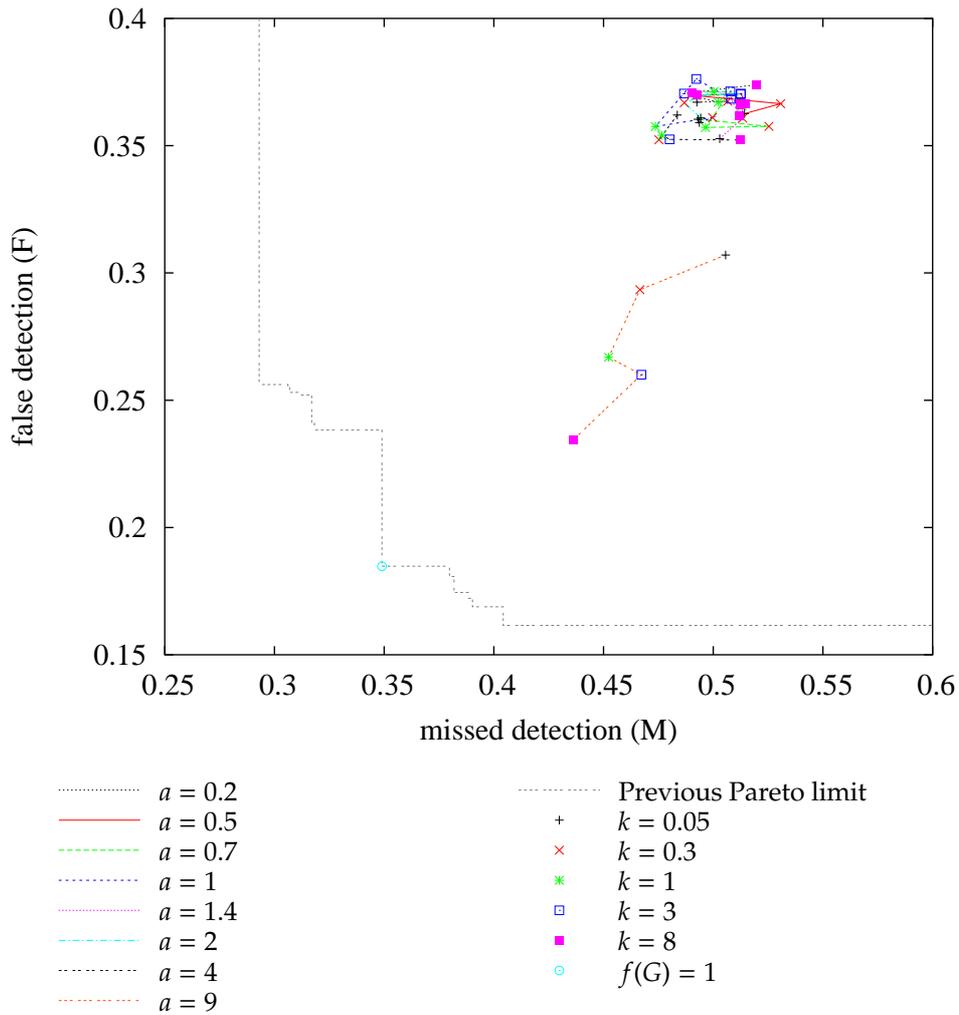


Figure 3.31: Quality measures for several segmentations using parameter set 3.597.26, $t_{\text{low}} = 200 \text{ m}^2$, $t_{\text{high}} = 400 \text{ m}^2$, $C(P) = T_{\sigma}$, and several values of k and a for a gradient-based data fit modification to the complexity energy of $f(G) = 1/(k + G^a)$ of equations (3.22) and (3.23). Also shown is the Pareto limit of the previous sections, and the results with $f(G) = 1$.

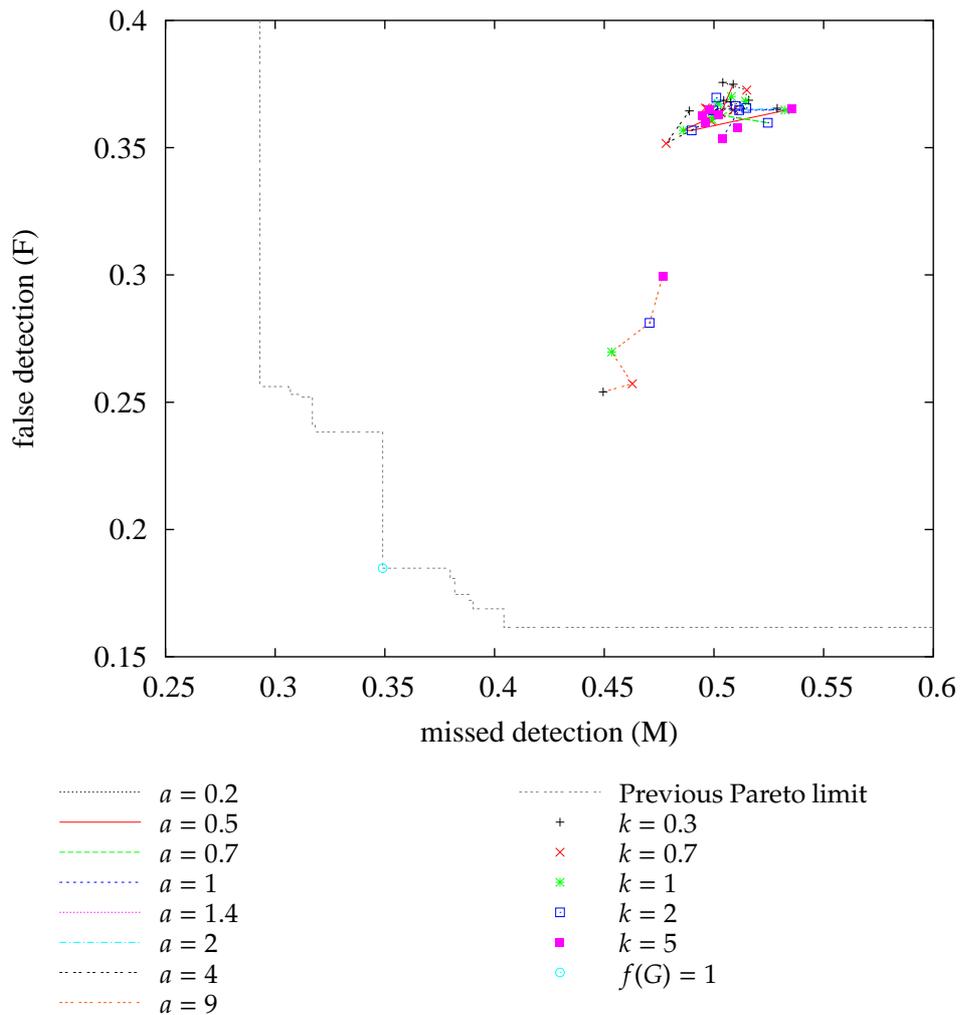


Figure 3.32: Quality measures for several segmentations using parameter set 3.597.26, $t_{\text{low}} = 200 \text{ m}^2$, $t_{\text{high}} = 400 \text{ m}^2$, $C(P) = T_\sigma$, and several values of k and a for a gradient-based data fit modification to the complexity energy of $f(G) = \exp(-kG^a)$ of equations (3.22) and (3.24). Also shown is the Pareto limit of the previous sections, and the results with $f(G) = 1$.

Figure 3.33 shows the resulting quality measures for several segmentations using parameter set 3.597.26, $t_{\text{low}} = 200 \text{ m}^2$, $t_{\text{high}} = 400 \text{ m}^2$, and no gradient-based data-fit, for all anisotropic variants of the geometric energies $C(P)$, with different definitions for anisotropic angles where appropriate, and different values of the parameter m . Notice that none of these results is better than the previous Pareto limit, obtained with isotropic energies.

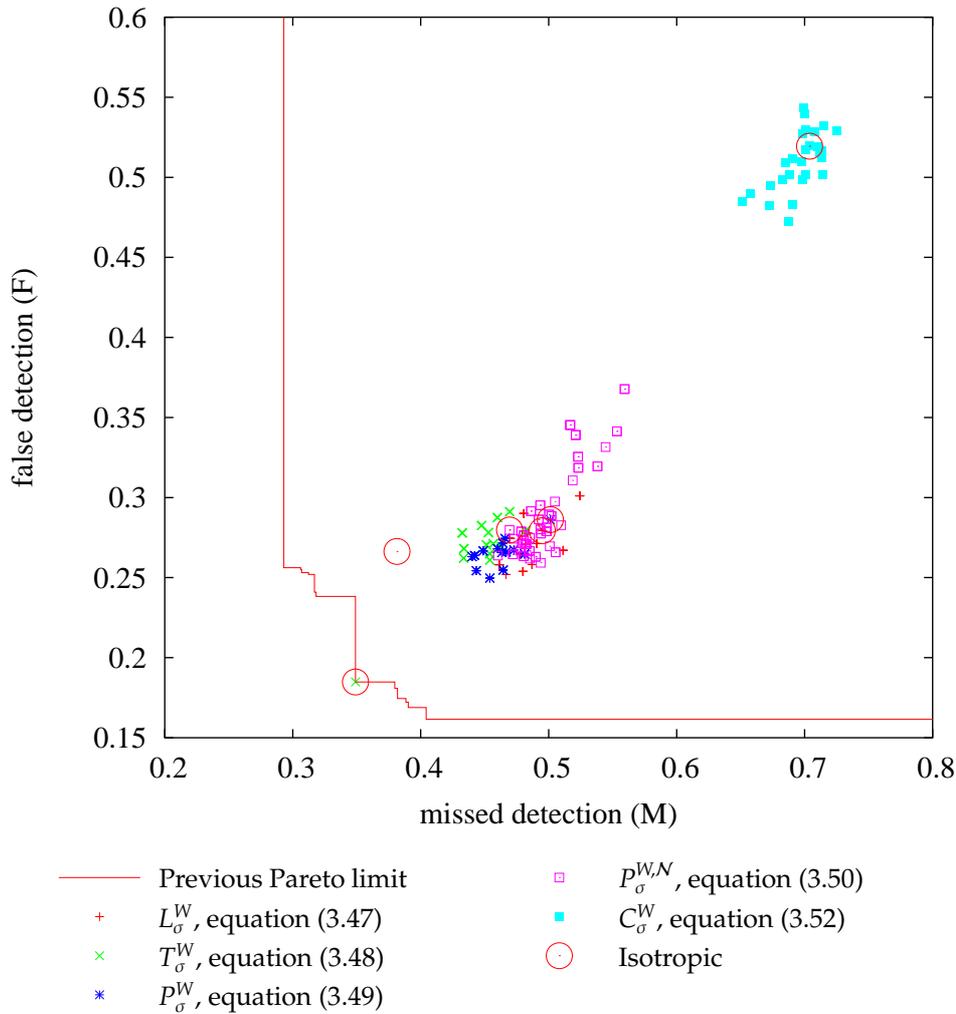


Figure 3.33: Quality measures for several segmentations using parameter set 3.597.26, $t_{\text{low}} = 200 \text{ m}^2$, $t_{\text{high}} = 400 \text{ m}^2$ and several *anisotropic* complexity energies $C(P)$ in equation (3.9). Also shown is the Pareto limit of the previous sections, and the results for isotropic geometric energies of section 3.4.5.

For the $P_\sigma^{W,N}$ and C_σ^W , an anisotropic definition of angles is needed. Figures 3.34 and 3.35 show the results for $P_\sigma^{W,N}$ and C_σ^W respectively, separately for isotropic angles, for the anisotropic angles $\text{ang}_{\int W_z}$ defined in equation (3.33), and for the anisotropic angles ang_{W_z} of equation (3.39). Also shown, for reference, are the performances of the corresponding isotropic energies P_σ^N and C_σ respectively.

It may be interesting to study how these anisotropic energies affect the global performance as the value of m in equation (3.28) is varied. Each graph in figures 3.36 and 3.37 shows the

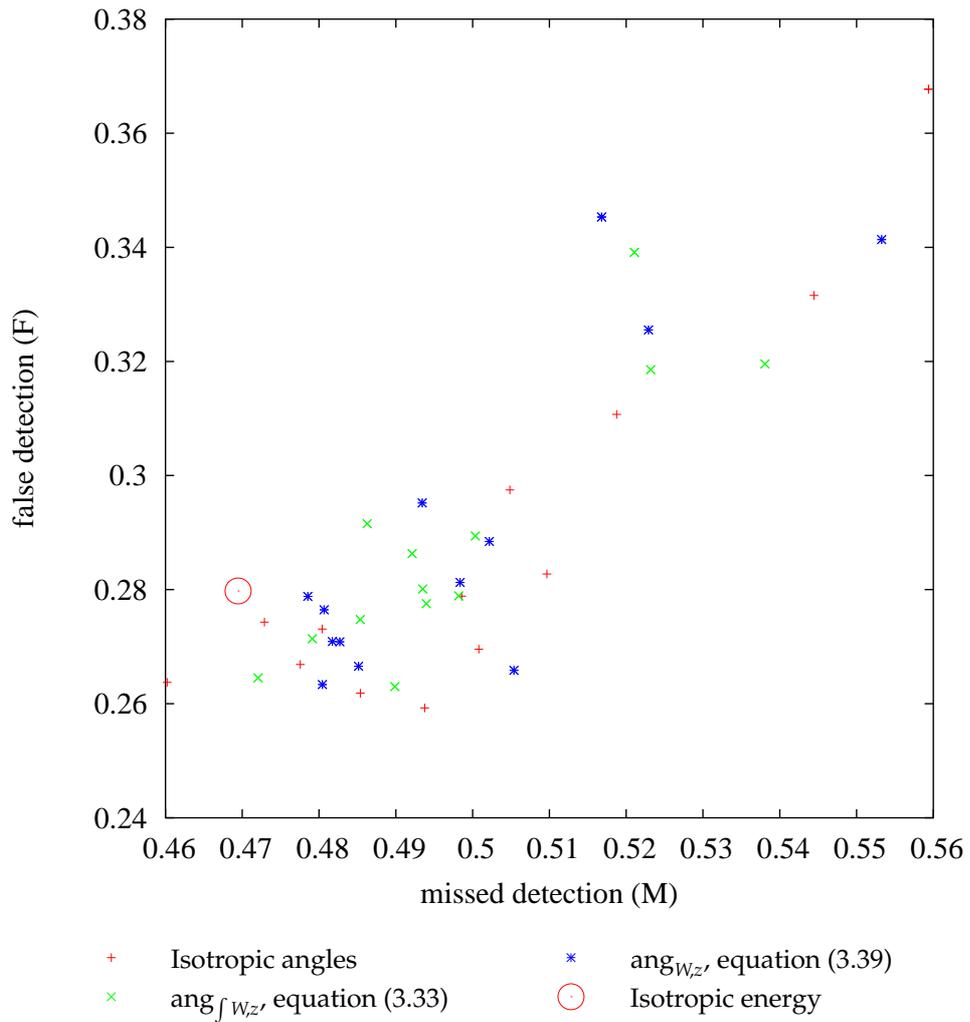


Figure 3.34: Quality measures for several segmentations using parameter set 3.597.26, $t_{\text{low}} = 200 \text{ m}^2$, $t_{\text{high}} = 400 \text{ m}^2$ and the anisotropic geometric energy $P_{\sigma}^{W,N}$, separately for each definition of angle.

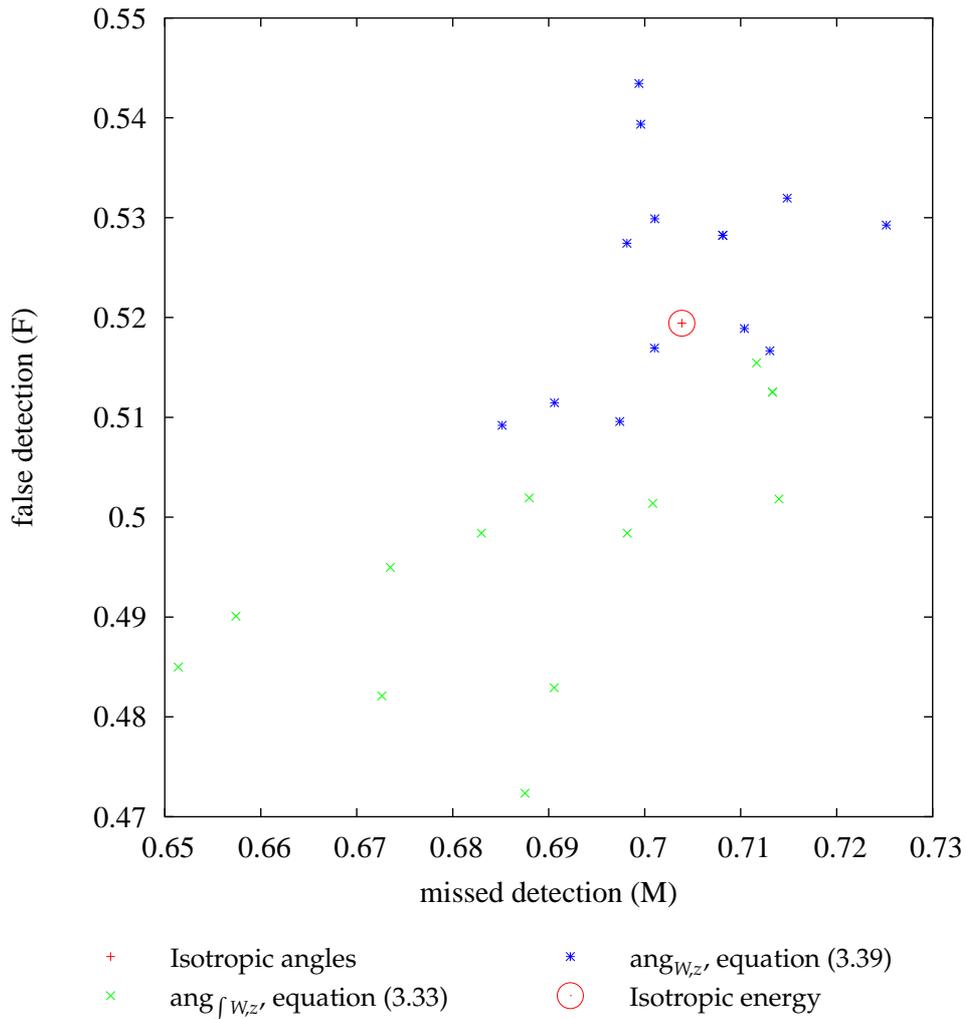


Figure 3.35: Quality measures for several segmentations using parameter set 3.597.26, $t_{\text{low}} = 200 \text{ m}^2$, $t_{\text{high}} = 400 \text{ m}^2$ and the anisotropic geometric energy C_{σ}^W , separately for each definition of angle.

quality measures for a single anisotropic energy (with a single definition for angles where appropriate), as m takes the values 0.001, 0.002, 0.005, 0.01, 0.02, 0.05, 0.1, 0.2, 0.5, 1, 2, 5, and 10. A line joins data points as m advances. The data point marked with a triangle corresponds to $m = 0.001$. Also shown, with a circle, is the data point for the corresponding isotropic energy. Note that we do not show results for C_σ^W with isotropic angles, since in that case the energy does not depend on m . We notice that in many cases (T_σ^W , P_σ^W , C_σ^W , and, in part, L_σ^W) the performance behaves erratically as m increases, suggesting either a programming error or that the effect of m is negligible. In other cases ($P_\sigma^{W,N}$ and, in part, L_σ^W), m seems to have a more definite effect on the performance, and, more importantly, the larger the m —and thus, the stronger the anisotropy—the better the performance.

It seems, therefore, that, although the anisotropic formulation in its current state is not adequate, further research might provide better results. Unfortunately, I could not conduct further research into this topic.

For reference, table B.8 (in the appendices) lists the full numerical results for this section.

3.5 Conclusion

As the first step in the processing chain for the system presented in this thesis, the input images are segmented. The result is a partition of the image into small homogeneous regions, where the boundaries between regions are attributed according to their saliency or strength. This partition is used in two further processing steps: First, when registering the cadastre onto the image (chapter 4), it is these boundaries that the cadastre will be registered onto. Second, if no cadastre data is available, the per-region classification algorithms of chapter 5 will use this partition and classify each segmentation region separately.

When high-resolution imagery is available, there is the possibility of using not only the raw image radiometry channels, but also to extract textural features and use them for segmentation. In that way, it is hoped, the patches obtained by segmentation will not be homogeneous in colour but in texture, perhaps giving a result closer to the ideal of semantically significant objects. It may also be that using transformed colour spaces, such as log-opponent chromaticities or the hue-saturation-value space, improves the results.

To explore this question, in this chapter I have presented a review of several transformed colour spaces and textural features which may be adequate for the problem of segmenting a high-resolution aerial image of rural areas. I have presented some minor adaptations of some of these textural features, and suggested a way of dealing with the fact that textural features should be calculated on texture-homogeneous regions, but these regions are not available before the segmentation itself. These texture descriptors and transformed colour spaces will also be used for classification in chapter 5.

Furthermore, in order to avoid the problems associated with single-scale segmentations, a multi-scale segmentation algorithm is used. Because of this, however, I had to develop a method for evaluating a multi-scale image segmentation, which is also presented. The selected segmentation algorithm is also described in this chapter.

Finally, I have presented a thorough evaluation in which I have tested a large number of combinations of input channels, in order to determine which raw radiometry channels, transformed colour channels, or texture features give the best performance for the specific problem of high-resolution aerial images of rural areas. The conclusion is that texture, which at first seems promising, does not in fact improve the results, and that input combinations that do not use texture perform significantly better. I believe that this is because of the non-local nature of texture. On the other hand, using transformed colour channels gives also better results

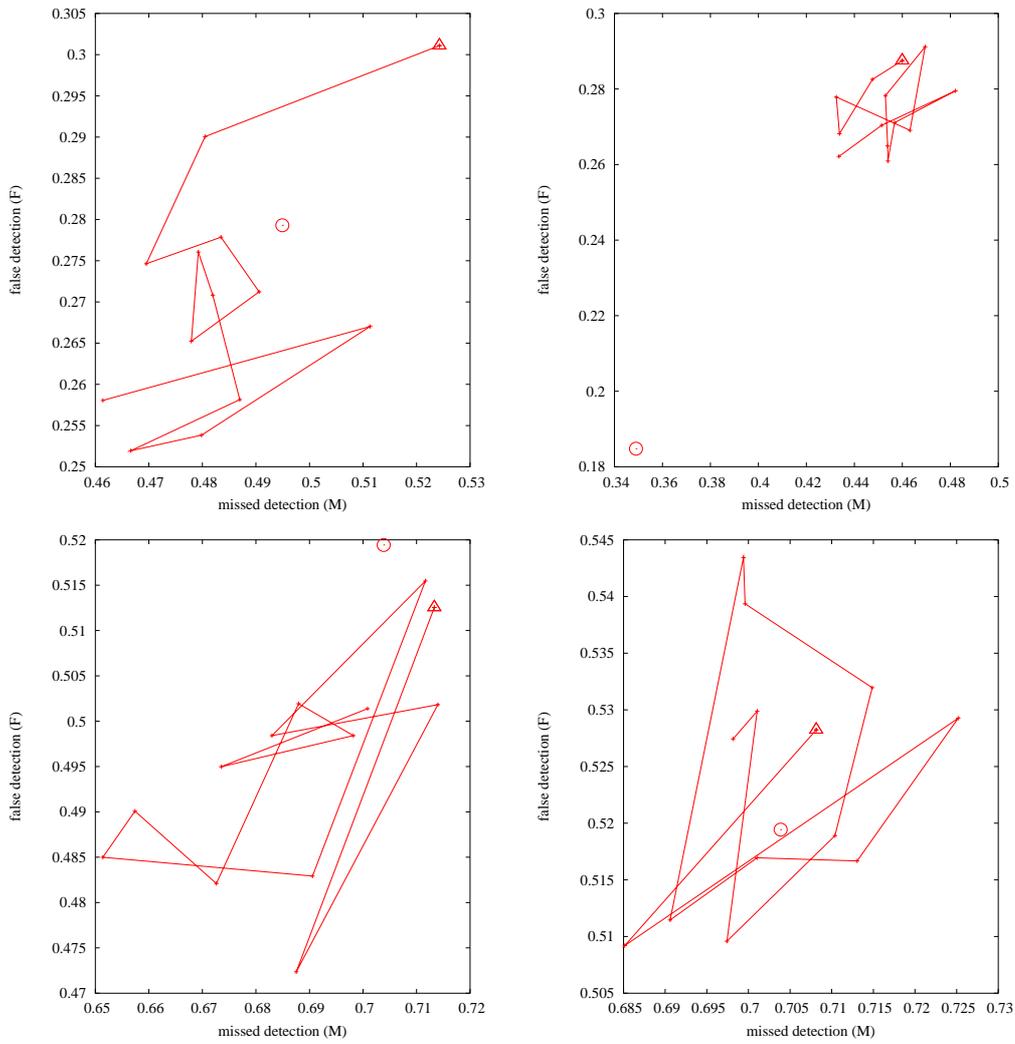


Figure 3.36: Quality measures for several segmentations using parameter set 3.597.26, $t_{\text{low}} = 200 \text{ m}^2$, $t_{\text{high}} = 400 \text{ m}^2$, and several anisotropic geometric energy and angles, for increasing values of m . The triangle indicates $m = 0.001$, and the circle the corresponding isotropic energy, for reference. Top left: L_{σ}^W , top right: T_{σ}^W , bottom left: C_{σ}^W with $\text{ang}_{\int W_z}$, bottom right: C_{σ}^W with ang_{W_z} .

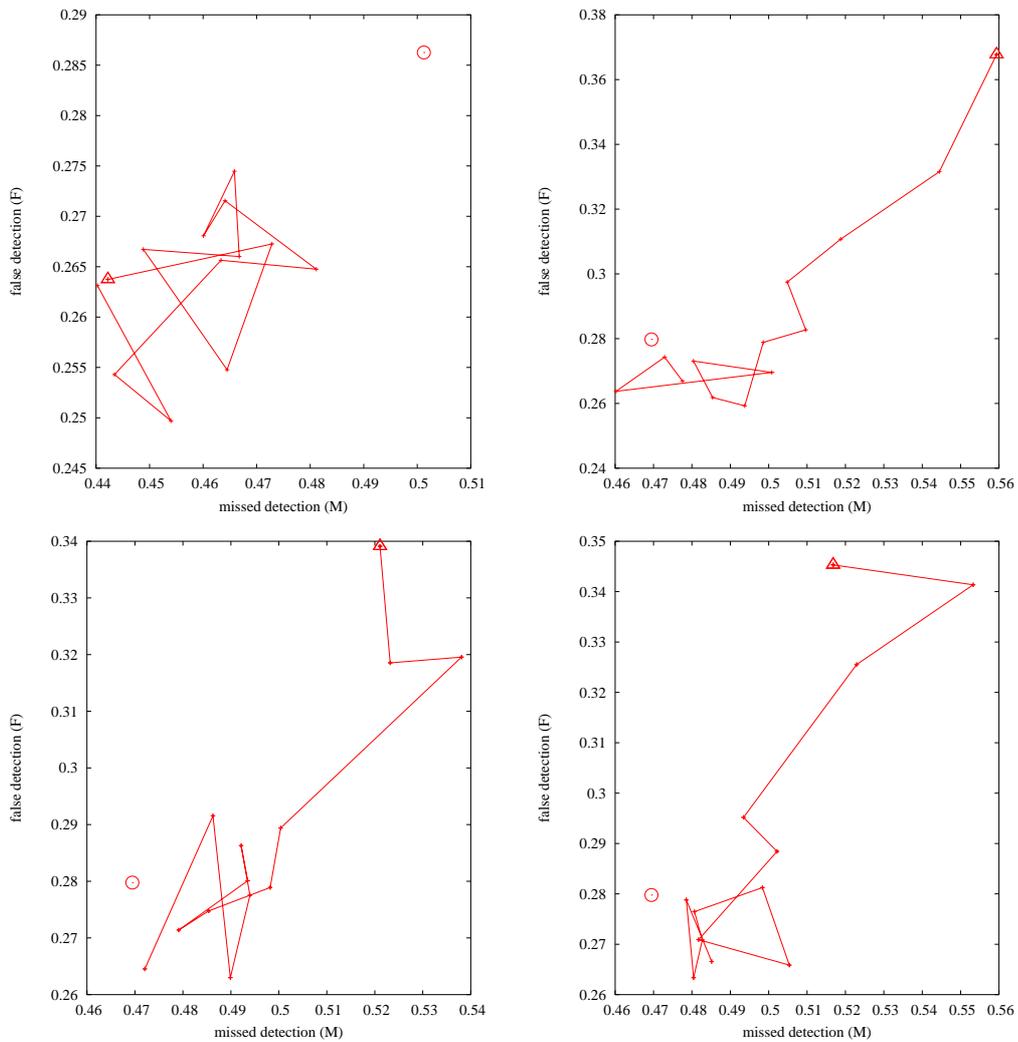


Figure 3.37: Quality measures for several segmentations using parameter set 3.597.26, $t_{\text{low}} = 200 \text{ m}^2$, $t_{\text{high}} = 400 \text{ m}^2$, and several anisotropic geometric energy and angles, for increasing values of m . The triangle indicates $m = 0.001$, and the circle the corresponding isotropic energy, for reference. Top left: P_{σ}^W , top right: $P_{\sigma}^{W,N}$ with isotropic angles, bottom left: $P_{\sigma}^{W,N}$ with $\text{ang}_f_{W,z}$, bottom right: $P_{\sigma}^{W,N}$ with $\text{ang}_{W,z}$.

than using the raw red, green, and blue channels. Note that, as we will see in chapter 5, this conclusion does not apply to the input data to be used for terrain classification. Finally, I have also shown the behaviour of the chosen segmentation algorithm for changes in some of its parameters, and concluded that it is better not to use either the gradient-based data fit proposed by Guigues, or the anisotropic variants proposed in section 3.2.5.

In conclusion, we have seen that, by exploring alternate input channels for the segmentation algorithm, the segmentation quality improves to $M = 0.3490$, $F = 0.1848$ (with $R_1 = \{satint\}$, $R_2 = \{ciez, log-bg\}$) compared to the quality $M = 0.3722$, $F = 0.2569$ of a segmentation using the raw radiometry data (with $R_1 = \{red, green, blue\}$, $R_2 = \emptyset$). The use of texture descriptors does not improve the results. Furthermore, we see that, in any particular application, it may be interesting to check whether transformed channels give better results than the raw radiometry data. These particular results are likely not to be extrapolable to images other than high-resolution aerial images of rural areas, but still there are many applications in that domain which would benefit from not having to repeat these long experiments.

4

Registration of external terrain partitions

4.1 Introduction

As we shall see in chapter 5, pixel-based classification methods produce noisy results which are not suitable for our application. To solve that problem, and to obtain better classification confidence measures which enable semi-automatic classification, in that chapter I will present several region-based classification algorithms. These algorithms process each image region separately, so there is a need to partition the image into such regions. Furthermore, the region-based algorithms of chapter 5 assume that regions are homogeneous, that is, each region contains only one type of terrain.

A simple way to obtain such partition into homogeneous regions is to calculate an over-segmentation of the image, using for example the methods of chapter 3. This partitions the image into a great number of small regions. Because segmentation algorithms try to obtain radiometrically homogeneous regions (or, in our case, also texturally homogeneous), and the algorithm is in this case not forced to produce large regions, we may safely assume that the obtained regions are homogeneous in terrain type. Experiments using such partitions are indeed described in chapter 5.

Another option is to use exogenous data as a partition. For example, we could use the cadastre. The cadastre partitions the terrain into plots according to fiscal information. In most cases, but not all, cadastre regions are roughly homogeneous in terrain type. They are, in general, much larger than the regions which could be obtained from chapter 3, and as we shall see algorithms in chapter 5 produce less-noisy results when the regions to classify are larger. There is a correspondence, although an incomplete one, between cadastre edges and edges in the desired final output for this system; thus finding an image edge that has a corresponding cadastre edge indicates a higher likelihood that the image edge is a relevant edge in the desired segmentation, and not an over-segmenting edge. In addition, cadastre data may help by providing additional information about crop distribution and position.

However, cadastre edges and their corresponding image edges rarely match exactly in the image, because of errors in the cadastre database and geometrical deformation, but most importantly because the cadastre contains fiscal information, whereas the image shows land use: Land owners are free not to follow cadastre limits when growing their crops; the internal arrangement of crop types in a set of cadastre regions owned by the same farmer may follow only partially, or not at all, the cadastre limits, and the outer limits of the cultivated area never

follow the outer limits of the fiscal region exactly. There is therefore the need to *register* the cadastre onto the image, that is, to slightly deform or move the cadastre edges—separating adjacent plots—so that they correspond to true terrain edges visible in the image. Without registration, many regions would contain at their edges pixels not belonging to their main class, thus complicating the classification.

Registration can also be used when the system described in this thesis is used to update land use databases: Once the initial land use database has been produced using cadastre data, successive runs can use the old land use database as input, instead of the cadastre; it is expected that most edges in the old database and in the current situation match, except for some deformation, which will be corrected by the registration mechanism.

Registration of cadastres to images has been viewed as a non-rigid registration problem—see [EM03a] for an example of rigid registration—. In non-rigid registration problems, the goal is to find the transformation or deformation within a class that best converts an image to the reference image; matching is then trivial. Transformation classes are usually a subset of C^2 functions $\mathbb{R}^2 \rightarrow \mathbb{R}^2$. For example, Viglino and Guigues [VG02] match cadastre graphs to terrain edges by finding the best global transformation from one to the other among the class of polynomial transformations of a certain degree. See Cachier *et al.* [CBD⁺03] and Goshtasby *et al.* [GSST03] for some reviews of non-rigid image registration methods; as they say, most of the work in this domain focuses on medical imaging. Chui and Rangarajan [CR03] propose a non-rigid registration method for sparse sets of points which could be used in some instances of graph matching problems.

These approaches assume that the initial mis-registration is due to sensor-induced deformation, to acquisition errors, or because the relative orientation of the input data is unknown. In our case the image is georeferenced, so the third problem can be ignored, and the initial mis-registration is not due to an acquisition error or a sensor-induced deformation, but rather to the two graphs representing data of different nature (see figure 4.1 for a detailed view of typical input data). We also want the cadastre to register onto image *edges* as much as possible. More precisely, we want to locally modify the cadastre graph so that its spatial structure is preserved (that is, the face topology of its planar representation) while incorporating the geometric details of corresponding salient edges in the image. To the best of the author's knowledge, this specific problem has not been previously dealt with.

We approach this as a graph matching problem, like Hivernat and Descombes [HD98] but with suitable modifications: in the context of graph matching, two graphs representing the same physical reality are given, and the goal is to match the edges and nodes that correspond to the same part of that reality [GB03, Wal98]. In our case, we use a multi-scale segmentation of the image—obtained using the methods described in chapter 3—to obtain a graph representation, whose edges separate homogeneous image regions and are weighted according to the dissimilitude between these regions. In doing this, we assume that all significant image edges—at least, all image edges that should be matched with the given cadastre edges—exist in the multi-scale segmentation from which we obtain the graph representation; by making this segmentation sufficiently fine, we can be fairly confident that this is the case. A similar approach is used by [KKC05].

I propose two approaches for solving this matching problem.

The first approach, which was sketched in [TS04a], uses simulated annealing to find the best match between edges in the cadastre graph and segmentation edges in the image. This is described in section 4.3. This *edge-based* method preserves well the face structure of the cadastre graph, but does not always follow salient image edges since auxiliary straight edges must be added to the solution.

The second approach, which was presented in [TSPD04], finds a near-optimal match between



Figure 4.1: Typical input data (close-up). Terrain image, with cadastre edges superimposed. Note that some image edges do not have corresponding cadastre edges, and vice versa, and that those that do have slightly different traces.

faces in the cadastre graph and homogeneous regions in the image. We define an initial mapping, and constraints on mappings which depend on the degree of saliency of edges between faces, and then obtain a near-optimal mapping using an optimization algorithm. This is described in section 4.4. I propose two variants, one using probabilistic relaxation for the optimization, and another using simulated annealing. In most graph matching problems [Wal98, TS04b, HD98, GB03] two graphs representing the same reality are given, and the goal is to match the edges and nodes that correspond to the same part of that reality; this second method can then be viewed as a loose graph matching procedure in which graph faces, instead of edges and nodes, are the primary object being manipulated. This *region-based* method always follows salient image edges, but does not preserve the face structure of the cadastre graph so well.

The best algorithm, in terms of quantitative evaluation, is the region-based method, using probabilistic relaxation for the optimization. Tests show a 32.2% reduction in the average distance between the cadastre and a ground truth (up to 43.7% when using a distorted cadastre as input). Improvements are slightly lower with the edge-based method, and even lower with the region-based method with simulated annealing. More important than this quantitative improvement is the fact that the registered portions of the resulting graph follow the edges of the image—exactly, in the case of the region-based method—which will allow us to perform statistical analysis of whole regions, without noise from adjacent regions. Although I apply this method to the registration of a cadastre graph to an aerial image, it can also be used in other contexts. In addition, registration quality measures presented in sections 4.3.8 and 4.4.4 can be used to determine which cadastre edges do not actually exist in the image, and may need to be removed for further processing.

In this chapter I present in detail the edge-based method in section 4.3, and the region-based method in section 4.4, together with a complete quantitative evaluation in section 4.5, and a comparison and conclusion in section 4.7. Section 4.2 describes the procedure used to obtain the initial graphs from the image and cadastre data; this procedure is common to both algorithms. Figure 4.2 shows as a flowchart the part of the complete process described in this chapter.

4.2 Image over-segmentation and input graphs

Both registration methods presented here are set as graph matching algorithms. That is, they try to match edges, nodes, or faces from a graph representing the image with edges, nodes, or faces from a graph representing the cadastre. For each match, the geometry of the image element is then transferred to the corresponding cadastre element. We need therefore to start with graph representations for both the cadastre and the image.

The cadastre input is already a graph, a planar one, with graph edges separating adjacent plots, graph *faces* corresponding to plots, and graph nodes where three or more adjacent plots meet. We simplify this graph using mathematical morphology to remove very thin faces (around 1 pixel wide) that might confuse the algorithms.

We also create a graph (the *segmentation graph*) representing the salient edges in the image and their geometry, as described later in this section. With these two graphs we can formalize this as a graph matching problem: We match the cadastre graph to the segmentation graph, and then transpose the geometrical information from the edges of the segmentation graph to the corresponding cadastre edges.

This graph matching is asymmetric: the segmentation graph contains many more *terrain edges* than the cadastre graph, but they are shorter. Typical values are 1000 terrain edges for 50 cadastre edges. For the edge-based algorithm, we may assume that each terrain edge matches

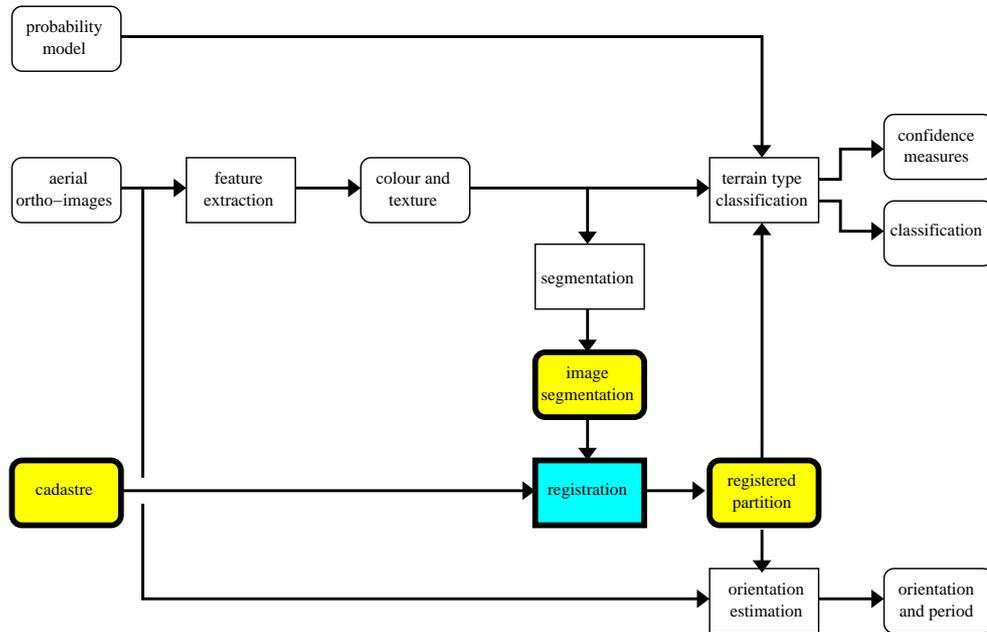


Figure 4.2: Flowchart depicting the processing steps (in blue rectangles) and intermediate results (in yellow rounded boxes) involved in the registration of cadastre to images, in the context of the complete system.

at most one cadastre edge, and that each cadastre edge may match several terrain edges. Most terrain edges will not have a match. Note that while most graph-matching algorithms can handle a certain amount of unmatched edges, they are usually not designed with such a degree of asymmetry in mind. Similarly, for the region-based algorithm, we may assume that each terrain face matches at most one cadastre face, and that each cadastre face may match several terrain faces; however, in this case all terrain faces will have a match.

The cadastre and segmentation graphs have a geometrical component: their edges and vertices are located in the image and, ultimately, on the ground. We will therefore use geometrical graphs (section 3.1.2) to describe them.

The cadastre graph is a geometrical graph,

$$C = (V_c, E_c, \text{trc}, \mathbb{Z}^2). \quad (4.1)$$

To obtain the segmentation graph, T , we need to extract the topology and geometry of the salient edges in an image and a measure of their saliency. A simple method would be to weight each segmentation edge in a watershed segmentation with the module of the image gradient in that edge. However, this measure of edge saliency would be local and single-scale, and therefore not satisfactory, since meaningful structures may appear at different scales of analysis [Mar82].

Several authors have proposed *multi-scale* algorithms to solve this. We use Guigues' *scale-sets* algorithm [GLMC03b] because it makes the segmentation criterion and the scale parameter λ explicit: For each λ this algorithm gives a *partition* of the image $p(\lambda)$; the set of values of λ for which a region exists in $p(\lambda)$ turns out to be an interval, $[\lambda_{\min}, \lambda_{\max}]$.

We *flatten* these results to obtain the segmentation graph using the method described in sections 3.1.4 and 3.1.4: We build a geometrical weighted graph, the *terrain graph* $T =$

$(V_t, E_t, \text{trc}, \mathbb{Z}^2, w)$ (w is the weight function), whose edges follow the boundaries of the regions given by $p(\cdot)$. We find for each edge e the highest λ_{\min} of all the regions whose boundary contains e , $\lambda_{\min}(e)$ (see figure 4.12(3)). To improve processing time, we discard edges with small λ_{\min} (in this implementation, the 30% of edges with smallest values of λ_{\min}). We sort the rest as $\lambda_{\min}(e_{s_0}) > \lambda_{\min}(e_{s_1}) > \dots > \lambda_{\min}(e_{s_{N-1}})$ and attribute each edge with the *apparition weight* w ,

$$w(e_{s_i}) = e^{-\alpha i/N}. \quad (4.2)$$

The apparition weight is a multi-scale, non-local, saliency measure. Figure 4.3 shows an image and its segmentation graph.

Both variants of the region-based registration algorithm work with the duals of the terrain and cadastre graphs T and C :

From the weighted terrain graph T we obtain its weighted dual graph $\bar{T} = (E_{\bar{T}}, V_{\bar{T}}, w)$, with nodes $V_{\bar{T}}$ and edges $E_{\bar{T}} \subset V_{\bar{T}} \times V_{\bar{T}}$ (see Figure 4.4). The weight of an edge in $E_{\bar{T}}$ joining two vertices $v_1, v_2 \in V_{\bar{T}}$ is that of the edge in E_T that separates the two graph faces corresponding to v_1 and v_2 .

Analogously, from the cadastre graph C (after simplification by mathematical morphology operations), we obtain its dual graph $\bar{C} = (E_{\bar{C}}, V_{\bar{C}})$, whose nodes represent cadastre regions.

4.3 Edge-based registration algorithm

For the edge-based registration algorithm, the desired result is a match from each edge in the cadastre graph C to a chain of edges in the segmentation graph T , that is, to an ordered sequence of edges, such that each edge has a common endpoint with the next edge in the sequence. However, internally the algorithm manipulates a different kind of solution: A match from each segmentation edge to a cadastre graph edge.

In section 4.3.1 we formalize this latter representation of a *solution*. In section 4.3.2 we study how to convert from this to the first representation of a solution, that is, from a mapping from each segmentation edge to a cadastre edge, to a mapping from each cadastre edge to a chain of terrain edges. We can evaluate the quality or *fitness* of such a mapping (section 4.3.3) and therefore use an optimization algorithm to find the optimal one. To avoid the combinatorial explosion associated with this kind of problem, we use simulated annealing [KGV83] to find a near-optimal solution. Because of that, we also need a way of exploring the solution space (section 4.3.4), and an initial solution (section 4.3.5).

However, I have found that, in order to register the cadastre onto image edges and at the same time preserve the spatial structure of the cadastre graph, the areas near cadastre nodes and the rest of the problem have to be processed separately: We first apply the simulated annealing optimization to the rest of the problem, and then an ad-hoc method (section 4.3.7) for the areas near cadastre nodes.

Throughout the description of the edge-based registration algorithm, $\alpha_1, \dots, \alpha_6$, are configurable parameters in $[0, +\infty)$, N_r is a configurable parameter in \mathbb{N} and p_{\perp} is a configurable parameter in $[0, 1]$; we chose their values empirically, although tests with several parameter sets show that the algorithm is not very sensitive to their values.

4.3.1 Solution representation

We represent a *solution* to a registration problem as a relation between cadastre edges and terrain edges. In what we call the *backward* representation, we label each terrain edge with

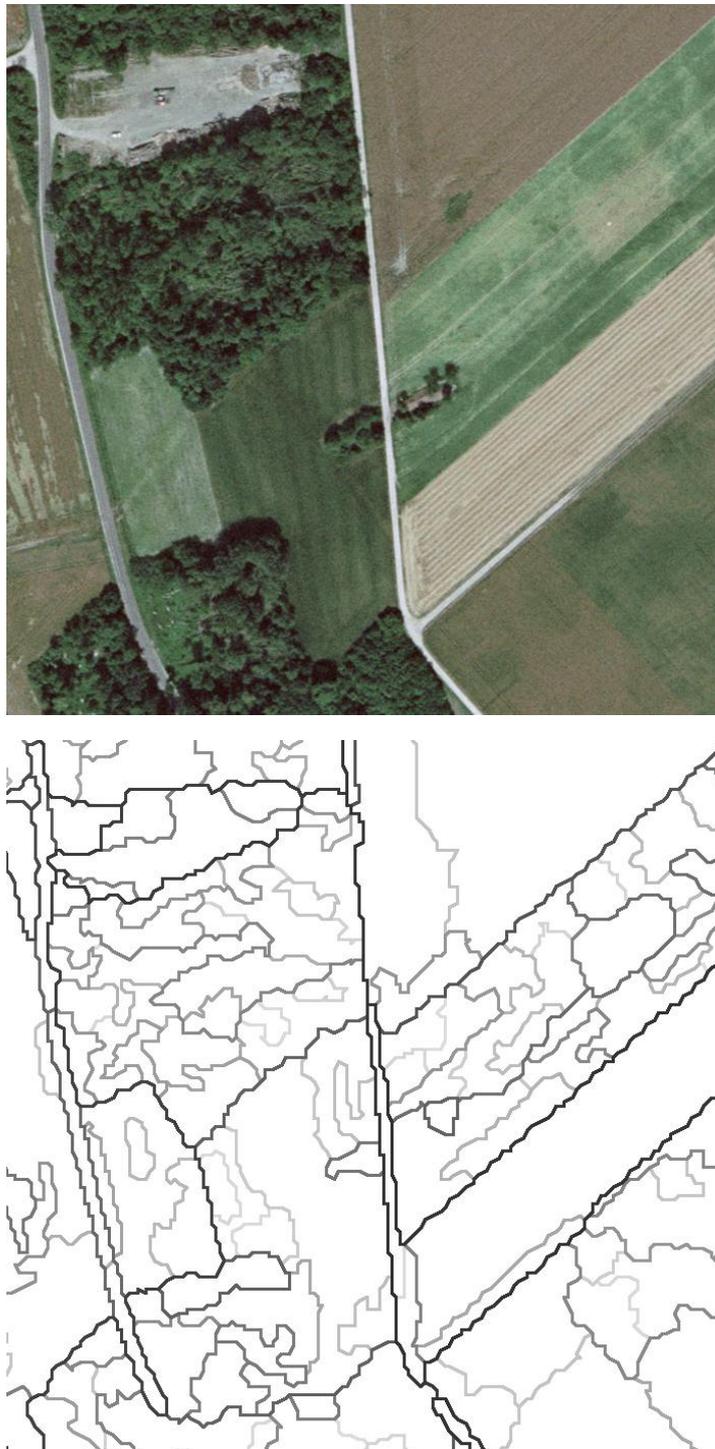


Figure 4.3: Example image extracted from the Saint-Léger test site (top), and its segmentation graph T (bottom; graph edges with higher weights are shown darker).

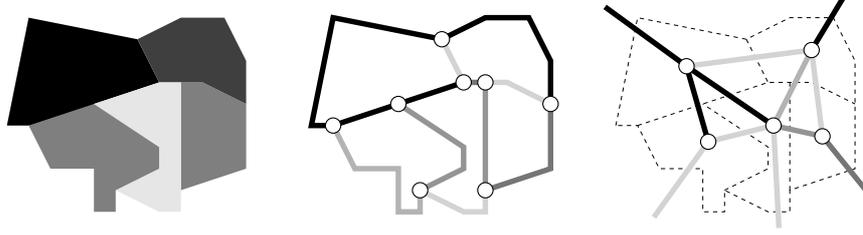


Figure 4.4: An image, its segmentation graph T , and its dual segmentation graph \tilde{T} . Darker edges have higher weights.

its corresponding cadastre edge, or with \perp to indicate that it is unmatched (see [HD98] for a similar representation, although they use it for Markov modelling and in a situation where the \perp -label is rarely used). We only allow labeling a terrain edge e_i with cadastre edges that are close to it, that is, labeling e_i with an element of the set of neighbours $N(e_i)$,

$$N(e_i) = \left\{ e \in E_c : \min_{\substack{z \in \text{trc } e_i \\ z' \in \text{trc } e}} \|z - z'\| < \epsilon \right\}, \quad (4.3)$$

and do not allow labelling with \perp for terrain edges with only one near cadastre edge: this disables the optimization process for these edges, leaving only the shortest-path search described in section 4.3.2. A *backward solution* is then a mapping S from each terrain edge e_i to $N_x(e_i)$,

$$S : e_i \mapsto S(e_i) \in N_x(e_i) \quad (4.4)$$

where

$$N_x(e_i) = \begin{cases} N(e_i) \cup \{\perp\} & \text{if } \text{card } N(e_i) \neq 1, \\ N(e_i) & \text{if } \text{card } N(e_i) = 1. \end{cases} \quad (4.5)$$

4.3.2 From backward to forward representation

Using the backward representation it is easy to find solutions similar to a given one (section 4.3.4) and to create initial solutions (section 4.3.5). But this representation is not a single chain of terrain edges for each cadastre edge—what we would like to have as final output from this algorithm—but rather a set of edges which probably do not form a chain. It is also difficult to calculate its fitness. Therefore, we need to convert from the backward representation to a *forward* representation, in which we map each cadastre edge to a—possibly empty—oriented chain of terrain edges, as follows:

For each cadastre edge $e_k \in E_c$, to which the terrain edges $m_k = \{e_i \in E_t : S(e_i) = e_k\}$ are mapped, we create a graph T_k which contains

- the terrain edges m_k and the nodes at their ends,
- the cadastre nodes n_0 and n_1 at the ends of e_k ,
- N_0 , the set of terrain nodes close to n_0 , and
- N_1 , the set of terrain nodes which are close to n_1 .

We attribute each edge $e \in T_k$ with a weight w_p (the *path matching weight*) that depends on its length $\ell(e)$, its apparition weight w_e , and an average “distance” $\bar{d}(e, e_k)$ between it and e_k , as

$$w_p = \alpha_1 \ell(e) (1 - w_e) + \alpha_2 \ell(e) \bar{d}(e, e_k), \quad (4.6)$$

with the average non-symmetric “distance” \bar{d} between a terrain edge $e_t \in E_t$ and a cadastre edge $e_c \in E_c$ defined as

$$\bar{d}(e_t, e_c) := \frac{1}{|\text{trc}(e_t)|} \sum_{z_t \in \text{trc}(e_t)} \min_{z_c \in \text{trc}(e_c)} \|z_t - z_c\|. \quad (4.7)$$

T_k contains several connected components (some of them composed of a single graph node). We create a graph T'_k which contains T_k and additional straight edges (which we call “connecting edges”), which join every two connected components in T_k by their closest nodes. We attribute each of these additional straight edges e with a path matching weight w_p that depends on its length $\ell(e)$ as

$$w_p = \alpha_3 \ell(e). \quad (4.8)$$

Then, using the path matching weight as the “length” of an edge, we find on T'_k the “shortest” path s_k from any node in $N_0 \cup \{n_0\}$ to any node in $N_1 \cup \{n_1\}$ using Dijkstra’s algorithm [Dij59, Jun99]. The forward solution for S is then the mapping of each cadastre edge e_k to the shortest path s_k obtained with this process. This is shown in figure 4.5.

Thanks to the connecting edges, there is always such a path; however, the path matching weight for connecting edges is high, to discourage their use. Note that since we allow these shortest paths to start and end not only on the endpoints of cadastre edges (n_0, n_1) but also on nearby terrain nodes (N_0, N_1) , this will, as intended, not register the cadastre in the areas near cadastre nodes.

4.3.3 Calculating a solution’s fitness

The length of the shortest path calculated as in section 4.3.2 is an indication of the quality of a registered cadastre edge. To get a suitable fitness measure for the whole solution s_k , we sum the shortest path lengths for all edges in s_k and we add three penalties:

Many terrain edges (which are mapped to actual cadastre edges, not to \perp) are not actually in the shortest paths for any cadastre edges. To drive down the complexity of solutions, we impose a penalty on the number of such unused terrain edges.

Also, the shortest-path weighting method evaluates registrations for each cadastre edge independently. As a consequence, a cadastre vertex usually corresponds to several terrain vertices. In some cases even, chains cross each other or the spatial distribution is otherwise not preserved. To push against these situations, we add two penalties. One, for each cadastre vertex, depends on the number of matching terrain vertices—a penalty applies if there is more than one terrain vertex for each cadastre vertex. The other, for each terrain vertex not corresponding to a cadastre vertex, depends on the number of incident terrain edges selected for shortest paths—a penalty applies if there are more than two incident terrain edges.

In all, the cost of a solution is

$$F = \sum_{e_k \in E_c} l_k + \alpha_4 \cdot n_u + \alpha_5 \sum_{v_j \in V_c} (\text{card } H_j - 1) + \alpha_6 \sum_{v_k \in V_t} (\text{card } \text{inc } v_k - 2) \quad (4.9)$$

where l_k is the length of the shortest path calculated for the cadastre graph e_k (the sum of the path matching weights of its edges), n_u is the number of unused terrain edges, H_j is the set of terrain vertices corresponding to the cadastre vertex v_j , and $\text{inc } v_k$ is the set of terrain edges incident to the terrain vertex v_k . Lower values of F are better.

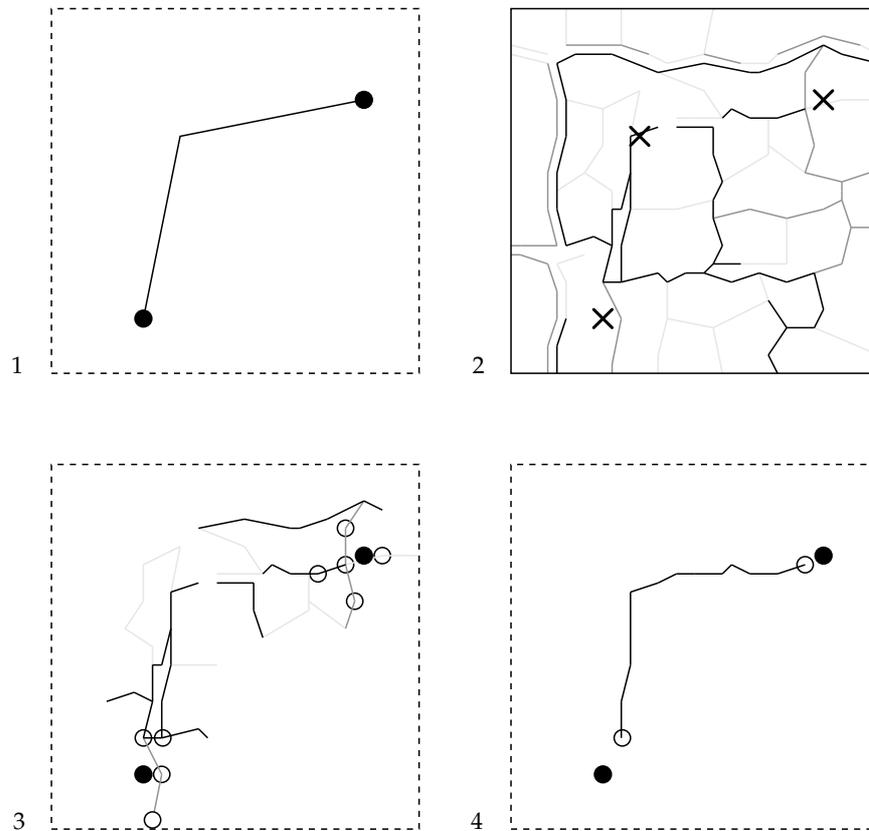


Figure 4.5: From backward to forward solution, for a simplified example. 1: a cadastre edge e_k . 2: terrain graph E_t (edge weight is shown by line darkness, “x”s show the vertices of e_k ’s trace). 3: subgraph T_k , with N_0 and N_1 (hollow dots), and n_0 and n_1 (solid dots). 4: shortest path, including connecting edges. Note that this process is run at each iteration of the annealing algorithm —different backward solutions may give different T_k s and possibly better shortest paths.

4.3.4 Finding a similar solution

In order to solve a problem using descent methods such as steepest-descent, simulated annealing or genetic algorithms, we need, in addition to a way of evaluating the quality of a solution, a way to find solutions which are “close” to a given solution. This is the *mutation* or *neighbourhood* operator (for genetic algorithms, we also need a *crossover* operator).

Given an evaluation function f , a neighbourhood operator n which returns a set of neighbouring solutions, and an initial solution S_0 , we can minimize f by steepest-descent as

1. $S \leftarrow S_0$;
2. $S_n \leftarrow \operatorname{argmin}_{c \in n(S)} f(c)$;
3. if $f(S_n) < f(S)$ then $S \leftarrow S_n$, go to step 2;
4. else S is a (local) minimum of f , stop.

Simulated annealing and genetic algorithms are variations on the steepest-descent algorithm

with improvements to avoid getting stuck in local minima and (for genetic algorithms) to explore the solution space in parallel.

In this registration algorithm, we can obtain one neighbour solution S' from S as follows: A number N_r of terrain edges are selected —however, edges which have only one or no cadastre edges nearby are never selected (eqs. (4.3)–(4.5))— and a new random label is given to each selected edge. The new label is \perp with probability p_{\perp} , and the remaining labels are equiprobable:

$$\begin{aligned} p(S'(e_i) = \perp) &= p_{\perp} \\ p(S'(e_i) = e_k) &= \frac{1 - p_{\perp}}{\text{card } N(e_i)}, \text{ for all } e_k \in N(e_i). \end{aligned} \quad (4.10)$$

4.3.5 Finding an initial solution

We also need to find an initial solution S_0 . We can obtain one by labelling each terrain edge with the cadastre edge that is closest to them in the sense of the average distance \bar{d} (equation (4.7))

$$S_0(e_i) = \underset{e_k \in N(e_i)}{\text{argmin}} \bar{d}(e_i, e_k). \quad (4.11)$$

Other options include starting with the null solution $S_0(e_i) = \perp$, or with a random solution.

4.3.6 Optimization

Once we have a representation (sections 4.3.1 and 4.3.2) which allows us to find solutions neighbouring a given one (section 4.3.4), to find an initial solution (section 4.3.5), and to calculate a solution's fitness (section 4.3.3), we can apply standard optimization algorithms such as steepest descent, simulated annealing [KGV83] or genetic algorithms to find the (possibly global) optimum. For this registration algorithm, simulated annealing is used.

4.3.7 Registration near cadastre nodes

The optimization process alone cannot at the same time register the cadastre onto salient image edges *and* preserve the spatial structure of the cadastre graph: To preserve the spatial structure in areas of the image with a low density of image edges we need to allow the use of *connecting edges*, which are straight and do not follow salient image edges. In these same low density areas, because image edges are sparse, strictly following image edges may modify the spatial structure of the cadastre, splitting or deleting faces in the cadastre graph, for example. Additionally, adding connecting edges may make the graph non-planar. Some of the penalties included in the fitness function (section 4.3.3) mitigate the problem, but do not eliminate it completely. The underlying reason is that it is not really possible to register cadastre edges which do not have a corresponding image edge.

These problems tend to occur at the areas near nodes of the cadastre graph. For this reason, the optimization procedure does not attempt to register the cadastre there (section 4.3.2). After the optimization phase, we process these areas as follows (see figure 4.6) to complete the registration:

For a cadastre node n , let $E = \{e_1, e_2, \dots, e_k\}$ be the set of cadastre edges incident to it. Each cadastre edge e_i is registered to a chain of terrain edges which can end at n , or at a terrain node close to n . Let's call n_i the endpoint for e_i which is close to or equal to n .

Let M be a subgraph of the terrain graph containing only edges and nodes close to n . We can find paths through M which connect some endpoints e_i, e_j together. Let's split E into subsets J_1, J_2, \dots, J_m grouping those edges whose endpoints can be connected through M —that is, there is a path through M between the endpoints of each two edges in one J_i , but not between those of two edges in different J -sets. We then connect these endpoints within each J_i using these terrain edges in M . Finally, we join the sets J_1, \dots, J_m , by adding appropriate straight edges to the registered cadastre graph.

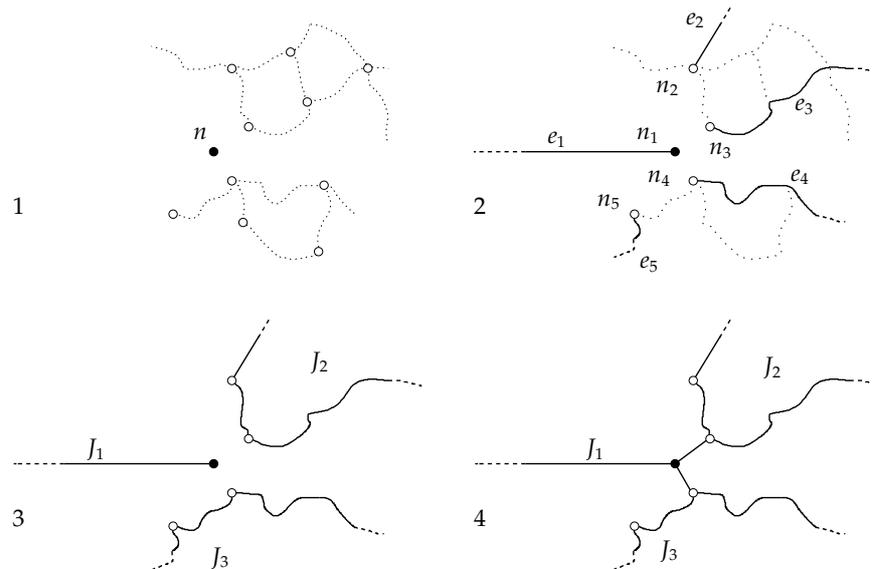


Figure 4.6: Registration near cadastre node n . 1: terrain subgraph M . 2: incident edges e_1, \dots, e_5 and endpoints n_1, \dots, n_5 ($n_1 = n$). 3: connection within subsets $J_1 = \{e_1\}$, $J_2 = \{e_2, e_3\}$, and $J_3 = \{e_4, e_5\}$. 4: connection between subsets J_1, J_2 , and J_3 .

4.3.8 Registration quality measure

For each registered cadastre edge e_k we calculate a quality measure, the *registration ratio*. The shortest path for e_k, s_k , is a chain of terrain edges and connecting edges. The registration ratio is the length of the *terrain* edges in s_k (that is, not counting connecting edges) divided by the total length of s_k (we use the Euclidean length, not the “path matching weight”-induced length of section 4.3.2). For example, a cadastre edge which is fully registered to terrain edges has a registration ratio of 1, while a cadastre edge which corresponds to a single connecting edge from end to end has a registration ratio of 0.

This appears to be a good indicator of whether or not the cadastre edge follows a true terrain limit. Since path matching weights are higher for connecting edges than for terrain edges, cadastre edges will tend to register onto terrain edges unless there is really no terrain edge strong enough to follow. In section 4.5.1 we further explore this. In subsequent processing, this can be used to decide whether to merge adjacent regions: if the cadastre edge between them has a low registration ratio, they probably can be merged since they have the same texture and colour.

4.4 Region-based registration algorithm

In contrast to the edge-based registration algorithm, which attempts to match corresponding edges in the terrain and the cadastre graph, the region-based registration algorithm works by trying to match nodes in the dual terrain graph $\bar{T} = (E_{\bar{T}}, V_{\bar{T}}, w)$ (corresponding to homogeneous terrain regions) to nodes in the dual cadastre graph $\bar{C} = (E_{\bar{C}}, V_{\bar{C}})$ (corresponding to cadastre plots).

Since T will be obtained by *over*-segmenting the image, we can assume that each node in \bar{T} belongs to only one node in \bar{C} , and formalize a *solution* as a mapping

$$s : V_{\bar{T}} \rightarrow V_{\bar{C}}, \quad (4.12)$$

and use different optimization algorithms to find an optimal or near-optimal solution. We propose two variants: In section 4.4.1 using probabilistic relaxation for the optimization process, and in section 4.4.2 using simulated annealing.

Throughout the description of the region-based registration algorithm, $k_0, \dots, k_7, \beta, \pi_0, \pi_1, \pi_2$ and π_r are configurable parameters in $[0, +\infty)$, and N_r is a configurable parameter in \mathbb{N} ; we chose their values empirically, although tests with several parameter sets show that the algorithm is not very sensitive to their values. $N(a)$ is the set of nodes at distance 1 to the graph node a —that is, reachable from a through a single edge—, and w_{ij} is the weight of the edge in \bar{T} between nodes i and j .

4.4.1 Registration by probabilistic relaxation

Probabilistic relaxation or relaxation labelling is an optimization method first proposed by Rosenfeld *et al.* [RHZ76]. Following Fu and Yan's notation [FY97], we define a set of objects to be labelled (the terrain nodes $V_{\bar{T}}$ in our case), a set of possible labels (the cadastre nodes $V_{\bar{C}}$), initial probabilities $p^{(0)}$, an influence function d and a compatibility function c .

The influence function d_{ij} , with $i, j \in V_{\bar{T}}$, measures the relative influence a face j has over the face i . For non-adjacent faces, it is 0. For adjacent faces, we make it dependent on the area a_j of the face j , and on the length ℓ_{ij} of the edge (i, j) . Specifically,

$$d_{ij} = \kappa_i \left(k_0 + k_1 \ell_{ij}^{k_2} + k_3 a_j^{k_4} \right), \quad (4.13)$$

where κ_i is chosen to fulfill the condition $\sum_j d_{ij} = 1$.

The compatibility function $c_{ij}(\lambda_i, \lambda_j)$, for $i, j \in V_{\bar{T}}$ so that $d_{ij} \neq 0$, and $\lambda_i, \lambda_j \in V_{\bar{C}}$, measures the compatibility of the labeling $i \mapsto \lambda_i$ and $j \mapsto \lambda_j$, with $0 \leq c \leq 1$ and $\sum_{\lambda_i} c_{ij}(\lambda_i, \lambda_j) = 1$. We have chosen

$$c_{ij}(\lambda_i, \lambda_j) = \begin{cases} (1 - w_{ij})^\beta & \text{if } \lambda_i = \lambda_j, \\ \frac{1 - (1 - w_{ij})^\beta}{\text{card } V_{\bar{C}}} & \text{if } \lambda_i \neq \lambda_j, \end{cases} \quad (4.14)$$

that is, the more salient the edge between two faces, the better it is that these faces be labeled differently.

The initial probability function $p_i^{(0)}(\lambda)$, with $i \in V_{\bar{T}}$ and $\lambda \in V_{\bar{C}}$, gives the initial state of the system (a sort of fuzzy initial solution). It should reflect the *a priori* probability that i is mapped to λ_i . Note that probabilistic relaxation works correctly even if these are not real *a priori* probabilities obtained through statistical analysis. I chose the following initial probabilities: For each node i in \bar{T} , we find the centre of gravity b_i of its corresponding face in T . We then find the face in C whose centre of gravity is closest to b_i , and its corresponding node in \bar{C} , c_i .

We define

$$p_i^{(0)}(\lambda) = \begin{cases} \alpha_i \pi_0 & \text{if } \lambda = c_i, \\ \alpha_i \pi_1 & \text{if } \lambda \in N(c_i), \\ \alpha_i \pi_2 & \text{if } \lambda \in \cup_{b \in N(c_i)} N(b) \setminus \{c_i\}, \\ \alpha_i \pi_r & \text{otherwise} \end{cases} \quad (4.15)$$

with α_i such that $\sum_{\lambda \in V_C} p_i^{(0)}(\lambda) = 1$. Recall that $N(a)$ is the set of neighbours of a .

An iterative process is then run, repeatedly updating the current probabilities with an update function defined in [FY97], $p^{(t+1)} = F(p^{(t)}, c, d)$, until convergence (or for a maximum number of iterations):

$$p_i^{(k+1)}(\lambda) = p_i^{(k)}(\lambda) \cdot \left(1 + \frac{a_i^{(k)}(\lambda)}{q_i^{(k)}} \right), \quad (4.16)$$

where

$$a_i^{(k)}(\lambda) = s_i^{(k)}(\lambda) - \bar{s}_i^{(k)}, \quad (4.17)$$

$$\bar{s}_i^{(k)} = \sum_{\lambda \in V_C} p_i^{(k)}(\lambda) s_i^{(k)}(\lambda), \quad (4.18)$$

$$s_i^{(k)}(\lambda) = \sum_{j \in V_T} d_{ij} \sum_{\eta \in V_C} c_{ij}(\lambda, \eta) p_j^{(k)}(\eta), \quad (4.19)$$

with different authors giving different choices for $q_i^{(k)}$,

$$q_i^{(k)} = \begin{cases} 1 + \bar{s}_i^{(k)} & \text{(Rosenfeld et al. [RHZ76]),} \\ \bar{s}_i^{(k)} & \text{(Zucker et al. [ZKH78]),} \\ 1 & \text{(Chen and Luh [CL94, CL95]).} \end{cases} \quad (4.20)$$

After convergence to $p^{(\infty)}$, the solution is the labelling

$$s^{(\infty)}(i) = \operatorname{argmax}_{\lambda \in V_C} p_i^{(\infty)}(\lambda). \quad (4.21)$$

In this implementation, we treat values of $p_i^{(t)}(\lambda)$ below a small threshold as 0, which greatly reduces processing time without modifying the results.

Furthermore, the naive implementation of probabilistic relaxation has a time complexity of $O(|V_T|^2 \cdot |V_C|^2)$ per iteration. In order to reduce processing time, we ignore non-adjacent regions for the compatibility and influence functions.

4.4.2 Registration by simulated annealing

Simulated annealing is a well-known heuristic optimization algorithm that can find near-optimal solutions for problems where steepest-descent-based optimizers tend to get stuck at local optima. To use it we need an initial solution $s^{(0)}$, a way of evaluating the *energy* of a solution, and a way of obtaining solutions similar to a given solution. Starting from $s^{(0)}$ the algorithm iteratively modifies it to obtain a near-optimal solution $s^{(\infty)}$.

To obtain $s^{(0)}$ we use $p^{(0)}$ defined in equation (4.15): $s^{(0)}(i)$ is chosen randomly following the probability distribution given by $p_i^{(0)}$. To obtain, from s , a similar solution s' , we replace a certain number N_r of its labellings; each new labelling $s'(i)$ is chosen randomly following $p_i^{(0)}$.

To evaluate the quality of a solution s , we do the following. For each $i \in V_{\bar{T}}$, we find $e_s(i)$

$$e_s(i) = a_i^{k_5} \cdot \sum_{j \in N(i)} e'_s(i, j) + a_i \delta_{\text{exterior}}(i) k_7, \quad (4.22)$$

with

$$e'_s(i, j) = \begin{cases} d_{ij} \cdot (1 - (1 - w_{ij})^\beta) & \text{if } s(i) = s(j), \\ d_{ij} \cdot (1 - w_{ij})^\beta & \text{if } s(i) \neq s(j), \end{cases} \quad (4.23)$$

where a_i is the area of the face in T corresponding to the node i in \bar{T} , d_{ij} is as defined in equation (4.13), and $\delta_{\text{exterior}}(i)$ is 1 if the node i is mapped to a special node in $V_{\bar{C}}$ corresponding to the *outside* of the cadastre, and 0 otherwise (this is a penalty to minimize such mappings). The energy of a solution s , $E(s)$, is the sum of the e_s of its nodes, plus a penalty for each node none of whose neighbours has the same label as itself —this penalises isolated nodes. If there are d such nodes,

$$E(s) = k_6 d + \sum_{i \in V_{\bar{T}}} e_s(i). \quad (4.24)$$

The lower the energy, the better the solution.

4.4.3 Post-processing

The output $s^{(\infty)}$ of any of these two variants is then processed as follows:

First, each isolated node n —a node which has no neighbour labelled like n —, is merged to the neighbour node $n' \in N(n)$ for which the edge $(n, n') \in E_{\bar{T}}$ has the lowest apparition weight w . This is to avoid very small isolated regions.

Second, following the mapping defined by $s^{(\infty)}$, the connected faces in T which have the same label are merged. The resulting primal graph R is taken as the registration of the cadastre graph C onto the image.

Finally, cadastre faces which lie outside the cadastre boundary are removed, following an application-specific requirement.

4.4.4 Registration quality measure

Each edge e in R is the concatenation of a certain set of edges $P(e) \subset T$. We compute $m(e)$, the average of the apparition weights of the edges in $P(e)$, weighted by their lengths:

$$m(e) = \frac{\sum_{t \in P(e)} w_t \ell(t)}{\sum_{t \in P(e)} \ell(t)}. \quad (4.25)$$

This gives a measure of how strong the image edge corresponding to a registered cadastre edge is. As in section 4.3.8, low values indicate that there is probably no corresponding image edge, and high values indicate that there is indeed an image edge at that position. In section 4.5.1 we further explore the validity of this indicator.

4.5 Experiments and evaluation

I have performed several experiments to investigate the behaviour and performance of these registration algorithms.

Although presented later in the chapter, I first systematically scanned the parameter space in order to determine the optimal parameter set for each of the algorithms and variants. This is described in section 4.5.2.

With these parameter sets the main experiment was run; in this experiment I ran the algorithms in the conditions that will be used in the final, production-grade, processing chain: parameters set for fast execution, images downsampled at 2 m per pixel, and realistic cadastre data. This is described in section 4.5.1.

Finally, other aspects of the algorithms are tested. In section 4.5.3 we explore the effect of input image resolution, convergence speed, and input cadastre quality on the algorithms' performance.

Because only for the Saint Léger test site we have reliable cadastre information, the tests described in this chapter have been run only for this site and not for the Toulouse site.

Figure 4.7 shows a flowchart of steps involved in testing and evaluating the registration algorithms.

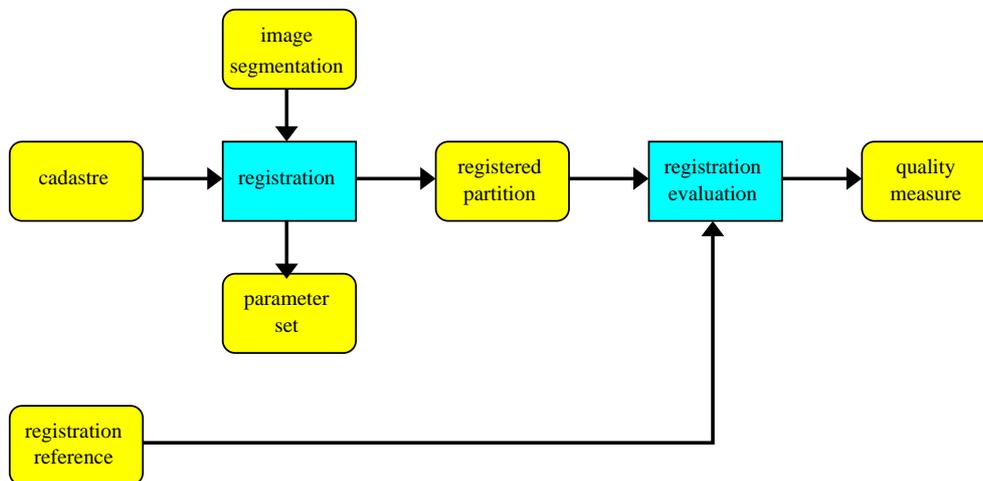


Figure 4.7: Flowchart depicting the processing steps (in blue rectangles) and intermediate results (in yellow rounded boxes) for the evaluation of the registration algorithms.

4.5.1 Main experiment

I have run these algorithms on a test site of 4 km², on which I defined a ground truth, containing the terrain edges in the image (field boundaries and other clear edges). The segmentation used to obtain the terrain graph T was computed with Guigues' scale-sets algorithm [GLMC03b] using the red, green, and blue colour components of an image, downsampled to 2 m per pixel to improve speed. In chapter 3 and section 3.4 I discuss how using texture information and region shape for segmenting may affect the quality of the terrain graph and the registration.

For the edge-based algorithm, I let the optimizer run for 2500 iterations with a fast cooling schedule, taking around 10 minutes to process the whole test site. For the region-based algorithm, I let the optimizers, in both variants, run for about 4 minutes. The best parameter sets were chosen for this experiment; section 4.5.2 describes how they were determined.

To measure the effect of the registration process, the ground truth and the registered cadastre graph were drawn on a grid, and we computed the distances between each pixel of the

registered cadastre graph and the ground truth. We did the same for the unregistered cadastre graph. Let $I_c \subset \mathbb{Z}^2$ be the set of pixels of the registered or the unregistered cadastre graph, and $I_g \subset \mathbb{Z}^2$ that of the ground truth. The distance between $i_c \in I_c$ and the ground truth is

$$d(i_c, I_g) = \min_{i_g \in I_g} \|i_c - i_g\|, \quad (4.26)$$

and the average distance between the cadastre graph and the ground truth is

$$d(I_c, I_g) = \frac{1}{\text{card } I_c} \sum_{i_c \in I_c} d(i_c, I_g). \quad (4.27)$$

A histogram of these distances $d(i_c, I_g)$ for the edge-based algorithm is shown in figure 4.8. One for the region-based algorithms is shown in figure 4.9.

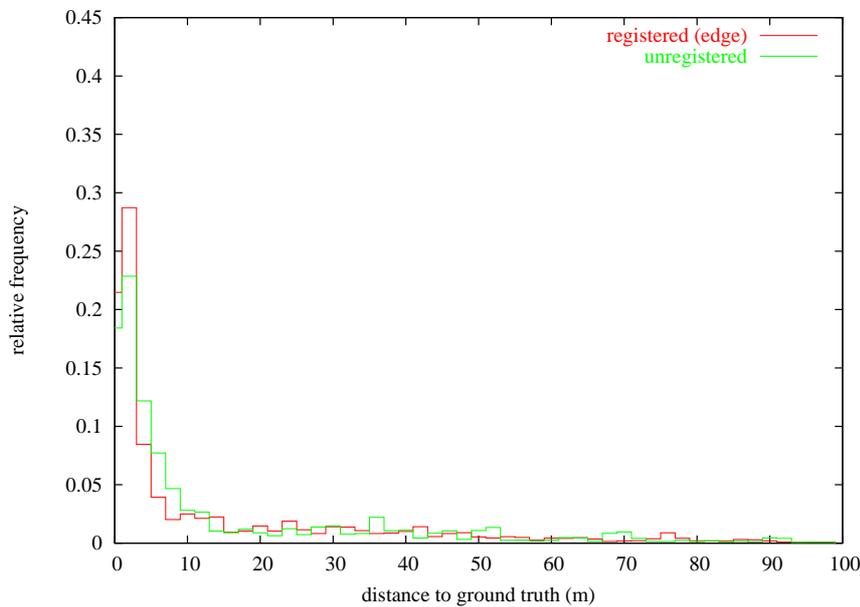


Figure 4.8: Histogram of distances from cadastre graph pixels to ground truth, in metres. Red line: cadastre registered by the edge-based algorithm; green line: unregistered cadastre.

Since some cadastre edges have no matching ground truth edge (because the cadastre edge does not correspond to any edge in the image), the histogram tails off to large distances. Therefore, in order to obtain a meaningful aggregate quality measure, we excluded the cadastre edges for which the average distance to the ground truth was larger than 6 m, or which had a pixel that was farther than 20 m away from the ground truth, and we calculated the average distance to the ground truth of the pixels in the remaining cadastre edges. These thresholds were manually chosen after inspecting the input data and these histograms, and are meant to exclude *non-matchable* edges (cadastre edges which do not have a corresponding image edge). Since they are applied to both the unregistered and the registered cadastre, their actual value is of little importance for finding how much the registration reduces the average distance in relative terms.

A histogram of the distances from the remaining cadastre edges to the ground truth is shown in figure 4.10 for the edge-based algorithm, and in figure 4.11 for the region-based algorithm. A summary of the results is shown in table 4.1. In this table, the *ground truth length* includes

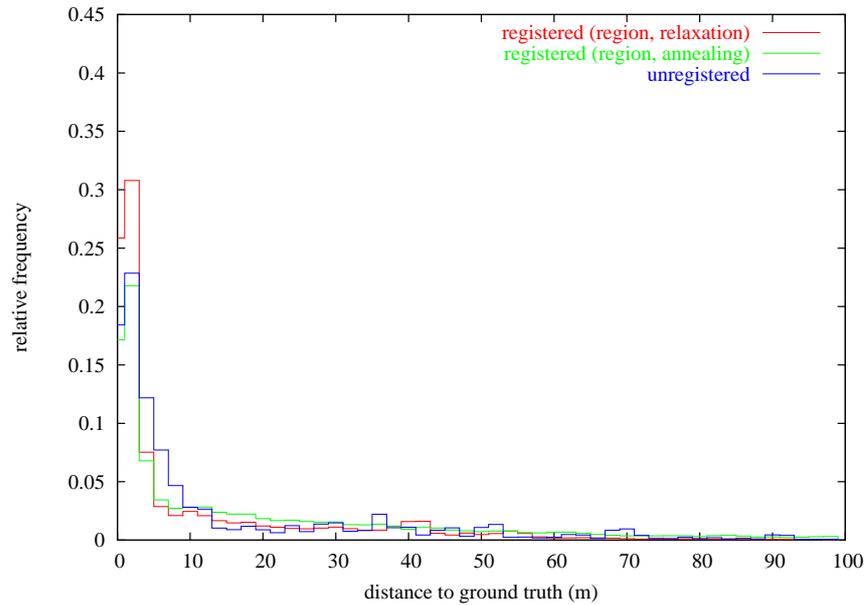


Figure 4.9: Histogram of distances from cadastre graph pixels to ground truth, in metres. Red line: cadastre registered by the region-based algorithm and probabilistic relaxation; green line: cadastre registered by the region-based algorithm and simulated annealing; blue line: unregistered cadastre.

many real field boundaries that do not have a corresponding cadastre edge, *length* is the total length of the unregistered or registered cadastre edges, including non-matchable edges, and *length (matchable)* is that *excluding* non-matchable edges. The average distances given in this table are these calculated using equation (4.27).

The large differences in table 4.1 in area between the unregistered and registered cadastres can be explained are due to the fact that, due to limitations in computational power, the test area has been divided into parts, and registration is performed separately on each part. Although the different parts are initially non-overlapping, a buffer area of some meters is added to each part, thus making them effectively overlapping. The registered cadastre has a tendency to spill over these buffer areas, thereby causing the sum of the areas covered by the registered cadastre of each part to be greater than the total surface of the test site. The large differences in length have a simpler explanation: unregistered cadastre edges are often long straight lines, whereas registered edges follow terrain details with pixel accuracy. Registered edges are therefore much longer than unregistered ones.

Figure 4.12 shows graphical results for the edge-based algorithm for a 1.4 km^2 region. Figure 4.13 show graphical results for the region-based algorithm with probabilistic relaxation, for a different 1 km^2 region. Note that we are working with 2 m-per-pixel images, so the resolution limit, both for the distance histograms and the average distances, is on the order of 2 m; in particular, this —and the fact that the first histogram bin is narrower than the others— may explain why so many more edges are at 0.5 m distance than at 0 m distance. See section 4.5.3 for further discussion.

The presented algorithms successfully register the cadastre onto the image. With the best parameter set, the edge-based algorithm reduces the average distance between the cadastre graph and a ground truth from 2.35 m to 1.64 m, 30.2% less; the region-based algorithm reduces it to 1.59 m, 32.2% less, with the probabilistic relaxation optimization, and to 1.84 m, 21.6%

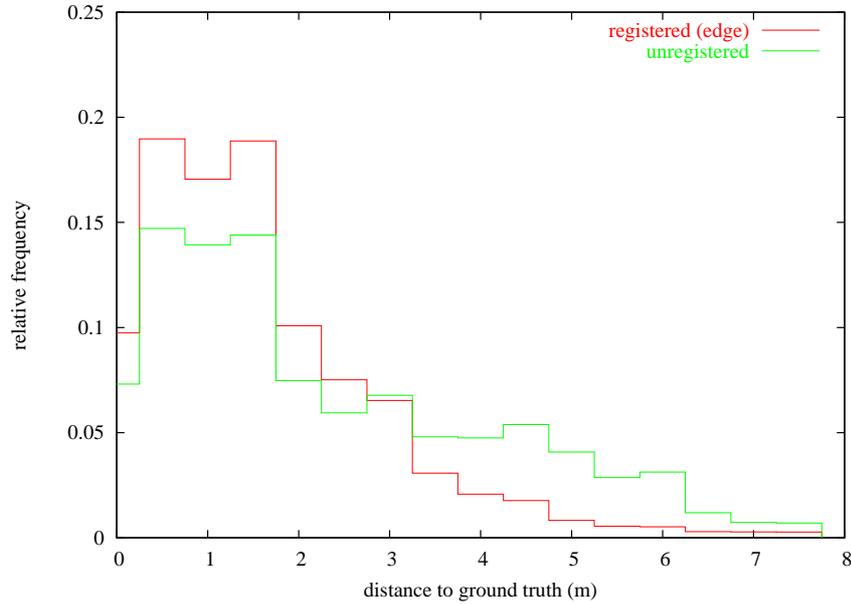


Figure 4.10: Histogram of distances from cadastre graph pixels to ground truth, in metres, excluding non-matchable edges (2 m per pixel, fast cooling). Red line: registered cadastre, using the edge-based algorithm; green line: unregistered cadastre. Note how the registered histogram shows higher counts than the unregistered histogram for low distances.

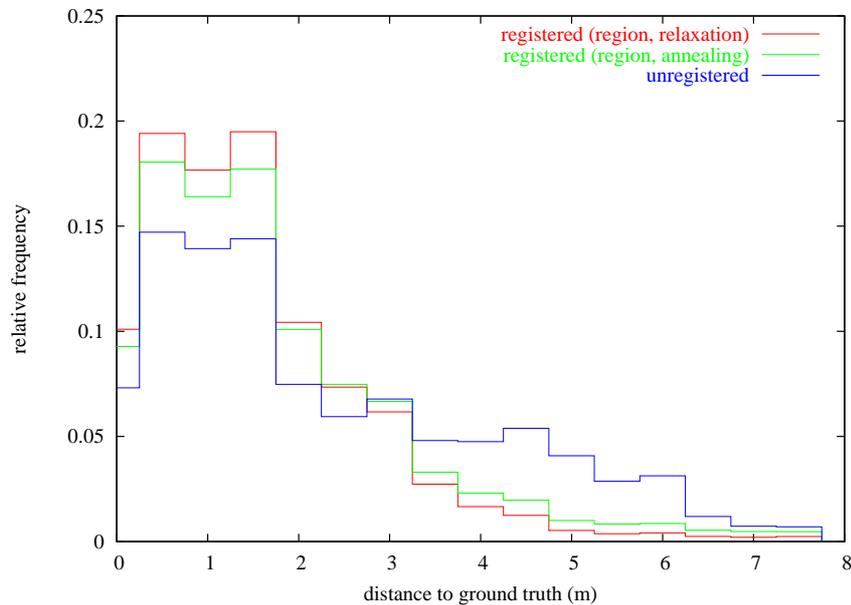


Figure 4.11: Histogram of distances from cadastre graph pixels to ground truth, in metres, excluding non-matchable edges (2 m per pixel, fast cooling). Red line: registered cadastre, using the region-based algorithm with probabilistic relaxation; green line: registered cadastre, using the region-based algorithm with simulated annealing; blue line: unregistered cadastre. Note how the registered histograms shows higher counts than the unregistered histogram for low distances.

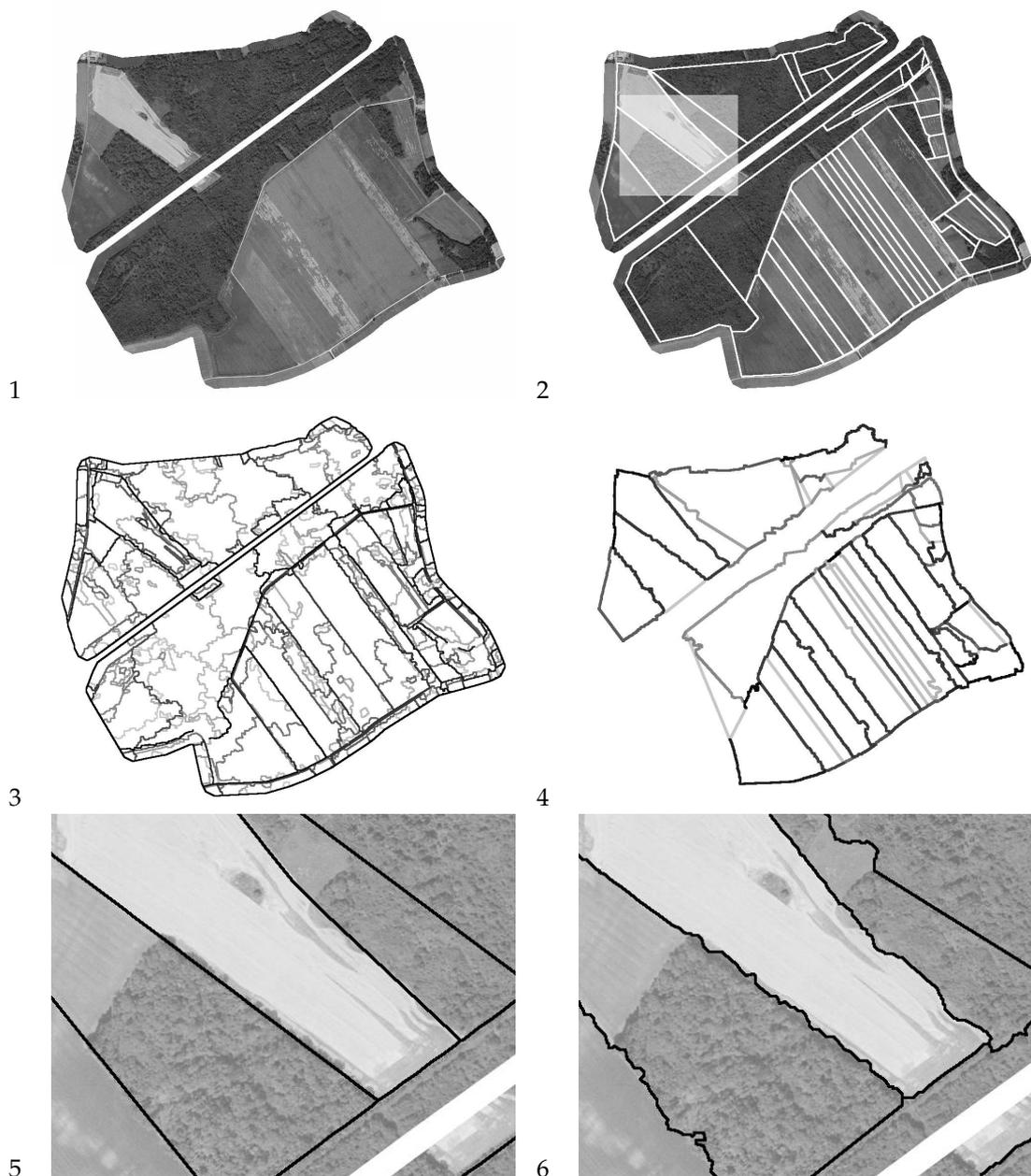


Figure 4.12: Results for a 1.4 km^2 portion of the test site, processed by the edge-based algorithm. Source image (1); cadastral graph C (2, boxed area is shown enlarged in 5 and 6); segmentation graph T (3, darker edges have higher apparition weights); registered cadastral (4, darker edges have higher registration ratios); close-up on the unregistered (5) and registered (6) cadastral. Edges which cannot be registered, such as the left-most edge of the lower area, are drawn as straight lines, which gives a strange visual effect; they will be removed in further processing steps anyway.

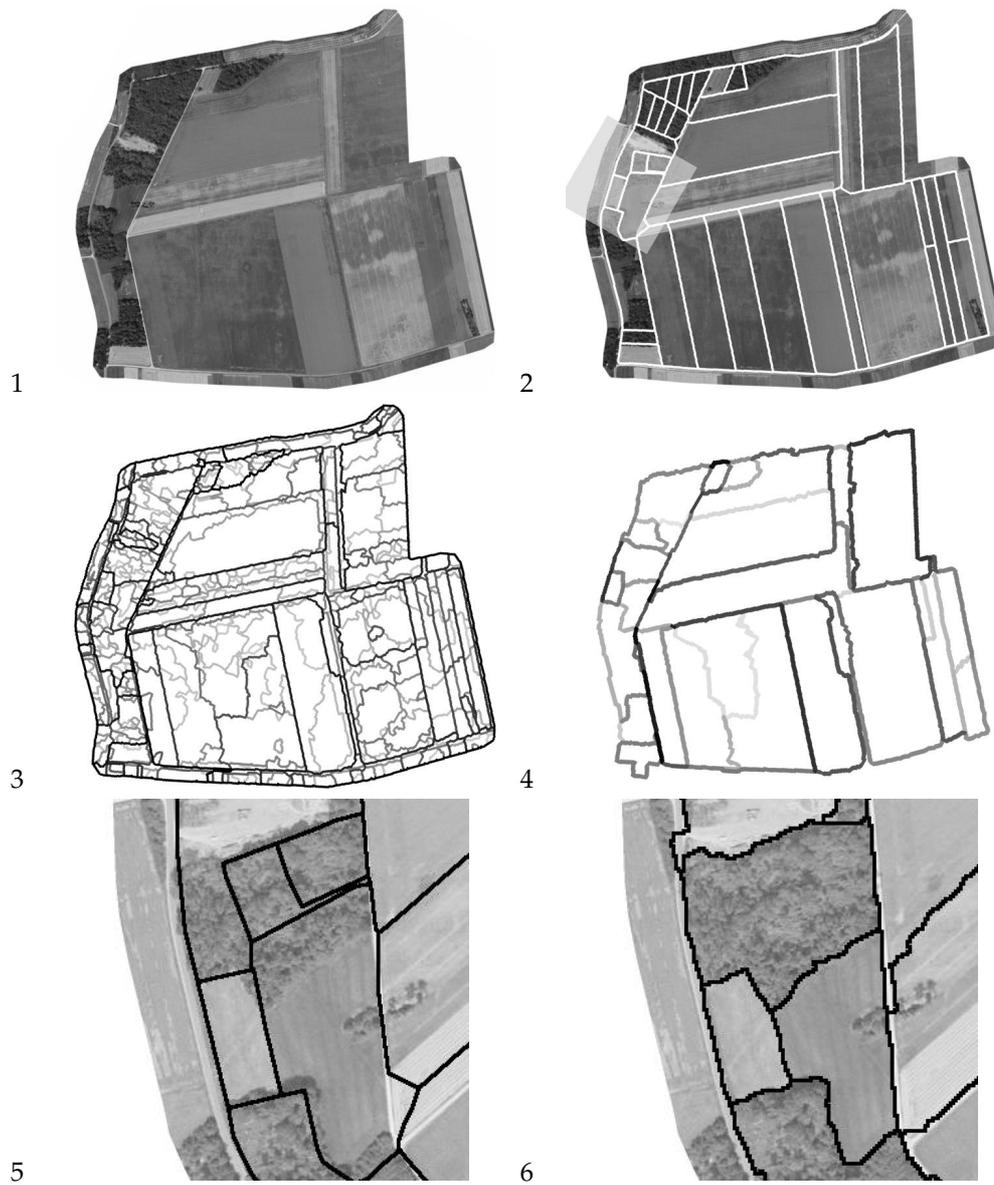


Figure 4.13: Results for a 1 km^2 portion of the test site, processed by the region-based algorithm, with probabilistic relaxation. Source image (1); cadastral graph C (2, boxed area is shown enlarged in 5 and 6); segmentation graph T (3, darker edges have higher apparition weights); registered cadastral (4, darker edges have higher values of the quality measure m); close-up on the unregistered (5) and registered (6) cadastral.

cadastre	unregistered	registered	registered	registered
algorithm		edge-based	region-based	region-based
algorithm variant		—	relaxation	annealing
area	3.993 km ²	7.547 km ²	8.386 km ²	8.011 km ²
ground truth length	208.9 km	208.9 km	208.9 km	208.9 km
length	95.9 km	188.0 km	210.6 km	283.5 km
length (matchable)	32.7 km	61.6 km	66.7 km	76.4 km
mean distance	2.351 m	1.640 m	1.594 m	1.844 m
iterations		2500	1000	15000
distance reduction		30.23%	32.19%	21.58%

Table 4.1: Evaluation results (2 m per pixel, fast convergence, best parameter set).

less, with the simulated annealing optimization. More important than this numerical result—which is nonetheless useful for comparison to other algorithms, or for selecting the best parameter set—is the fact that the resulting graph is registered onto segmentation edges, hence onto salient edges in the image, and therefore statistical analysis of these regions will be less perturbed by adjacent regions.

Visual inspection (see figure 4.12(4) and figure 4.13(4)) seems to show that the registration quality measures presented in sections 4.3.8 and 4.4.4 do indicate if a cadastre edge has a corresponding edge in the image. Using this information to delete cadastre edges that cannot be found in the image should be straightforward. In order to determine if this quality measure actually indicates if a cadastre edge exists in the image, I have calculated, for each value of this quality measure (suitably discretised and binned) the average, for all pixels in the registered cadastre which have the given value for the quality measure, of the distance to the nearest pixel in the ground truth. This is shown in figures 4.14, 4.15, and 4.16, for the edge-based algorithm, region-based algorithm with probabilistic relaxation, and region-based algorithm with simulated annealing, respectively. As before, we should consider that for pixels with a distance to ground truth below a certain threshold (for example, 6 m, as before) there is a corresponding image edge, and that for pixels above that threshold there is actually no corresponding image edge. We clearly see, for the edge-based algorithm and for the region-based algorithm with simulated annealing, that pixels with corresponding image edge have, on average, a high value of the registration quality measure, whereas those pixels that do not have a corresponding image edge have, on average, a low value of the registration quality measure. For example, for the edge-based algorithm, a threshold of 0.3 on the registration quality measure could be used to distinguish cadastre edges that exist in the image from those that do not. This trend can also be seen in the region-based algorithm with probabilistic relaxation, although not as clearly as for the other methods, so perhaps more research is needed for this case.

4.5.2 Parameter set selection

These algorithms use several parameters and weighting factors, and it is therefore necessary to find appropriate values for them. I found these optimal parameter values by scanning over the parameter space. It is usually desirable to have as few parameters as possible in a system; however, I found that the precise value of these parameters is not critical, as we obtain good results with a wide range of parameter sets.

To find the best parameter set for the edge-based algorithm, I ran two iterations of a steepest-descent method: on the first iteration, I ran the registration algorithm for several parameter sets,

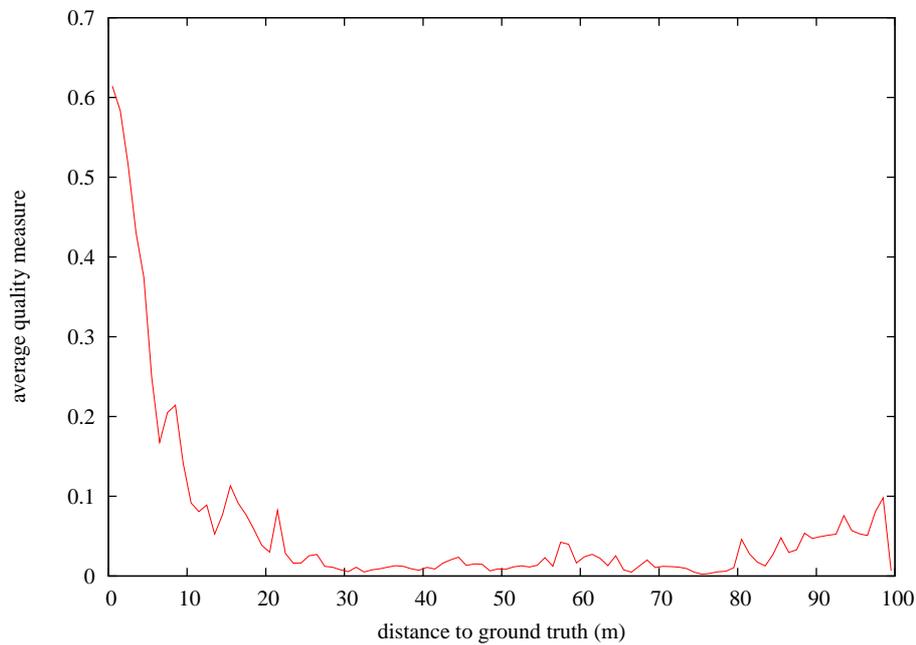


Figure 4.14: Average value, for each pixel in the registered cadastre, of the registration quality measure of section 4.3.8 as a function of the distance from that pixel to the nearest ground truth pixel, for the edge-based algorithm.

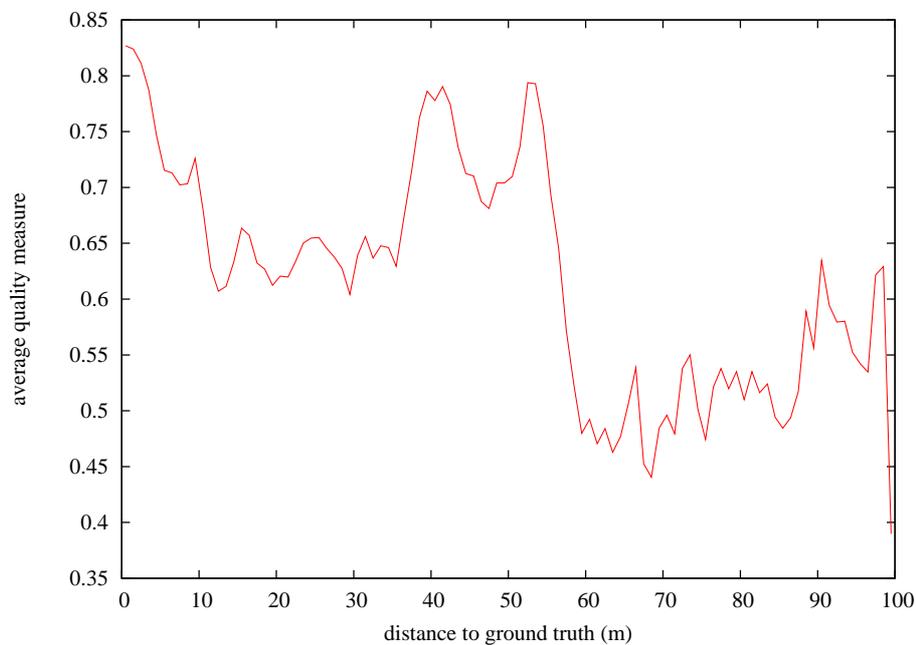


Figure 4.15: Average value, for each pixel in the registered cadastre, of the registration quality measure of section 4.3.8 as a function of the distance from that pixel to the nearest ground truth pixel, for the region-based algorithm with probabilistic relaxation.

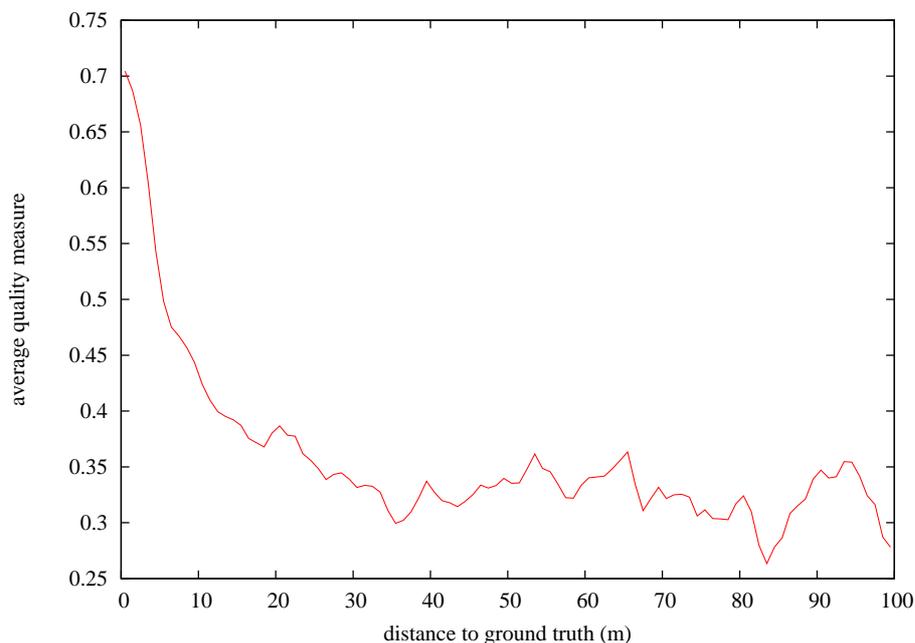


Figure 4.16: Average value, for each pixel in the registered cadastre, of the registration quality measure of section 4.3.8 as a function of the distance from that pixel to the nearest ground truth pixel, for the region-based algorithm with simulated annealing.

distributed on a grid on the parameter space; 243 parameter sets were tested. On the second iteration, the test parameter sets were chosen to be near the best-performing parameter set of the first iteration; 162 parameter sets were tested. Figure 4.17 shows the histograms, for the first and the second iterations, of the reduction of the distance between the registered cadastre and the ground truth, relative to the distance between the unregistered cadastre and the ground truth, for the different parameter sets; larger reductions are better. The best parameter set of the first iteration gave a reduction of 27.0%. That of the second iteration, 30.2%.

We proceeded similarly for the other algorithms. For the region-based algorithm with probabilistic relaxation, we tested 486 parameter sets on the first iteration, and 48 on the second one. Figure 4.18 shows the histograms, for the first and the second iterations, of the reduction of the distance between the registered cadastre and the ground truth, relative to the distance between the unregistered cadastre and the ground truth. The best parameter set of the first iteration gave a reduction of 31.7%, and that of the second iteration, 32.2%.

For the region-based algorithm with simulated annealing, we tested 324 parameter sets on the first iteration, and 81 on the second one. Figure 4.19 shows the corresponding histograms. The best parameter set of the first iteration gave a reduction of 21.6%, and that of the second iteration, 20.7%. We can see that we can reach very low performance values with some parameter sets: for this algorithm variant, the choice of a parameter set is critical, and the overall algorithm performance inferior. As we have seen in the main experiment of section 4.5.1, and as we will see in further sections, the region-based algorithm with simulated annealing is the one that performs worst.

The actual best parameter sets are given in table 4.2. By observing the histograms we can conclude that —except for the simulated-annealing variant of the region-based algorithm— the selection of parameter set is not critical: even for the first iteration of the steepest-descent algorithm, the histograms are centred around relatively good performance values, so even for

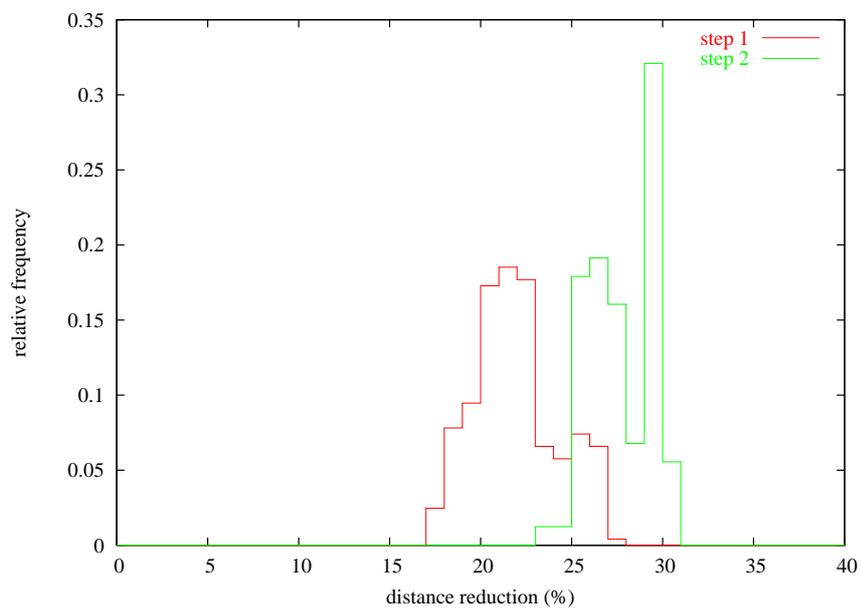


Figure 4.17: Influence of the parameter set on evaluation results (2 m per pixel, fast convergence), for the edge-based algorithm. Histograms of the reduction of the distance between the registered cadastre and the ground truth, relative to the distance between the unregistered cadastre and the ground truth, for the different parameter sets tested in the first and second iterations of a steepest-descent algorithm used to find the best parameter set. Larger reductions are better.

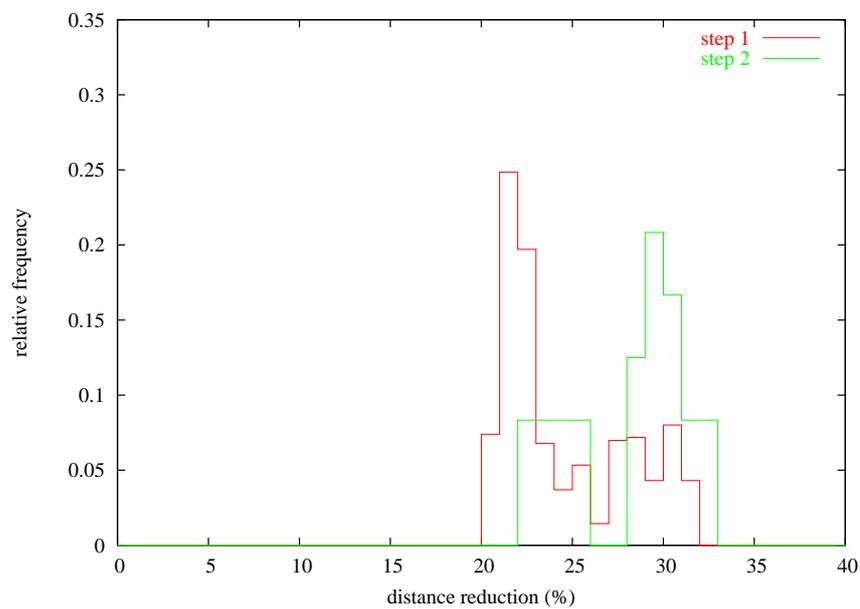


Figure 4.18: Influence of the parameter set on evaluation results (2 m per pixel, fast convergence), for the region-based algorithm with probabilistic relaxation. Histograms of the reduction of the distance between the registered cadastre and the ground truth, relative to the distance between the unregistered cadastre and the ground truth, for the different parameter sets tested in the first and second iterations of a steepest-descent algorithm used to find the best parameter set. Larger reductions are better.

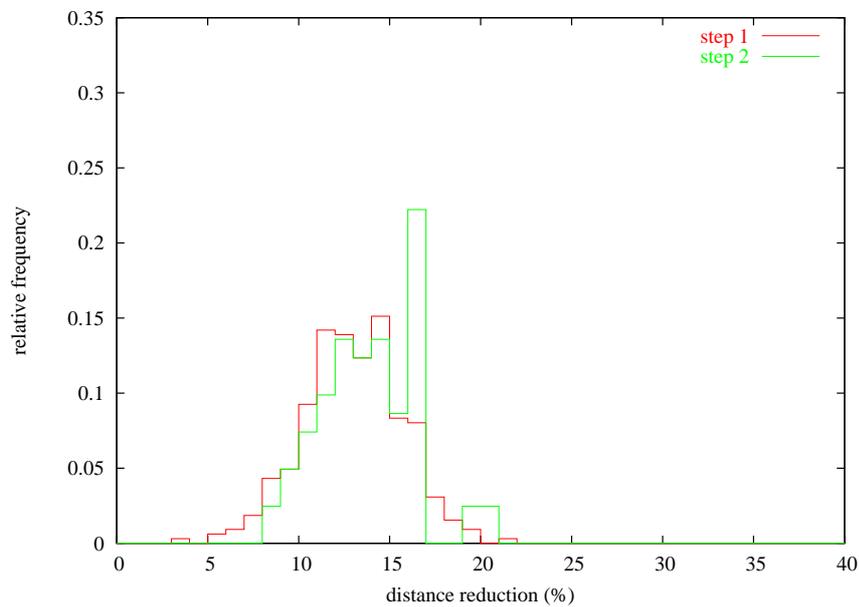


Figure 4.19: Influence of the parameter set on evaluation results (2 m per pixel, fast convergence), for the region-based algorithm with simulated annealing. Histograms of the reduction of the distance between the registered cadastre and the ground truth, relative to the distance between the unregistered cadastre and the ground truth, for the different parameter sets tested in the first and second iterations of a steepest-descent algorithm used to find the best parameter set. Larger reductions are better.

sub-optimal parameter sets chances are that the algorithms will perform well enough.

edge-based		region-based relaxation		region-based annealing	
α_1	6.5	k_0	0	k_0	0
α_2	0.22	k_1	1	k_1	1
α_3	3	k_2	1	k_2	1
α_4	18	k_3	1	k_3	2
α_5	0	k_4	1	k_4	0.7
α_6	8000	$q_i^{(k)}$	1	k_5	0.7
				k_6	0.1
				k_7	0.1
N_r	5	N_r	5	N_r	5
p_\perp	0.3	π_0	1.1	π_0	0.83
		π_1	0.6	π_1	0.13
		π_2	0.4	π_2	0.10
		π_r	0.1	π_r	0.03
		β	1	β	2

Table 4.2: Best-performing parameter sets.

4.5.3 Other experiments

In this section I present the results of additional tests that explore the effect of image resolution, cooling schedule, and cadastre quality on the algorithms' performance.

Increased resolution and number of iterations

Execution speed is critical for this application, which is why I used downsampled images and let the optimizer run for a relatively small number of iterations (2500 for the edge-based method, 1000 for the relaxation region-based method, and 15000 for the annealing region-based method), with parameters set for a fast convergence, or a fast cooling schedule. For example, for simulated annealing, I have used a cooling schedule $T_n = T_0 \cdot k^n$ with k close to, but less than, 1; a slower cooling schedule is then one with a k closer to 1 than a faster cooling schedule. For the edge-based algorithm, I took $k = 0.9$, and for the simulated annealing region-based algorithm, $k = 0.98$. These give fast execution times of the order of 1 minute per square kilometre.

However, to fully evaluate its performance I also ran the algorithms for more iterations (15000 for the edge-based method, 10000 for the relaxation region-based method, and 150000 for the annealing region-based method), with parameters set for a slower convergence ($k = 0.97$ for the edge-based algorithm, and $k = 0.99$ for the simulated annealing region-based algorithm), and with higher-resolution images (downsampled at 1 m per pixel instead of 2 m per pixel). The remaining parameters for all these test were the optimal ones found in section 4.5.2.

Figures 4.20, 4.21, and 4.22 show the results for the experiments with slow convergence compared to those with fast convergence, with 2 m-per-pixel images, for the edge-based, region-based with relaxation, and region-based with annealing algorithms respectively. Tables 4.3 and 4.8 summarize the results quantitatively. It can be seen that there is little difference between forcing a fast convergence and allowing a slower convergence—actually, slower convergence gives slightly worse results, although this could be just bad chance, since there is

a random component in these algorithms. Slower convergence, however, comes at the cost of increased computation time. This justifies the choice of a fast convergence as the default setting.

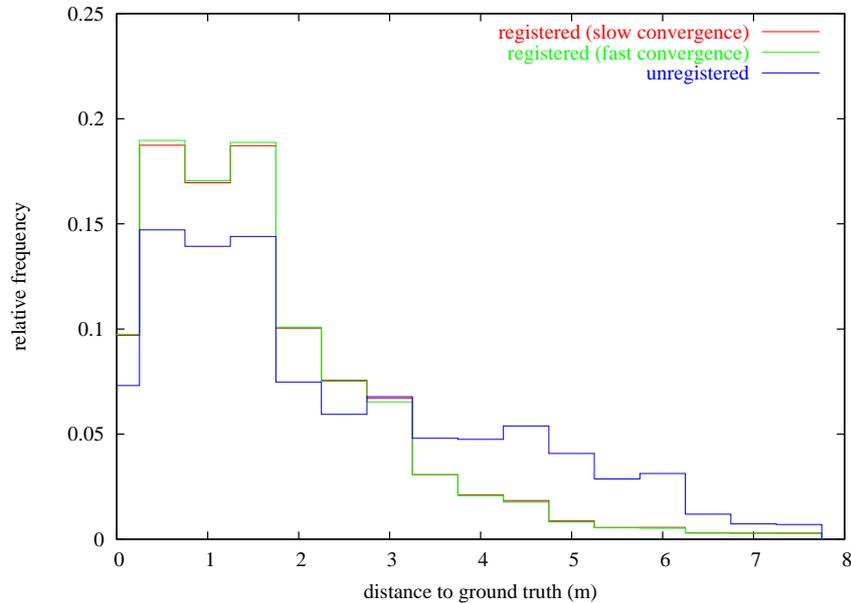


Figure 4.20: Histogram of distances from cadastral graph pixels to ground truth, in metres, excluding non-matchable edges, for the edge-based algorithm. Red line: registered cadastral, with parameters set for slow convergence; green line: registered cadastral, with parameters set for fast convergence; blue line: unregistered cadastral. Images are at 2 m per pixel.

Figures 4.23 and 4.24 show the results for the experiments with 1 m-per-pixel images—and input cadastral rasterized to 1 m per pixel—compared with results for 2 m-per-pixel images, for the region-based with relaxation and region-based with annealing algorithms respectively. The optimal parameter sets (with fast convergence) of section 4.5.2 have been used. Tables 4.4 and 4.8 summarize the results quantitatively. Similar tests with the edge-based algorithm could not be performed for lack of computational power. We can see that the algorithms exhibit worse performance at 1 m per pixel than at 2 m per pixel, probably because they cannot cope with the increase in data volume caused by this higher resolution.

Finally, figures 4.25 and 4.26 explore what happens when both slow convergence and 1 m-per-pixel images are used, for the region-based with relaxation and region-based with annealing algorithms respectively. Tables 4.4 and 4.8 summarize the results quantitatively. Similar tests with the edge-based algorithm could not be performed for lack of computational power. We can see that the algorithms perform worse than at 2 m per pixel, also probably because they cannot cope with the increase in data volume caused by this higher resolution, despite the slower convergence allowed in these tests. This slower convergence, however, makes the region-based algorithm with simulated annealing perform better than with fast convergence at 1 m per pixel, but still not as well as in the main experiment (2 m per pixel with fast convergence). Despite showing weaknesses of these algorithms, these tests justify the choice of default operating mode—images at 2 m per pixel, and fast convergence—which also make the algorithms run fastest.

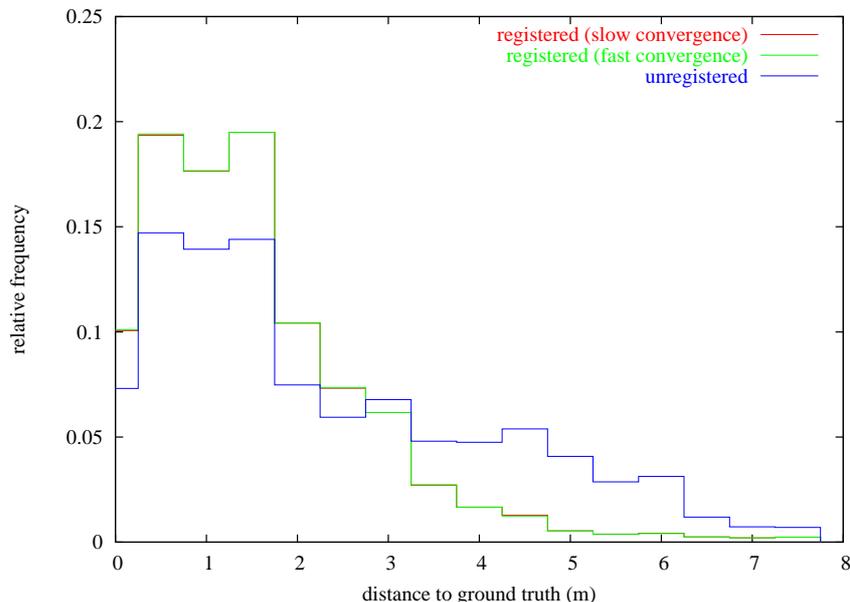


Figure 4.21: Histogram of distances from cadastral graph pixels to ground truth, in metres, excluding non-matchable edges, for the region-based algorithm with probabilistic relaxation. Red line: registered cadastral, with parameters set for slow convergence; green line: registered cadastral, with parameters set for fast convergence; blue line: unregistered cadastral. Images are at 2 m per pixel.

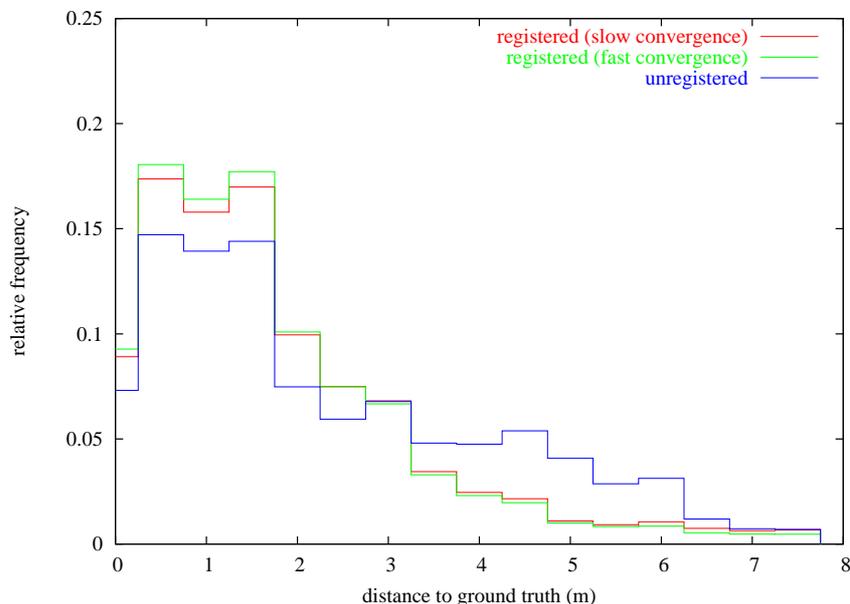


Figure 4.22: Histogram of distances from cadastral graph pixels to ground truth, in metres, excluding non-matchable edges, for the region-based algorithm with simulated annealing. Red line: registered cadastral, with parameters set for slow convergence; green line: registered cadastral, with parameters set for fast convergence; blue line: unregistered cadastral. Images are at 2 m per pixel.

cadastre	unregistered	registered	registered	registered
algorithm		edge-based	region-based	region-based
algorithm variant		—	relaxation	annealing
area	3.993 km ²	7.530 km ²	8.387 km ²	8.012 km ²
ground truth length	208.9 km	208.9 km	208.9 km	208.9 km
length	95.9 km	187.3 km	210.7 km	278.0 km
length (matchable)	32.7 km	61.2 km	66.9 km	79.7 km
mean distance	2.351 m	1.665 m	1.605 m	2.001 m
distance reduction		29.18%	31.72%	14.90%
mean dist. (fast convergence)	2.351 m	1.640 m	1.594 m	1.844 m

Table 4.3: Evaluation results at 2 m per pixel, parameters controlling convergence speed set for slow convergence; other parameters are taken from the best parameter set. For easier comparison, the mean registered distance for algorithms set for fast convergence (table 4.1) is also given.

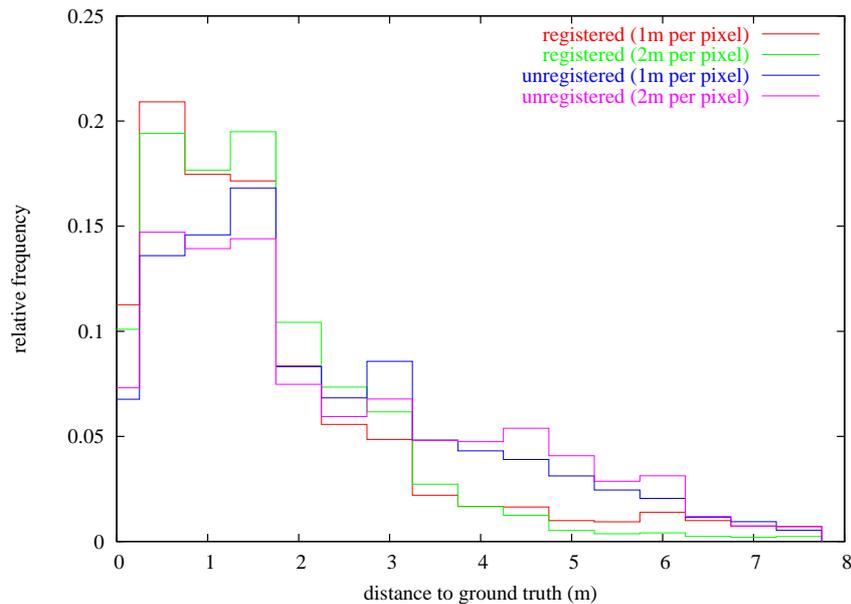


Figure 4.23: Histogram of distances from cadastral graph pixels to ground truth, in metres, excluding non-matchable edges, for the region-based algorithm with probabilistic relaxation. Red line: registered cadastral, using 1 m-per-pixel images; green line: registered cadastral, using 2 m-per-pixel images; blue line: unregistered cadastral, rasterised to 1 m-per-pixel; magenta line: unregistered cadastral, rasterised to 2 m-per-pixel. Parameters are set for fast convergence. Unregistered cadastral data at 1 m per pixel and 2 m per pixel are different because of the different rasterisation grid used.

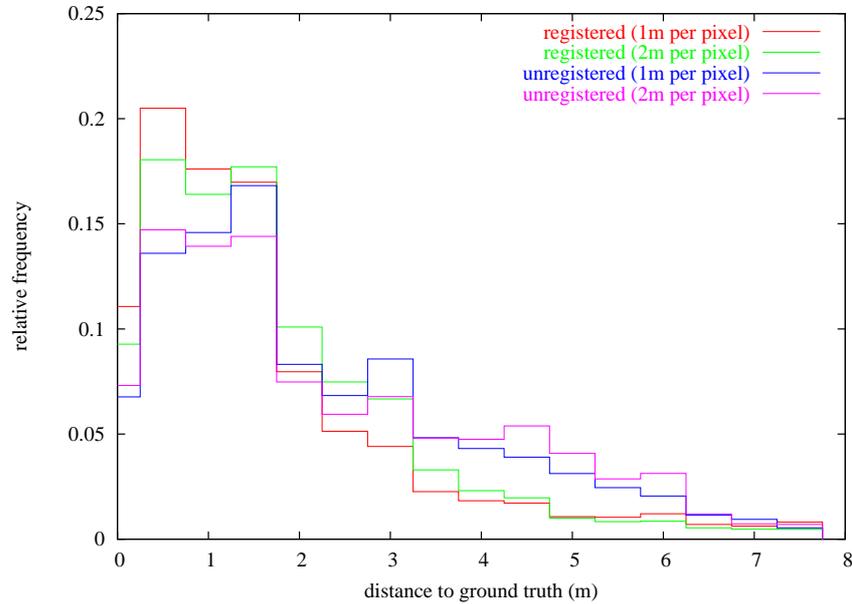


Figure 4.24: Histogram of distances from cadastral graph pixels to ground truth, in metres, excluding non-matchable edges, for the region-based algorithm with simulated annealing. Red line: registered cadastral, using 1 m-per-pixel images; green line: registered cadastral, using 2 m-per-pixel images; blue line: unregistered cadastral, rasterised to 1 m-per-pixel; magenta line: unregistered cadastral, rasterised to 2 m-per-pixel. Parameters are set for fast convergence. Unregistered cadastral data at 1 m per pixel and 2 m per pixel are different because of the different rasterisation grid used.

cadastre	unregistered	registered	registered	registered
algorithm		edge-based	region-based	region-based
algorithm variant		—	relaxation	annealing
area	3.990 km ²	not tested	8.074 km ²	7.977 km ²
ground truth length	208.9 km	not tested	208.9 km	208.9 km
length	109.5 km	not tested	260.3 km	323.9 km
length (matchable)	38.1 km	not tested	83.5 km	74.3 km
mean distance	2.196 m	not tested	1.795 m	1.989 m
distance reduction		not tested	18.29%	9.43%
mean dist. (2 m)	2.351 m	1.640 m	1.594 m	1.844 m

Table 4.4: Evaluation results at 1 m per pixel, optimal parameter set (and fast convergence). Due to a different discretisation, the values for the unregistered cadastral graph are different to those in table 4.1. For easier comparison, the mean registered distance for algorithms set for 2 m-per-pixel images (table 4.1) is also given.

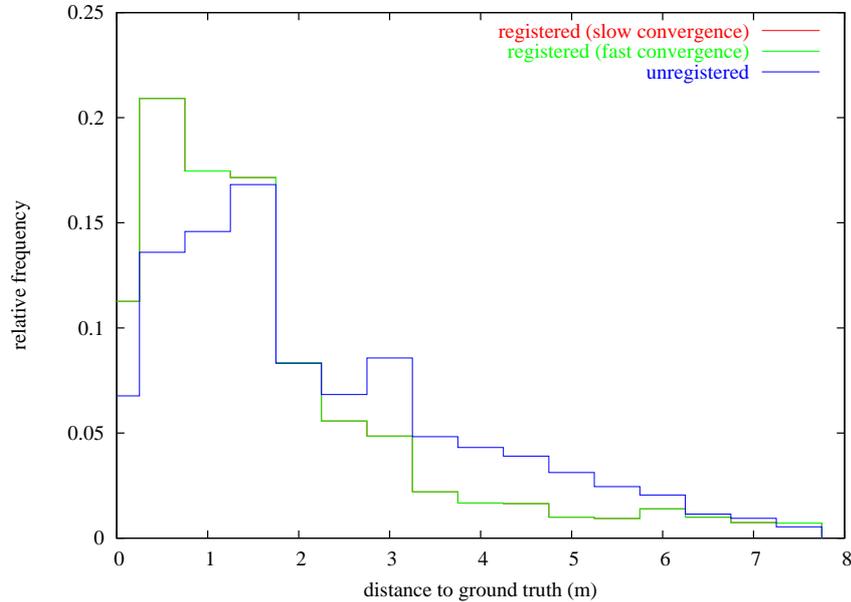


Figure 4.25: Histogram of distances from cadastral graph pixels to ground truth, in metres, excluding non-matchable edges, for the region-based algorithm with probabilistic relaxation. Red line: registered cadastral, using 1 m-per-pixel images and slow convergence; green line: registered cadastral, using 1 m-per-pixel images and fast convergence; blue line: unregistered cadastral, rasterised to 1 m-per-pixel. The green line overlaps the red line almost everywhere.

cadastre	unregistered	registered	registered	registered
algorithm		edge-based	region-based	region-based
algorithm variant		—	relaxation	annealing
area	3.990 km ²	not tested	8.074 km ²	7.993 km ²
ground truth length	208.9 km	not tested	208.9 km	208.9 km
length	109.5 km	not tested	260.3 km	329.4 km
length (matchable)	38.1 km	not tested	83.5 km	74.9 km
mean distance	2.196 m	not tested	1.795 m	1.821 m
distance reduction		not tested	18.29%	17.08%
mean dist. (fast, 2 m)	2.351 m	1.640 m	1.594 m	1.844 m

Table 4.5: Evaluation results at 1 m per pixel, slow convergence. Due to a different discretisation, the values for the unregistered cadastral graph are different to those in table 4.1. For easier comparison, the mean registered distance for algorithms set for 2 m-per-pixel images and fast convergence (table 4.1) is also given.

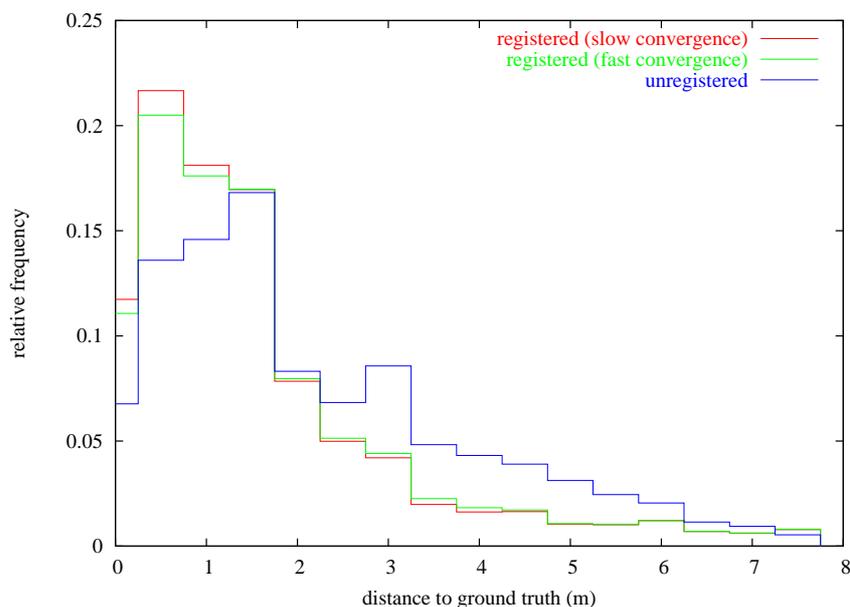


Figure 4.26: Histogram of distances from cadastre graph pixels to ground truth, in metres, excluding non-matchable edges, for the region-based algorithm with simulated annealing. Red line: registered cadastre, using 1 m-per-pixel images and slow convergence; green line: registered cadastre, using 1 m-per-pixel images and fast convergence; blue line: unregistered cadastre, rasterised to 1 m-per-pixel.

Worse input cadastre

A related issue is that it may be difficult to fully appreciate the performance of these algorithms with this cadastre data, since the initial mis-registration is not very large in terms of distance to the ground truth (but note that reducing this distance is only one of the goals of this algorithm, the other being to obtain cadastre edges that follow the geometrical details of the image edges). That is, should the best performance of these algorithms be given as “reduce average distance to 1.7 m” or “reduce average distance by 30%”?

To decide that, I manually distorted the cadastre graph and ran the algorithm, with 1 m-per-pixel and 2 m-per-pixel images, with the fast cooling algorithm and the optimal parameter set.

Figures 4.27, 4.28, and 4.29 show the results for the experiments with the manually distorted input cadastre compared to those with the original cadastre, with 2 m-per-pixel images, for the edge-based, region-based with relaxation, and region-based with annealing algorithms respectively, for fast convergence. Tables 4.6 and 4.8 summarize the results quantitatively. It can be seen that despite large quality differences between the distorted and original unregistered cadastres (blue and magenta lines), the registration of both cadastres reaches similar distances to ground truth (red and green lines). It seems that the obtained quality levels are some sort of upper limit that the algorithms reach independently of the quality of the input data, but beyond which they cannot go, and that therefore the performance should be described as “reduction to 1.7 m” instead of “reduction by 30%”.

Figures 4.30 and 4.31 show the results for the experiments with the manually distorted input cadastre compared to those with the original cadastre, with 1 m-per-pixel images, for the region-based with relaxation, and region-based with annealing algorithms respectively, for a

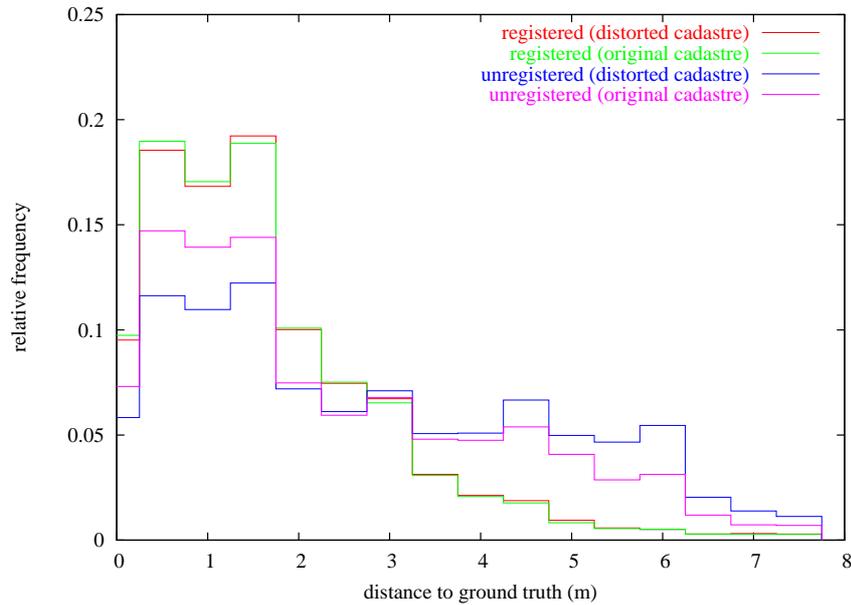


Figure 4.27: Histogram of distances from cadastral graph pixels to ground truth, in metres, excluding non-matchable edges, for the edge-based algorithm. Red line: registration of manually distorted cadastre; green line: registration of original cadastre; blue line: unregistered distorted cadastre; magenta line: unregistered distorted cadastre. Images are at 2 m per pixel.

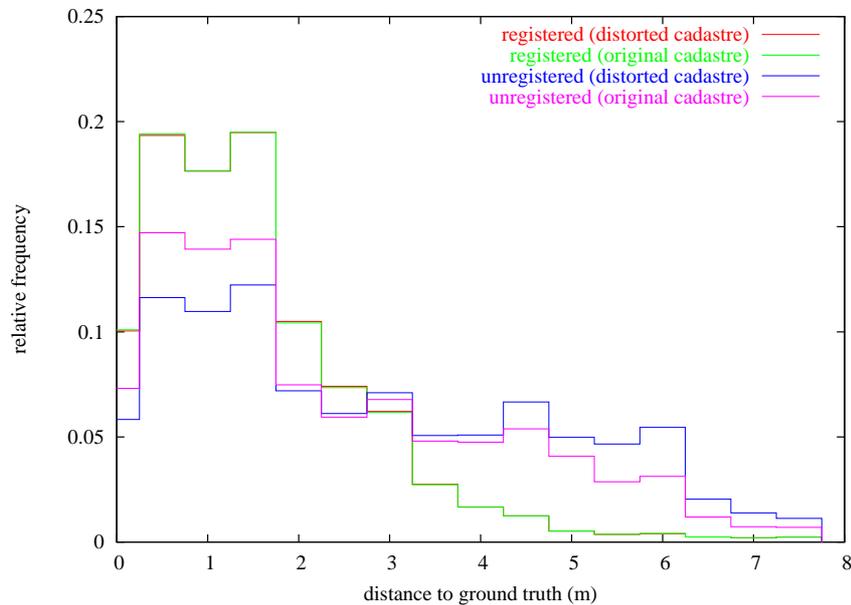


Figure 4.28: Histogram of distances from cadastral graph pixels to ground truth, in metres, excluding non-matchable edges, for the region-based algorithm with probabilistic relaxation. Red line: registration of manually distorted cadastre; green line: registration of original cadastre; blue line: unregistered distorted cadastre; magenta line: unregistered distorted cadastre. Images are at 2 m per pixel.

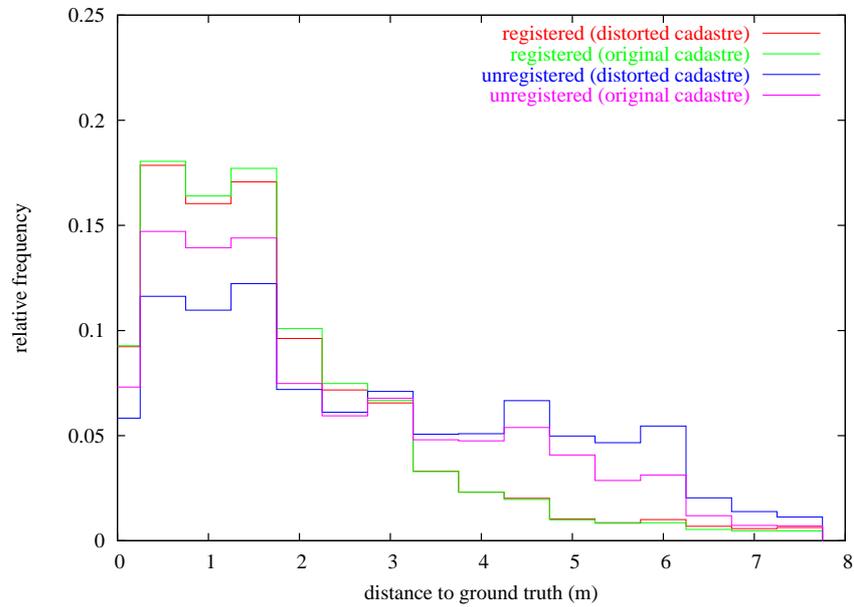


Figure 4.29: Histogram of distances from cadastre graph pixels to ground truth, in metres, excluding non-matchable edges, for the region-based algorithm with simulated annealing. Red line: registration of manually distorted cadastre; green line: registration of original cadastre; blue line: unregistered distorted cadastre; magenta line: unregistered distorted cadastre. Images are at 2 m per pixel.

cadastre	unregistered	registered	registered	registered
algorithm		edge-based	region-based	region-based
algorithm variant		—	relaxation	annealing
area	3.985 km ²	7.464 km ²	8.379 km ²	7.974 km ²
ground truth length	208.9 km	208.9 km	208.9 km	208.9 km
length	95.4 km	183.7 km	211.3 km	291.3 km
length (matchable)	31.7 km	57.9 km	66.1 km	81.1 km
mean distance	2.831 m	1.665 m	1.595 m	2.020 m
distance reduction		41.20%	43.65%	28.62%

Table 4.6: Evaluation results for registration at 2 m per pixel, best parameter set and fast convergence, using a manually distorted cadastre as input.

fast convergence. Similar tests with the edge-based algorithm could not be performed for lack of computational power. Tables 4.7 and 4.8 summarize the results quantitatively. As for the 2 m-per-pixel images, it can be seen that despite large quality differences between the distorted and original unregistered cadastres (blue and magenta lines), the registration of both cadastres reaches similar distances to ground truth (red and green lines). The probabilistic relaxation variant performs better at 2 m-per-pixel and the simulated annealing variant performs better at 1 m-per-pixel for this manually distorted cadastre.

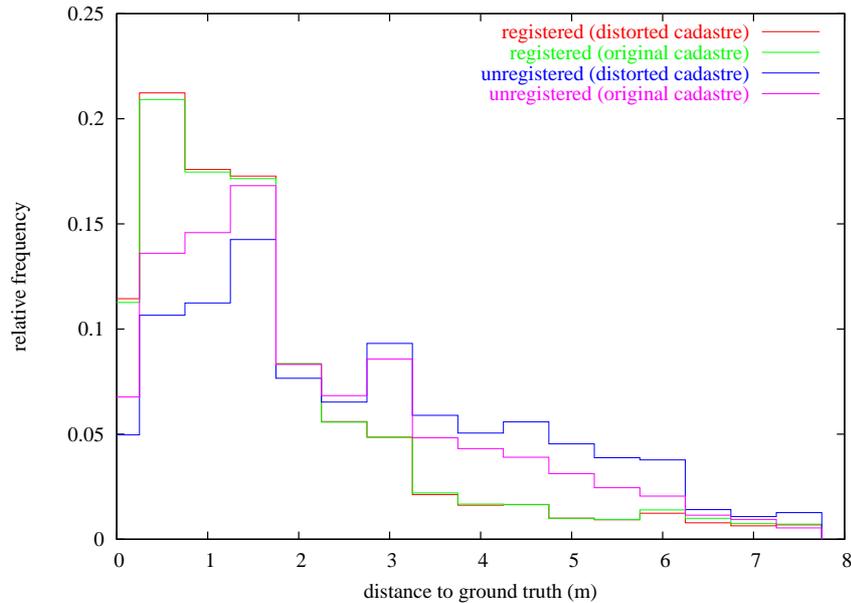


Figure 4.30: Histogram of distances from cadastre graph pixels to ground truth, in metres, excluding non-matchable edges, for the region-based algorithm with probabilistic relaxation. Red line: registration of manually distorted cadastre; green line: registration of original cadastre; blue line: unregistered distorted cadastre; magenta line: unregistered distorted cadastre. Images are at 1 m per pixel.

cadastre	unregistered	registered	registered	registered
		edge-based	region-based	region-based
algorithm		—	relaxation	annealing
algorithm variant				
area	3.982 km ²	not tested	8.067 km ²	7.963 km ²
ground truth length	208.9 km	not tested	208.9 km	208.9 km
length	113.0 km	not tested	263.1 km	328.2 km
length (matchable)	38.7 km	not tested	83.3 km	72.2 km
mean distance	2.734 m	not tested	1.756 m	1.906 m
distance reduction		not tested	35.77%	30.28%

Table 4.7: Evaluation results for registration at 1 m per pixel, best parameter set and fast convergence, using a manually distorted cadastre as input.

Even though the unregistered distance was greater, the algorithms still manage to register the cadastre graph to a distance of about 1.7 m, which, as stated before, appears to be a performance limit.

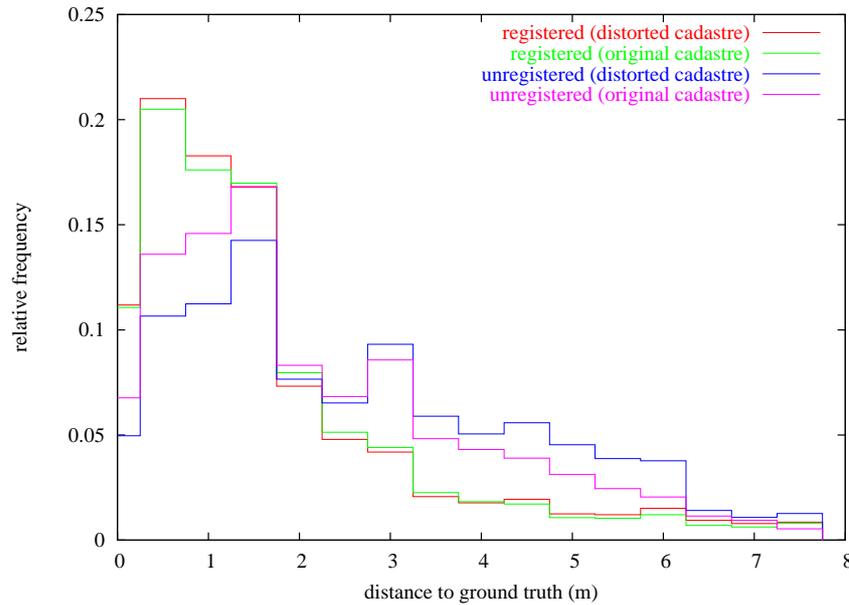


Figure 4.31: Histogram of distances from cadastre graph pixels to ground truth, in metres, excluding non-matchable edges, for the region-based algorithm with simulated annealing. Red line: registration of manually distorted cadastre; green line: registration of original cadastre; blue line: unregistered distorted cadastre; magenta line: unregistered distorted cadastre. Images are at 1 m per pixel.

4.5.4 Summary of experiments

Table 4.8 summarizes the results of all the experiments. Neither using a slower cooling schedule nor working with higher-resolution images improves the registered distance— actually, in most cases they seem cause worse performances. On the other hand, tests with a manually distorted cadastre graph show that the upper performance limits are to be measured in absolute distances, not as a relative improvement, for both with the small unregistered distance of the original cadastre and the larger unregistered distance of the distorted cadastre the algorithms reach similar registered distances.

4.6 Homogeneity tests

The algorithms described above do not always produce homogeneous regions. If a cadastre region is itself not homogeneous —and not just because of some stray pixels in the border, which would be corrected by the registration algorithms, but because it actually contains more than one plot— in most cases the registration algorithms will produce a heterogeneous region. Since the classification algorithms in chapter 5 assume regions to be homogeneous, we need a way to detect these heterogeneous regions and, eventually, to decompose them into homogeneous sub-regions. I propose two ideas, which unfortunately I did not have the time to implement in this thesis:

The first idea is to analyse the terrain edges contained in the registered cadastre regions, excluding terrain edges closer than a certain distance to the region edge. The maximum the maximum appearance weight of the remaining edges would be calculated. High values would

conv.	resolution	cadastre	algorithm	unreg. dist.	distance	
fast	2 m	original	edge	2.351 m	1.640 m	-30.23%
fast	2 m	original	region relax.	2.351 m	1.594 m	-32.19%
fast	2 m	original	region anneal.	2.351 m	1.844 m	-21.58%
fast	1 m	original	edge	test results not available		
fast	1 m	original	region relax.	2.196 m	1.795 m	-18.29%
fast	1 m	original	region anneal.	2.196 m	1.989 m	-9.43%
fast	2 m	distorted	edge	2.831 m	1.665 m	-41.20%
fast	2 m	distorted	region relax.	2.831 m	1.595 m	-43.65%
fast	2 m	distorted	region anneal.	2.831 m	2.020 m	-28.62%
fast	1 m	distorted	edge	test results not available		
fast	1 m	distorted	region relax.	2.734 m	1.756 m	-35.77%
fast	1 m	distorted	region anneal.	2.734 m	1.906 m	-30.28%
slow	2 m	original	edge	2.351 m	1.665 m	-29.18%
slow	2 m	original	region relax.	2.351 m	1.605 m	-31.72%
slow	2 m	original	region anneal.	2.351 m	2.001 m	-14.90%
slow	1 m	original	edge	test results not available		
slow	1 m	original	region relax.	2.196 m	1.795 m	-18.29%
slow	1 m	original	region anneal.	2.196 m	1.821 m	-17.08%

Table 4.8: Evaluation results for all experiments (“conv.”: cooling schedule, or speed of convergence; “edge”: edge-based algorithm; “region relax.”: region-based algorithm with probabilistic relaxation; “region anneal.”: region-based algorithm with simulated annealing; “unreg. dist.”: distance for unregistered cadastre).

indicate that the region is not homogeneous and should be split into several regions. We could also take into account the area of the region: in small regions, lack of high-valued terrain edges is not an indicator of homogeneity.

The second idea is to use the fact that, if a region is heterogeneous, the classification algorithms of chapter 5 would give a very low confidence to whatever terrain class is assigned to the region. In that case, we should attempt to partition the region following high-saliency edges, classify each of the sub-regions, and compare their confidences to that of the whole region.

Another problem is that it is possible that a single plot spans several cadastre regions. Although not as important a problem as that of registered regions being heterogeneous, it may be appropriate to merge adjacent cadastre regions containing the same crop. This can be done in two ways.

First, the registration quality measures of both algorithms are good indicators of whether or not the edge follows a true terrain limit. This can be used as a clue to merge adjacent regions: if the cadastre edge between them has a low registration ratio, they probably can be merged since they have the same texture and colour.

Second, the class assigned to adjacent regions can be used. However, because classes are coarse, we cannot simply merge any two adjacent regions of the same terrain type. For example, adjacent regions, one containing a yellow-coloured field and another containing a brown-coloured field should not be merged together.

4.7 Discussion and conclusion

In this chapter I have presented two graph matching algorithms specifically tailored for asymmetric graph matching problems —where most of the elements of one graph do not have a match in the other graph— which can be used to register a cadastre graph onto an aerial image. The graph corresponding to the image is derived from a multi-scale segmentation.

In the first algorithm, edge-based, the edges of this graph are asymmetrically matched to the edges of the cadastre graph —which correspond to edges between land plots— by optimizing, using simulated annealing, the fitness of a solution. We use an ad-hoc method for the registration near cadastre nodes. In the second algorithm, region-based, it is the faces of the segmentation graph which are matched to the faces of the cadastre graph —each such face corresponds to a land plot—, and either probabilistic relaxation or simulated annealing can be used for the optimization.

Extensive tests show that, numerically, the region-based method with probabilistic relaxation performs best, followed by the edge-based method, and, far behind, the region-based method with simulated annealing. The choice between the first two is not so clear-cut, however, because the algorithms exhibit qualitatively different behaviour: The edge-based algorithm preserves the spatial distribution of the cadastre graph much better, at the cost of adding auxiliary edges that do not correspond to image edges. The region-based algorithm, on the other hand, strictly follows image edges, at the cost of not always preserving spatial distribution. Although numerical evaluation gives better figures for the region-based algorithm with probabilistic relaxation, for a given application the trade-off between geometrical precision and topology preservation should be taken into account and perhaps the edge-based algorithm may be a better choice.

As a side effect of the registration we obtain a parameter, the registration quality measure, which seems useful as an indicator of which cadastre edges actually exist in the image. Formal tests to determine if this is actually a good indicator for that purpose support this, but show that further work may be necessary.

We have performed additional tests to more precisely determine the behaviour of these algorithms. First, a series of tests were run to obtain the best parameter set; these tests also allowed us to determine how sensitive the algorithms are to the precise choice of parameters: it turns out that they are not very sensitive to this. Furthermore, we studied how the algorithms behaved when higher-resolution images were used, or when they were given more time to converge: results are mixed but, overall, the algorithms seem to perform as well as before, or worse; this justifies the initial choice of low-resolution input and fast convergence, which at the same time makes the algorithms fast enough for production use. Finally, we tested how well the algorithms coped with a manually-distorted cadastre: we found out that their output for this worse-quality cadastre is as good as the output for the original cadastre, which seems to suggest that the algorithms' performance limit does not depend on the quality of the input data.

5

Classification

5.1 Introduction

The main processing step in the land-cover classification system presented in this thesis is the actual classification. In this chapter I will describe methods for performing this classification. These methods take an image as input—and its transformed colour channels and texture features, as described in appendix A—, a partition of this image into regions which are homogeneous in class—all pixels in a region belong to the same type of terrain—and a mathematical model that describes the behaviour of different terrain types. After analysing the image, region by region, and comparing it to the model, the classification algorithm decides which terrain type is the most likely for each region, and also reports on its own confidence on this decision.

Figure 5.1 shows as a flowchart the part of the complete process described in this chapter.

Classification involves some sort of model estimation, where a description of the relationship between terrain types and image pixels belonging to that type is obtained. Classification algorithms differ in the type of model used, the methods used to obtain a model given some data, and the methods used to determine the terrain class for each pixel given the models and the data. In a kind of classification algorithms, called *supervised classification* algorithms, models are estimated from *training data*, a subset of data manually annotated by a human operator. The models are then used to classify the remaining data. In *unsupervised classification* or *clustering*, data is grouped into classes in a single step, without learning a model from training data; it could be said that the final grouping is itself a model.

In supervised probabilistic modelling, the parameters for a random variable are estimated for each terrain class. Each pixel belonging to that class is assumed to be a realisation of its class' random variable. Many classification systems which are not explicitly probabilistic can be interpreted as probabilistic, where the random variables are uniform with a support determined by the estimation phase.

This chapter presents two contributions to supervised probabilistic classification. In the first one, *simple* or *flat per-region classification*, I present new classification algorithms that use standard probabilistic models but operate on a region-by-region basis: Instead of deciding on the most likely terrain type of each pixel separately, a single decision is taken for all pixels in a region—the partition of an image into regions being an input to the algorithm. This has two advantages. First, salt-and-pepper noise is greatly reduced. Second, because decisions are taken based on more information, we can expect them to be more accurate.

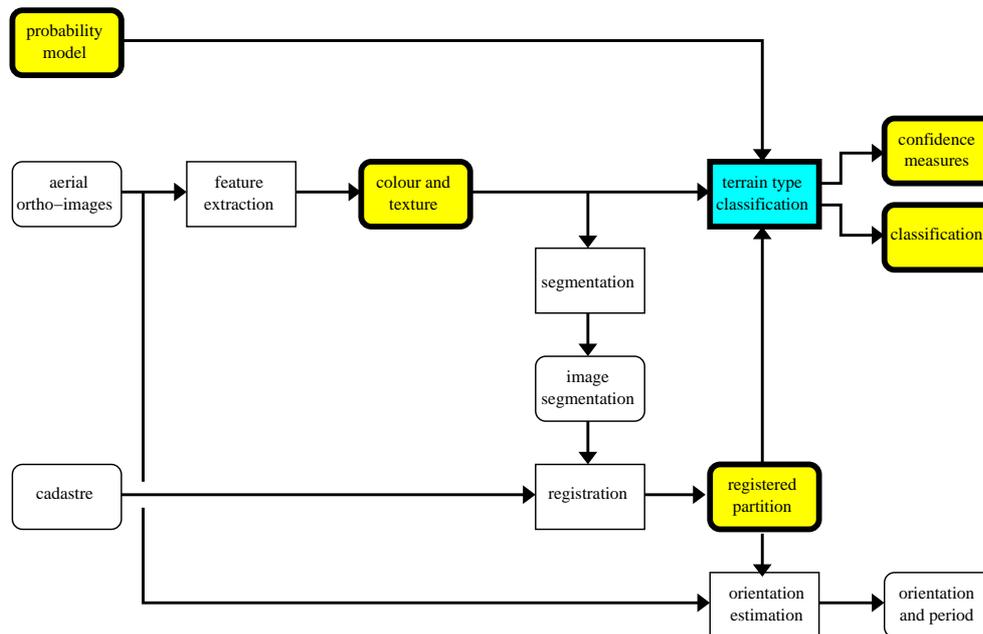


Figure 5.1: Flowchart depicting the processing steps (in blue rectangles) and intermediate results (in yellow rounded boxes) involved in the classification of image regions, in the context of the complete system.

The second contribution of this chapter is a probabilistic model for image classification which does not assume that each pixel is directly a realization of its class' random variable. Again using a partition of the image into regions, I hypothesize that, for certain terrain classes, pixel behaviour depends not only on the terrain class, but also on the specific region that the pixel is in. That is, pixels in the same region are realizations of the same random variable, but pixels in different regions, even if they are of the same terrain type, are realizations of different random variables. This is justified by the fact that a single terrain class, such as "forest", may look different depending on tree species, slope, season, weather, time of the day, and many other factors. However, and instead of splitting the "forest" class into several sub-classes, one for each value of these factors, and separately learning models for each of them—which would require enormous training sets—I further hypothesize that all these random variables corresponding to a single terrain class are related in ways that can themselves be modelled. The probabilistic model presented in this chapter describes this two-layered structure with what we call *nested random variables* or *nested per-region classification*. I also present classification algorithms that use this new probabilistic model.

This chapter is structured as follows. In section 5.2.1 the classical per-pixel Bayesian classification model is introduced; the reader will surely already be familiar with this model, which is presented just for the sake of establishing a uniform notation for the remainder of the text and for clarity of exposition. In section 5.2.2 the flat models for per-region classification presented in [TSB05] are described in detail, and some additional flat models are presented. In section 5.3 the nested probabilistic models which are the core of this chapter are presented and described in detail. Classification methods using these nested models are presented in section 5.4, and the procedure to estimate parameters for a nested model from training data is detailed in section 5.5. These probability models are evaluated in three sections: A report on model estimation on real data is given in section 5.7. In section 5.8 I reproduce, because of

their practical relevance to this thesis, the results of [TSB05]. A thorough evaluation of these flat and nested classification methods is given in section 5.9. Finally, the chapter ends with some concluding remarks in section 5.10.

5.2 Flat models

5.2.1 Pixel classification

In the classical setting for image classification, there is an image I , defined over a domain \mathbf{S} , the *site domain*, which is usually a rectangular subset of \mathbb{Z}^2 , such as $\{0, \dots, w-1\} \times \{0, \dots, h-1\}$. For images with d channels, each pixel in that image has values in $\mathbf{D} = \{0, \dots, 255\}^d$, the *data domain*. The pixel at coordinates $s \in \mathbf{S}$ has values $I(s)$, and $I : \mathbf{S} \rightarrow \mathbf{D}$. Let us represent the value of the k -th channel of a pixel at s as $I_k(s)$.

Pixel values are determined by the terrain objects occupying the corresponding region in the imaged land area. What a “terrain object” is depends on the application, but typically it includes concepts such as fields, forests, rivers, lakes, rooftops, or roads. For each application, a set of *terrain classes* $\Omega = \{\omega_1, \omega_2, \dots, \omega_c\}$ is defined, and each terrain object is assumed to belong to one class in Ω (a special “other” class can be defined to group all otherwise unclassified objects). For example, a class set could contain a *forest* class, a *grassland* class, a *water* class, and an *other* class.

For images of sufficiently high spatial resolution, it may be assumed that each pixel contains information from a single terrain object. In that case, we define a *class map*

$$c : \mathbf{S} \rightarrow \Omega \quad (5.1)$$

(with, for all s , $\sum_{\omega \in \Omega} c(s, \omega) = 1$) which gives, for each pixel, the terrain class of its corresponding terrain object. In images of lower spatial resolution, the terrain area that corresponds to a single image pixel may be large enough that several terrain objects contribute to that pixel’s value. In that case, mixture models are typically used to obtain fractional class memberships, and the class map is instead a function

$$c : \mathbf{S} \times \Omega \rightarrow [0, 1]. \quad (5.2)$$

The model described in this chapter is to be used for non-mixture applications, where the class map follows equation (5.1).

The goal of land cover classification systems is to obtain a class map c given an image I and appropriate models describing terrain behaviour and characteristics.

The classical approach is to suppose that the classification of each pixel is independent of the other pixels. Each pixel is then classified depending only on its value, in what is called *per-pixel classification*.

Classification approaches can also be described as *generative* and *discriminative*. In a generative approach, a mathematical description is made of each terrain class, which attempts to reproduce the behaviour of the terrain class as faithfully as possible. Each pixel or region to be classified is compared to these models, and the best model is selected. In a discriminative approach simpler descriptions are made that do not faithfully reproduce the terrain behaviour, but that are good enough for classification; these discriminative descriptions typically involve a partition of the data domain. See figure 5.2 for a comparison of generative and discriminative models for the same data.

Generative models work with partially-labelled data, and can more easily model user-defined invariants —we will present in section 5.3 generative models which explicitly decouple the

variant and invariant part of data. However, they do not scale well for a large number of classes or a large number of invariant transformations to model, and effort is wasted in modelling variations which are not important for classification. Discriminative models, on the other hand, use model flexibility and descriptive power only where it is necessary for classification, and, once trained, are fast. However, they interpolate between training data, which may not be good in some cases, and do not easily handle compositionality and the separation of variants and invariants [Bis04].

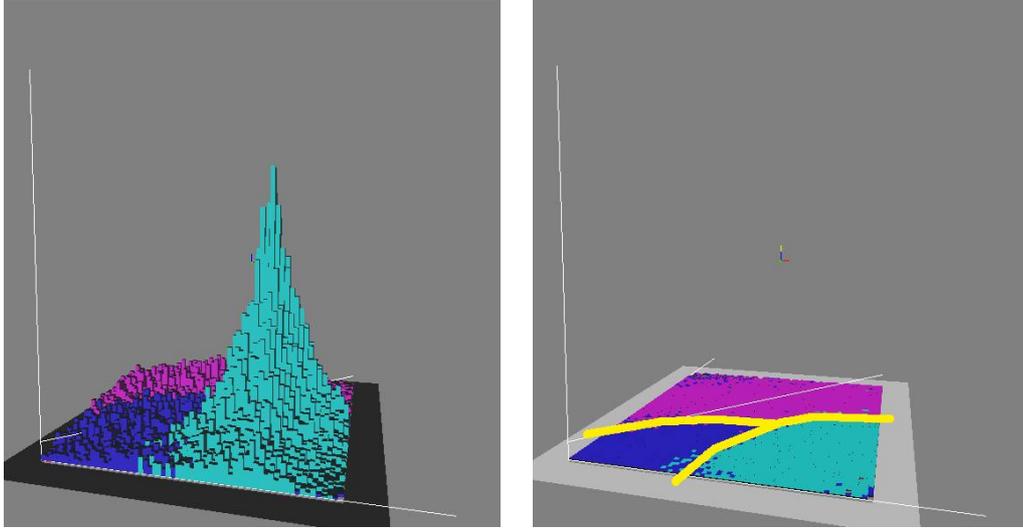


Figure 5.2: Comparison of generative and discriminative models for the same data. Left: a generative model for a three-class, two-dimensional system. The radiometry of pixels in each class roughly follows a Gaussian distribution, each class with different means and covariance matrices. Right: a discriminative model for the same system. The discriminative model contains only the thresholds (in yellow) needed to determine the most likely class of an unclassified pixel, given its radiometry; the actual radiometry distributions are not needed for discrimination, and are not included in the model.

The classical per-pixel, generative, setting associates with each class ω a random variable V_ω with probability function $v_\omega : \mathbf{D} \rightarrow [0, 1]$, that is, $v_\omega(x) := p(I(s) = x | c(s) = \omega)$ (since \mathbf{D} is finite, V_ω is a discrete random variable, so $\sum_{x \in \mathbf{D}} v_\omega(x) = 1$). Then, one solves the classification problem by choosing for each pixel the most probable class, that with highest probability. The “most probable” class map \hat{c} is given by

$$\hat{c} = \operatorname{argmax}_{c \in \Omega^S} p(c | I). \quad (5.3)$$

Note that the set of maps from A to B , usually denoted by $(A \rightarrow B)$ or B^A , is isomorphic to $B^{\operatorname{card} A}$ for a finite set A , so there is no formal problem in assigning a probability to mappings I or c as a whole, as in $p(c | I)$. If we assume that the classes of each pixel are independent, that is, that $p(\bigcap_s c(s) | I) = \prod_s p(c(s) | I)$, we can obtain \hat{c} by decomposing it as

$$\begin{aligned} \hat{c}(s) &= \operatorname{argmax}_{\omega \in \Omega} p(c(s) = \omega | I) = \operatorname{argmax}_{\omega \in \Omega} \frac{p(I | c(s) = \omega) p(c(s) = \omega)}{p(I)} \\ &= \operatorname{argmax}_{\omega \in \Omega} p(I | c(s) = \omega) p(c(s) = \omega), \end{aligned}$$

where we have used the fact that $\operatorname{argmax}_x f(x)g = \operatorname{argmax}_x f(x)$ if g does not depend on x and $g > 0$. By the assumption of pixel independence (that is, the observation at a pixel is independent on the classes of the other pixels), we obtain

$$\hat{c}(s) = \operatorname{argmax}_{\omega \in \Omega} p(I(s) | c(s) = \omega) p(c(s) = \omega) = \operatorname{argmax}_{\omega \in \Omega} v_\omega(I(s)) p(c(s) = \omega).$$

Indeed,

$$p(I | c(s) = \omega) = p(I(s) | c(s) = \omega) \cdot \prod_{s' \neq s} p(I(s') | c(s) = \omega) = p(I(s) | c(s) = \omega) \cdot \prod_{s' \neq s} p(I(s')).$$

Then, with $p(\omega)$ being the *a priori* probability of a pixel belonging to class ω , we have

$$\hat{c}(s) = \operatorname{argmax}_{\omega \in \Omega} v_\omega(I(s)) p(\omega), \quad (5.4)$$

which completes a traditional MAP (*maximum a posteriori*) setting.

This setting is the most commonly used framework for classifying parts of images, such as in land use or land cover classification problems. It is not always made explicit. Other generative and discriminative approaches such as k -means, neural networks, or SVM (support vector machines) operating on pixel data can be seen as equivalent, in a way, to this independent-pixels MAP setting, sometimes with implicit assumptions about the distributions v_ω . The literally thousands of papers devoted to per-pixel classification mostly deal with how to produce more accurate models—either generative or discriminative—from data with certain properties.

To facilitate the description of several combinations of classifiers in the evaluation sections, sections 5.7–5.9, let's rewrite $\hat{c}(s)$ as

$$\hat{c}(s) = \operatorname{argmax}_{\omega \in \Omega} K_{\text{P:MAP}}(\omega, s), \quad (5.5)$$

$$K_{\text{P:MAP}}(\omega, s) = v_\omega(I(s)) p(\omega). \quad (5.6)$$

An alternative to the MAP approach, the *maximum likelihood* (ML) approach does not take into account the prior probabilities of each class, with

$$\hat{c} = \operatorname{argmax}_{c \in \Omega^s} p(I | c), \quad (5.7)$$

which, with a similar derivation, gives

$$\hat{c}(s) = \operatorname{argmax}_{\omega \in \Omega} K_{\text{P:ML}}(\omega, s), \quad (5.8)$$

$$K_{\text{P:ML}}(\omega, s) = v_\omega(I(s)). \quad (5.9)$$

In both the MAP and the ML approach, we can use as confidence measure $\max_{\omega \in \Omega} K(\omega, s)$.

5.2.2 Flat region classification

However, in the per-pixel setting it is assumed that the classifications of pixels are independent of each other. This is obviously false in most images, in particular for images used in land cover classification: the class of a pixel is usually the same as that of the surrounding pixels, and more complex and long-range relationships exist. This problem, which ultimately is the cause for the salt-and-pepper noise usually found in classification results, has long been noticed, and is usually solved, after the classification process itself, by running a regularization

algorithm on the class map \hat{c} —relaxation labelling [RHZ76, Wan98], median filtering, ad-hoc methods [TSL03], or many others—, in order to try to attain a higher degree of spatial coherence. This had given mixed results.

A better mathematically justified regularization is obtained by the use of Markov fields (MRF) for classification [DC04, GG84]; the usage of Markov fields is usually restricted to obtaining a local regularity in the classification—in particular, a priori knowledge of homogeneous regions is not taken into account— although these models are in theory capable of describing much more complex dependencies. However, MRF-based classification is notoriously slow.

Finally, another simple approach to contextual classification is to define neighbourhoods of N pixels (with d -dimensional values) and to classify not the values of individual pixels but the $N \cdot d$ -dimensional values of each neighbourhood. It can be argued that classifying an image containing textural features instead of the original radiometry data is a variant of this form of contextual classification.

In some image classification systems, there is *a priori* knowledge about *image regions*, sets of pixels (usually connected) which belong to the same terrain class. This is the case, for example, for the land use analysis system described here. In this system, cadastre regions are registered to the image (chapter 4, and [TS04a, TSPD04]) so that their edges more closely match image edges. Most of the resulting regions are homogeneous, that is, all pixels in a region belong to the same terrain class. The remaining regions are heterogeneous because of a missing boundary in the cadastral graph—missing in the sense of a field boundary not existing in the cadastral map; farmers are free to divide their parcels of land into several plots if they wish—and should be easily identified and further subdivided. This may be better than the “blind” contextualisation methods described above in that external knowledge, not derived from the images themselves, can be incorporated into this image partition.

Alternatively—for example, when cadastre data is not available—, a partition can be obtained by simply running a segmentation algorithm (such as that of chapter 3) configured to give a “fine” segmentation, one with small regions; there is a risk of obtaining biased results in the classification confidence coefficients if the segmentation uses the same image channels as the classification, which is why using the cadastre is a better option.

If a partition of the image into regions is available, whether it is a registered cadastre or a fine segmentation, each region can be classified as a whole. In contrast to the approach outlined in section 5.2.1, this is called *per-region classification*. A common per-region classification method consists in first performing a per-pixel classification and then, for each region, selecting the majority class [AA04, DWC04]. Other authors [KSF05] compute, for each region, a set of global descriptors such as the average intensity value, the region area, or the standard deviation of the red channel, and use these descriptors for classification. In [TSB05] we presented several classification methods that go beyond the majority vote of [AA04, DWC04] and the simple aggregate descriptors of [KSF05] by using the sample distribution of pixel values in each region. It takes advantage of this a priori knowledge to obtain improved classification results as well as a confidence coefficient indicating the reliability of the classification given to each region. In contrast to the nested model described in section 5.3, we will name these models *flat*, *non-nested*, or *simple*.

The approach of [TSB05] is described in this section. The mathematical formulation given here is more detailed than the one in that paper, and some additional models will be introduced.

Let $\mathbf{R} = \{R_1, \dots, R_n\}$ be the set of regions that the image is partitioned into. Each region R_i is described by the set of image coordinates of its pixels, $R_i \in 2^S$, and these regions are disjoint, $i \neq j \Rightarrow R_i \cap R_j = \emptyset$. In the remainder of the chapter we will assume that the class of a region is independent of the other regions, and will focus on the classification of a single region R

containing N pixels of coordinates

$$R = \{r_1, r_2, \dots, r_N\},$$

where $r_k \in \mathbf{S}$. Let $I(R)$ be the sequence of pixel values —radiometry, transformed colour channels, and texture descriptors— of all pixels in region R taken in arbitrary order, for example,

$$I(R) = (x_1, x_2, \dots, x_N), \quad x_n = I(r_n).$$

Per-region flat classification involves two steps, training and classification.

In the training step, which is to be performed only once for each different type of landscape or cartographic application, for each terrain class a model —the parameters for a random variable— is estimated. This involves, first, manually defining, in a training image, a set of polygons for each terrain class; next, constructing a histogram for the radiometries, or other input features such as texture or derived colour channels, of pixels in a class; then, estimating parameters for several random variables —Gaussian, uniform, Laplacian, ...— using the radiometry histograms; and finally, selecting the random variable that best fits the training data. Generalization occurs at this point, by preferring a worse-fitting but simpler model over a more complex, better-fitting one. This is described in section 5.5.1. Let V_{ω_i} be the selected random variable for class ω_i , and v_{ω_i} its probability function. In addition, a prior probability for class ω_i , $p(\omega_i)$, is calculated, as the ratio of training pixels of that class to the total number of training pixels —region size is assumed not to depend on terrain class.

In the classification step, which is to be performed for every new image, each region in the image to be classified is processed in turn. The histogram of the region is constructed, and compared to the probability models for each terrain class. The most probable class, or that with the probability model closest to the region's histogram, is selected for that region. In [TSB05] we proposed four ways of selecting the best class for region R , $\hat{c}(R)$, and an indicator of the confidence in that classification, $\hat{\kappa}(R)$. Including some additional methods, I propose the following:

Per-region maximum a posteriori (MAP) This approach assumes that all pixels in a region belong to the same class, and selects the class with maximum *a posteriori* probability,

$$\hat{c} = \operatorname{argmax}_{c \in \Omega^R} p(c | I). \quad (5.10)$$

If we assume that the classes of each region are independent, that is, $p(\cap_{\rho} c(\rho) | I) = \prod_{\rho} p(c(\rho) | I)$, we can obtain \hat{c} by decomposition, as

$$\begin{aligned} \hat{c}(R) &= \operatorname{argmax}_{\omega \in \Omega} p(c(R) = \omega | I) \\ &= \operatorname{argmax}_{\omega \in \Omega} \frac{p(I | c(R) = \omega) p(c(R) = \omega)}{p(I)} \\ &= \operatorname{argmax}_{\omega \in \Omega} p(I | c(R) = \omega) p(c(R) = \omega) \\ &= \operatorname{argmax}_{\omega \in \Omega} p(\cap_{s \in R} I(s) \cap \cap_{s \notin R} I(s) | c(R) = \omega) p(c(R) = \omega). \end{aligned}$$

Given that the class of a region has no effect on the observations of pixels outside the region, that is, $p(\cap_{s \notin R} I(s) | c(R) = \omega) = p(\cap_{s \notin R} I(s))$, this factor does not affect the result $\operatorname{argmax}_{\omega \in \Omega}$, we can write

$$\begin{aligned}
\hat{c}(R) &= \operatorname{argmax}_{\omega \in \Omega} p(\cap_{s \in R} I(s) | c(R) = \omega) p(c(R) = \omega) \\
&= \operatorname{argmax}_{\omega \in \Omega} p(c(R) = \omega) \prod_{s \in R} p(I(s) | c(R) = \omega) \\
&= \operatorname{argmax}_{\omega \in \Omega} p(c(R) = \omega) \prod_{s \in R} v_{\omega}(I(s)) = \\
&= \operatorname{argmax}_{\omega \in \Omega} p(\omega) \prod_{s \in R} v_{\omega}(I(s)), \tag{5.11}
\end{aligned}$$

or, in the alternate notation,

$$\hat{c}(R) = \operatorname{argmax}_{\omega \in \Omega} K_{S:\text{MAP}}(\omega, R), \quad K_{S:\text{MAP}}(\omega, R) = p(\omega) \prod_{s \in R} v_{\omega}(I(s)). \tag{5.12}$$

The following confidence coefficient is proposed in [TSB05]:

$$\hat{\kappa}(R) = \log \max_{\omega \in \Omega} p(\omega) \left(\prod_{s \in R} v_{\omega}(I(s)) \right)^{1/|R|}, \tag{5.13}$$

that is,

$$\hat{\kappa}(R) = \log \max_{\omega \in \Omega} K_{S:\text{MAPn}}(\omega, R), \quad K_{S:\text{MAPn}}(\omega, R) = p(\omega) \left(\prod_{s \in R} v_{\omega}(I(s)) \right)^{1/|R|}. \tag{5.14}$$

The $1/|R|$ exponent is necessary to make the confidence coefficient independent of region size. As shown in [TSB05], this gives a more meaningful confidence coefficient. However, in the evaluation of this chapter I also tested the following combination:

$$\hat{c}(R) = \operatorname{argmax}_{\omega \in \Omega} K_{S:\text{MAP}}(\omega, R) \quad \hat{\kappa}(R) = \log \max_{\omega \in \Omega} K_{S:\text{MAP}}(\omega, R), \tag{5.15}$$

$$\hat{c}(R) = \operatorname{argmax}_{\omega \in \Omega} K_{S:\text{MAPn}}(\omega, R) \quad \hat{\kappa}(R) = \log \max_{\omega \in \Omega} K_{S:\text{MAPn}}(\omega, R). \tag{5.16}$$

The first combination uses the true MAP probability also as a confidence measure —except for a constant factor. The second combination attempts to solve the problem that, even though the MAP approach is supposed to take into account class prior probabilities (by including a $p(\omega)$ factor), this factor is dwarfed by the product of $|R|$ pixel probabilities; by raising the product to $1/|R|$ also in $\hat{c}(R)$ we give similar weights to both probabilities.

Assuming regions are homogeneous and of a modelled class If we assume that the class of a region is one of the classes in Ω , we can obtain similar but slightly different results. This assumption can be broken down into two parts: That regions are indeed homogeneous, and that there is no terrain class not included in Ω . Then, we can calculate the probability that a region R is of class $c(R)$ given the image I , and given that we *know* that this class is in $c(R) \in \Omega$, as follows:

$$p(c(R) = \omega | I \cap c(R) \in \Omega) = \frac{p(c(R) = \omega \cap c(R) \in \Omega | I)}{p(c(R) \in \Omega | I)}$$

which, since $c(R) = \omega \Rightarrow c(R) \in \Omega$, gives

$$= \frac{p(c(R) = \omega | I)}{\sum_{\pi \in \Omega} p(c(R) = \pi | I)}$$

which, by Bayes' rule, can be written as

$$= \frac{p(I | c(R) = \omega) p(c(R) = \omega) p(I)^{-1}}{\sum_{\pi \in \Omega} p(I | c(R) = \pi) p(c(R) = \pi) p(I)^{-1}}.$$

We separate the image I into the region of interest R and the rest R^c , assume the independence of pixel values in R and those in R^c , conditional on the class of R , and note that pixel values in R^c are independent on the class of R ,

$$\begin{aligned} &= \frac{p(R | c(R) = \omega) p(R^c | c(R) = \omega) p(c(R) = \omega) p(I)^{-1}}{\sum_{\pi \in \Omega} p(R | c(R) = \pi) p(R^c | c(R) = \pi) p(c(R) = \pi) p(I)^{-1}} \\ &= \frac{p(R | c(R) = \omega) p(R^c) p(c(R) = \omega) p(I)^{-1}}{\sum_{\pi \in \Omega} p(R | c(R) = \pi) p(R^c) p(c(R) = \pi) p(I)^{-1}} \\ &= \frac{p(R | c(R) = \omega) p(c(R) = \omega)}{\sum_{\pi \in \Omega} p(R | c(R) = \pi) p(c(R) = \pi)}. \end{aligned}$$

As before, we assume that pixel values in a region are independent, conditional on the region's class,

$$\begin{aligned} &= \frac{p(c(R) = \omega) p(\cap_{s \in R} I(s) | c(R) = \omega)}{\sum_{\pi \in \Omega} p(c(R) = \pi) p(\cap_{s \in R} I(s) | c(R) = \pi)} \\ &= \frac{p_\omega \prod_{s \in R} p(I(s) | c(R) = \omega)}{\sum_{\pi \in \Omega} p_\pi \prod_{s \in R} p(I(s) | c(R) = \pi)} \\ &= \frac{p_\omega \prod_{s \in R} v_\omega(I(s))}{\sum_{\pi \in \Omega} p_\pi \prod_{s \in R} v_\pi(I(s))}. \end{aligned} \quad (5.17)$$

We can then choose the most probable class for a region

$$\hat{c}(R) = \operatorname{argmax}_{\omega \in \Omega} p(c(R) = \omega | I \cap c(R) \in \Omega) = \operatorname{argmax}_{\omega \in \Omega} \frac{p_\omega \prod_{s \in R} v_\omega(I(s))}{\sum_{\pi \in \Omega} p_\pi \prod_{s \in R} v_\pi(I(s))}. \quad (5.18)$$

And we can use as a confidence measure the probability of the region being of the selected class, assuming again that the real class is in Ω . This gives

$$\hat{c}(R) = \log \max_{\omega \in \Omega} p(c(R) = \omega | I \cap c(R) \in \Omega) = \log \max_{\omega \in \Omega} \frac{p_\omega \prod_{s \in R} v_\omega(I(s))}{\sum_{\pi \in \Omega} p_\pi \prod_{s \in R} v_\pi(I(s))}, \quad (5.19)$$

that is,

$$\hat{c}(R) = \operatorname{argmax}_{\omega \in \Omega} K_{S:\text{MAP}_0}(\omega, R), \quad \hat{c}(R) = \log \max_{\omega \in \Omega} K_{S:\text{MAP}_0}(\omega, R), \quad (5.20)$$

with

$$K_{S:\text{MAP}_0}(\omega, R) = \frac{p_\omega \prod_{s \in R} v_\omega(I(s))}{\sum_{\pi \in \Omega} p_\pi \prod_{s \in R} v_\pi(I(s))}. \quad (5.21)$$

Per-region maximum likelihood (ML) There is open debate on whether the MAP approach is better than the maximum likelihood (ML) approach, or it is the other way around. Briefly, MAP gives less probability to classes found rarely in the training data. ML gives equal chances to all classes, which is not optimal for risk minimization but may be desired in some cases. For an ML approach we start from the classification

$$\hat{c} = \operatorname{argmax}_{c \in \Omega^R} p(I|c)$$

which, following a similar derivation to that of equation (5.11), gives

$$\hat{c}(R) = \operatorname{argmax}_{\omega \in \Omega} \prod_{s \in R} v_{\omega}(I(s)). \quad (5.22)$$

Correspondingly, the following confidence coefficient can be used:

$$\hat{\kappa}(R) = \log \max_{\omega \in \Omega} \left(\prod_{s \in R} v_{\omega}(I(s)) \right)^{1/|R|}. \quad (5.23)$$

Equivalently,

$$\hat{c}(R) = \operatorname{argmax}_{\omega \in \Omega} K_{S:ML}(\omega, R), \quad \hat{\kappa}(R) = \log \max_{\omega \in \Omega} K_{S:MLn}(\omega, R) \quad (5.24)$$

with

$$K_{S:ML}(\omega, R) = \prod_{s \in R} v_{\omega}(I(s)), \quad (5.25)$$

$$K_{S:MLn}(\omega, R) = \left(\prod_{s \in R} v_{\omega}(I(s)) \right)^{1/|R|}. \quad (5.26)$$

As with the MAP approach, we also tested the “true” ML approach, where the correcting $1/|R|$ exponent is not used,

$$\hat{c}(R) = \operatorname{argmax}_{\omega \in \Omega} K_{S:ML}(\omega, R), \quad \hat{\kappa}(R) = \log \max_{\omega \in \Omega} K_{S:ML}(\omega, R). \quad (5.27)$$

Kullback-Leibler distribution divergence (KL) Instead of calculating the probability that the pixels in a region come from a certain random variable, as we have done in the previous paragraphs, we could directly compare the distributions of pixel values in the region being classified to the probability functions of the terrain models, and choose the terrain class with most similar distribution.

For this classification method and the next, the value histogram for the region of interest will be needed. Let us partition the data domain \mathbf{D} into equally-shaped histogram bins, $\{D_i\}_i$, with $\cup_i D_i = \mathbf{D}$ and $\forall i \neq j. D_i \cap D_j = \emptyset$. Let us construct a relative histogram h from a sequence of samples $X = (x_1, x_2, \dots, x_N)$, $x_i \in \mathbf{D}$, by counting the number of values in X that belong to the i -th histogram bin, for each i : $h_i = \operatorname{card}\{j : x_j \in D_i\}/|X|$.

In the non-parametric model estimation of section 5.5.1, the choice of the partition of \mathbf{D} will have an impact on the amount of generalization. This is not the case here, where this choice only impacts the balance between domain and frequency resolution.

In this implementation, d -dimensional square bins of side 8 were chosen.

Distribution distance-based classification algorithms depend on the definition of $d(X, V)$, an indication of the dissimilarity between a sample X and a random variable V . For a given

dissimilarity indicator, the selected class for region R is the one closest to the sample sequence $I(R)$, and we propose to use as confidence coefficient the coefficient itself:

$$\hat{c}(R) = \operatorname{argmax}_{\omega \in \Omega} (-d(I(R), V_\omega)), \hat{k}(R) = \max_{\omega \in \Omega} (-d(I(R), V_\omega)). \quad (5.28)$$

One such coefficient is the Kullback-Leibler divergence, also called the relative entropy. The Kullback-Leibler divergence d between a sample X and a discrete random variable V of probability function v is

$$d_{\text{KL}}(X, V) = 2 \sum_i (h_i \log h_i - h_i \log p(V \in D_i)), \quad (5.29)$$

where, of course, $p(V \in D_i) = \int_{z \in D_i} v(z) dz$, from which we obtain

$$\hat{c}(R) = \operatorname{argmax}_{\omega \in \Omega} (-d_{\text{KL}}(I(R), V_\omega)), \quad (5.30)$$

$$\hat{k}(R) = \max_{\omega \in \Omega} (-d_{\text{KL}}(I(R), V_\omega)). \quad (5.31)$$

There is also a symmetrical version of the Kullback-Leibler divergence, which can also be used:

$$d_{\text{KLs}}(X, V) = \sum_i (h_i \log h_i - h_i \log p(V \in D_i)) + \sum_i (p(V \in D_i) \log p(V \in D_i) - p(V \in D_i) \log h_i). \quad (5.32)$$

Chi-square distribution dissimilarity coefficient (χ^2/N) Another dissimilarity coefficient is the χ^2 statistic, commonly used in the χ^2 test [Cra46],

$$\chi^2(X, V) = N \sum_i \frac{(h_i - p(V \in D_i))^2}{p(V \in D_i)}, \quad (5.33)$$

omitting from the sum any term with $h_i = p(V \in D_i) = 0$. However, the χ^2 statistic depends linearly on N , the number of pixels in the region. While this does not affect the choice of class for the region, it may affect the confidence coefficients. If we refuse to classify regions where the confidence coefficient falls under a certain threshold, we might keep an incorrect classification for a small region while rejecting a correct classification for a large region. The problem is avoided as in equation (5.13) by normalizing by region size, which gives

$$\hat{c}(R) = \operatorname{argmax}_{\omega \in \Omega} (-\log \chi^2(I(R), V_\omega)), \quad (5.34)$$

$$\hat{k}(R) = \max_{\omega \in \Omega} \left(-\log \frac{\chi^2(I(R), V_\omega)}{|R|} \right). \quad (5.35)$$

As before, we also tested a variant without the $1/|R|$ correcting term,

$$\hat{c}(R) = \operatorname{argmax}_{\omega \in \Omega} (-\log \chi^2(I(R), V_\omega)), \quad (5.36)$$

$$\hat{k}(R) = \max_{\omega \in \Omega} (-\log \chi^2(I(R), V_\omega)). \quad (5.37)$$

Majority vote For comparison, the majority vote technique of [AA04, DWC04] will also be tested. In this method, a per-pixel classification such as a per-pixel MAP is performed, and then, for each region, the most frequent per-pixel class is selected as the per-region class:

$$\hat{c}_{\text{Maj}}(R) = \operatorname{argmax}_{\omega \in \Omega} \operatorname{card} \left\{ s \in R : \omega = \operatorname{argmax}_{v \in \Omega} K_{\text{P:MAP}}(v, s) \right\}, \quad (5.38)$$

$$\hat{k}_{\text{Maj}}(R) = \frac{1}{|R|} \sum_{s \in R} K_{\text{P:MAP}}(\hat{c}_{\text{Maj}}(R), s). \quad (5.39)$$

5.3 Nested models

The flat models described in the previous sections assume that there is one random variable for each terrain class, and that pixels belonging to that class are realizations of its random variable. However, terrain belonging to a single class, for example, “forest” terrain, may look differently depending on factors other than land cover, such as tree species, slope, season, weather, or time of the day. We can say that the “forest class” can be subdivided into many subclasses, such as “evergreen forest”, “deciduous forest in winter”, “deciduous forest in summer on a flat surface”, “deciduous forest in summer on a steep slope”, and many others. Although a random variable that models the whole “forest” class can be estimated, perhaps random variables modelling each subclass would represent reality better.

Suppose two regions R_1 and R_2 that belong to the same class but different subclasses, $\omega_{1,1}$ and $\omega_{1,2}$, ω_1 being the common class for both regions. Let us say that $\omega_{1,1}$ is modelled by the random variable $V_{1,1}$, and $\omega_{1,2}$ by $V_{1,2}$, that is, pixels in R_1 are realizations of $V_{1,1}$ and pixels in R_2 are realizations of $V_{1,2}$. Suppose also a synthetic region R_{1+2} some of whose pixels are realizations of $V_{1,1}$ and the rest of $V_{1,2}$. Regions like R_{1+2} do not exist in reality, as regions are required to be homogeneous—and most subclasses, such as “deciduous forest in summer on a steep slope” and “deciduous forest in winter on a flat surface” are incompatible.

A flat per-region classification system like those described in section 5.2.2 trained at the class level would have estimated a random variable V_1 of which pixels in R_1 and R_2 would be realizations. This random variable would, however, better model the synthetic region R_{1+2} than the regions R_1 or R_2 , because a single random variable V_1 must cover the cases covered by both $V_{1,1}$ and $V_{1,2}$. This can be seen in figures 5.3 and 5.4. Figure 5.3 is an example where flat models are good enough. The problem described in this section here corresponds to figure 5.4. Distributions for regions R_1 and R_2 are shown in the top row, and the random variable V_1 is shown in the middle row, right graph. We can see that V_1 does not model correctly R_1 nor R_2 .

The solution is, of course, to train not at the class level but at the subclass level, that is, to estimate one random variable per subclass and not per class. In that way, random variables $V_{1,1}$ and $V_{1,2}$ would be estimated of which pixels in R_1 and R_2 respectively would be realizations. Region R_1 would be correctly classified as subclass $\omega_{1,1}$ and region R_2 as subclass $\omega_{1,2}$. Region R_{1+2} would be classified with a very low confidence coefficient since none of the subclass models correctly fit the data in R_{1+2} .

That is, using subclass classification we would obtain more tightly fitting models, and we could increase the rejection ratio—how many regions or pixels that do not belong to any class used in the training phase are rejected as not classifiable. In some systems—such as the subject of this thesis—it is more important to have a very reliable classification, even if some elements remain unclassified, rather than have a total but unreliable coverage.

The problem is that, in principle, to train for $N \cdot M$ subclasses instead of for N classes we need M times more training data—or, alternatively, we can decide to train with the same amount of

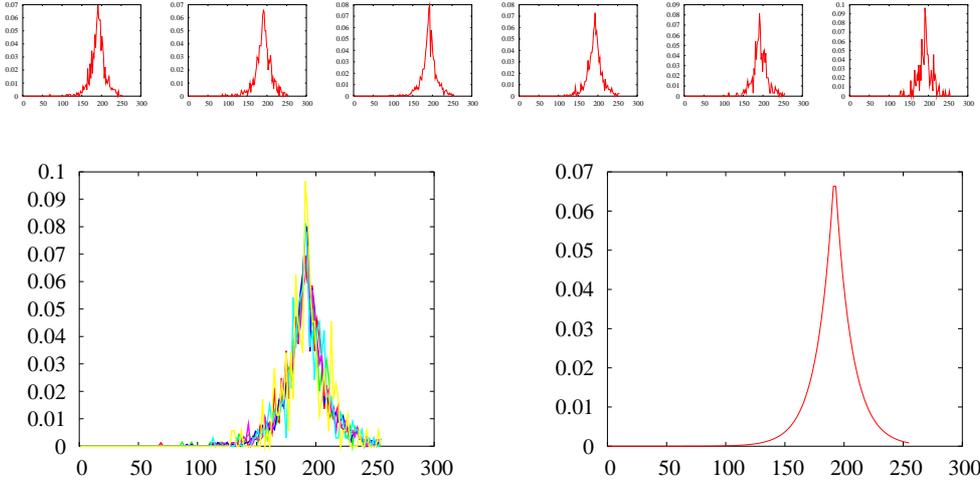


Figure 5.3: Example of terrain class where flat models correctly describe data. Top row: radiometry distributions of six training regions. Bottom left: superposition of these six distributions. Bottom right: distribution of a random variable estimated from these six training regions. We can see that the random variable is a good model for the training regions.

data, which will produce a probability model, and therefore a classification system, of much lower quality. Some classes, furthermore, may theoretically be divided into an infinite number of subclasses; for example, pixels in the “water” class look differently depending on water depth, which is a continuous magnitude.

There is one case in which it may be possible to work at the subclass level with a much smaller increase in the required amount of training data. For that, we need to define the concept of *random variable class*. Let us write \mathcal{V}_B the set of random variables taking values in B . For example, a two-dimensional Gaussian random variable of mean $(5 \ 2)^T$ and covariance matrix $\begin{pmatrix} 2 & 0 \\ 0 & 2 \end{pmatrix}$ is an element of the set $\mathcal{V}_{\mathbb{R}^2}$. A *random variable class* is a function from values in a domain A to random variables taking values in a domain B . For example, if we note $G(x, \Sigma)$ a Gaussian variable of mean vector x and covariance matrix Σ , we can define the class of unidimensional Gaussian random variables of undefined mean and unit variance as

$$H_1 : \mathbb{R} \rightarrow \mathcal{V}_{\mathbb{R}} \\ x \mapsto G(x, (1)).$$

In this example, H_1 is a random variable class, and $H_1(5)$ is a one-dimensional Gaussian random variable (of mean 5 and variance 1). As a more convoluted example —just for illustration— the class of two-dimensional Gaussian random variables whose mean is also the diagonal of their diagonal covariance matrix can be defined as

$$H_2 : \mathbb{R}^2 \rightarrow \mathcal{V}_{\mathbb{R}^2} \\ (x, y) \mapsto G\left(\begin{pmatrix} x \\ y \end{pmatrix}, \begin{pmatrix} x & 0 \\ 0 & y \end{pmatrix}\right).$$

Let us note by $\mathcal{K}_{A \rightarrow B}$ the set of random variable classes from values in A to random variables taking values in B . For the examples above, $H_1 \in \mathcal{K}_{\mathbb{R} \rightarrow \mathbb{R}}$ and $H_2 \in \mathcal{K}_{\mathbb{R}^2 \rightarrow \mathbb{R}^2}$.

Back to the problem of sub-class models, I hypothesize that, in some cases, the random variables $V_{i,j}$ for all subclasses $\omega_{i,j}$ belonging to the same class ω_i are *similar*, and that this similarity can

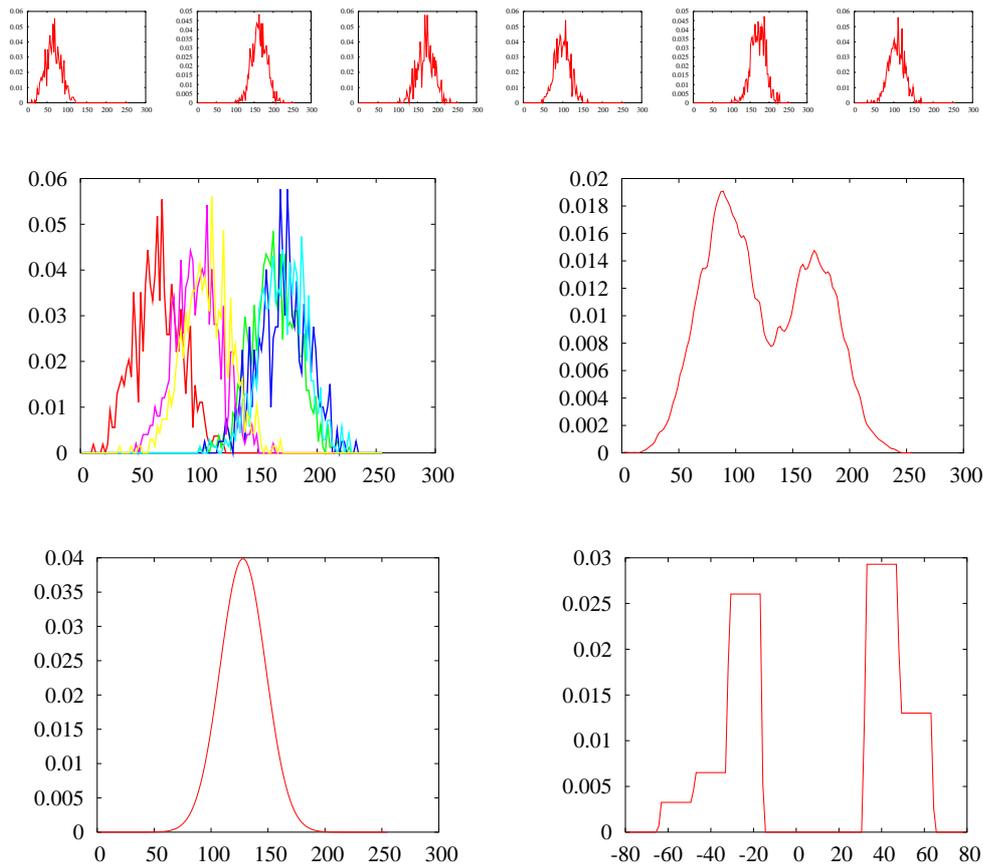


Figure 5.4: Example of terrain class where flat models *do not* correctly describe data. Top row: radiometry distributions of six training regions. Middle left: superposition of these six distributions. Middle right: distribution of a random variable estimated from these six training regions. We can see that the random variable is bimodal whereas none of the training regions is, and has a much wider support than the training regions. The bottom row shows a nested model; each region follows a random variable which has the shape of the bottom-left graph, and a mean drawn from the random law of the bottom-right graph.

be modelled using random variable classes in the following way: For each land cover class ω_i there exists a random variable class Ξ_{ω_i} and a random variable V_{ω_i} . To each region R_k belonging to class ω_i corresponds a value q_k , which is a realization of V_{ω_i} , and a random variable W_k which is the parametrization of Ξ_{ω_i} by q_k , that is, $W_k = \Xi_{\omega_i}(q_k)$. Finally, the pixels in R_k are all realizations of the random variable W_k .

Where this hypothesis is true, we can estimate models at the subclass level with only a small increase in the amount of required training data.

We name this kind of models *nested models*. In section 5.7 I show that some land cover classes do follow this behaviour, and therefore the hypothesis made is justified. The examples given before, where a class was divided into a finite number of subclasses can be treated as a nested model with the random variable V_{ω_i} taking values in a finite set. An example of nested model is given in figure 5.4.

A flat model (section 5.2) for a class ω is defined by the pair (p_ω, V_ω) , where p_ω is the class per-region *a priori* probability and V_ω the random law followed by the pixels belonging to the class. In contrast, a nested model is defined by the triplet

$$(p_\omega, \Xi_\omega, V_\omega), \quad (5.40)$$

that is, the class per-region *a priori* probability (the probability that a region belongs to a class, not given the image contents), the random variable class for this terrain class, and the random law governing the parameters used to obtain instances of Ξ_ω . The probability density function of V_ω is v_ω and the probability function that of $\Xi_\omega(x)$ is $\xi_\omega(x)$ (note that $\xi_\omega(x)$ is itself a function; the value of that function at point y is $\xi_\omega(x)(y)$). Let us name Q_ω the set of the values taken by V_ω . We have

$$V_\omega \in \mathcal{V}_{Q_\omega}, \quad (5.41)$$

$$\Xi_\omega \in \mathcal{K}_{Q_\omega \rightarrow \mathbf{D}}, \quad (5.42)$$

$$\Xi_\omega(q) \in \mathcal{V}_{\mathbf{D}}. \quad (5.43)$$

Since \mathbf{D} is a finite set, $\Xi_\omega(q)$ is a discrete random variable and we have $\sum_{x \in \mathbf{D}} \xi_\omega(q)(x) = 1$. The set Q_ω may be finite or infinite, and so V_ω will be a discrete random variable (with $\sum_{q \in Q_\omega} v_\omega(q) = 1$) or a continuous random variable (with $\int_{q \in Q_\omega} v_\omega(q) dq = 1$), respectively.

A nested probability model is not the same as a composition of random variables. Given a terrain class described by (p, Ξ, V) , the random variable $W \in \mathcal{V}_{\mathbf{D}}$ governing the values of pixels in a region R of that class is *not* the composed random variable $(\Xi \circ V) \in \mathcal{V}_{\mathbf{D}}$, since the same instance of Ξ is used for all pixels in a region.

This model—like the previous per-pixel and flat per-region—does not take into account spatial regularities and relations among pixels in the same region: the position of pixels in a region is ignored by the model. Such spatial phenomena—texture, in other words—can be brought into the model by using textural features in addition to—or instead of—the raw radiometry channels.

Nested models are to be compared to the proposals of Kumar, Loui, and Hebert [KLH03] and Marroquin, Arce Santana, and Botello [MASB03]. Kumar *et al.* [KLH03] describe the problem encountered in classification by generative models when, in the image to be classified, the radiometries of the pixels for a certain class have a much smaller support than those in the training image. This leads to assigning an incorrect class membership probability to pixels in the input image. They solve the problem by spatially clustering pixels of similar radiometry in the input image, selecting a class for each cluster, constraining the probability model for the class to the cluster's support, and obtaining the class membership probability for pixels in the

cluster using this constrained class model. This is not unlike the problem which motivates the use of nested probability models here, although we believe nested models to be much more flexible —although possibly more computationally expensive.

Marroquin *et al.* [MASB03] propose a model whereby each region in a segmentation follows a certain random law parametrized by a region-specific value. The underlying random law class is the same for all regions. Furthermore, regions are not defined a priori, but are the result of the segmentation. The segmentation is performed using Markov Random Field techniques. They suggest using this method for generic unsupervised segmentation. In contrast, our technique is geared towards classification, is supervised, the image partition is defined a priori, and each semantic class may use different random law classes. The class parameters also follow a class-dependent random law, unlike in [MASB03]. Furthermore, we show —through BIC values in model estimation— that we are using this to model radiometric and textural behaviours that actually follow our compound model; in [MASB03] they use it as a generic segmentation tool for images which do not necessarily behave in this way, and the nestedness is used only as a tool to handle heavy noise in images.

5.4 Classification of nested models

After defining, in section 5.3, a new kind of probability models for per-region classification that we have named *nested models*, in this chapter several classification algorithms that use these models will be presented.

As in section 5.2.2, classification is performed region by region in the nested model framework, and these nested classification methods provide not only the most probable class for each region, \hat{c} , but also an indication of the classifier confidence in that choice, $\hat{\kappa}$.

Again, let $\mathbf{R} = \{R_1, \dots, R_n\}$ be the set of disjoint image regions to be classified, each region described as a set of image coordinates, $R_i \in 2^{\mathbf{S}}$, with $i \neq j \Rightarrow r_i \cap r_j = \emptyset$. We again assume that the class of a region is independent of the other regions, and focus on the classification of a single region R containing N pixels of coordinates $R = \{r_1, r_2, \dots, r_N\}$, where $r_k \in \mathbf{S}$, and pixel value sequence $I(R)$.

The set of terrain classes is $\Omega = \{\omega_1, \omega_2, \dots, \omega_c\}$, and for each terrain class ω there is a nested model $(p_\omega, \Xi_\omega, V_\omega)$, with $\Xi_\omega \in \mathcal{K}_{\mathcal{Q}_\omega \rightarrow \mathcal{D}}$, $V_\omega \in \mathcal{V}_{\mathcal{Q}_\omega}$, and with $\xi_\omega(q)$ and v_ω the probability distributions of $\Xi_\omega(q)$ and V_ω respectively.

5.4.1 Classical probabilistic classification

In a *maximum a posteriori* (MAP) setting, the most probable region classification is

$$\hat{c} = \operatorname{argmax}_{c \in \Omega^{\mathbf{R}}} p(c | I). \quad (5.44)$$

Since we assume that the classes of each region are independent, that is, $p(\cap_\rho c(\rho) | I) = \prod_\rho p(c(\rho) | I)$, \hat{c} can be obtained by decomposition as follows:

$$\begin{aligned} \hat{c}(R) &= \operatorname{argmax}_{\omega \in \Omega} p(c(R) = \omega | I) \\ &= \operatorname{argmax}_{\omega \in \Omega} \frac{p(I | c(R) = \omega) p(c(R) = \omega)}{p(I)} \end{aligned}$$

which, since $p(I)$ does not affect the argmax operator, can be rewritten as

$$= \operatorname{argmax}_{\omega \in \Omega} p(I | c(R) = \omega) p(c(R) = \omega)$$

and, because of region independence, as

$$\begin{aligned} &= \operatorname{argmax}_{\omega \in \Omega} p_{\omega} p(I | c(R) = \omega) \\ &= \operatorname{argmax}_{\omega \in \Omega} p_{\omega} p(\cap_{s \in S} I(s) | c(R) = \omega) \\ &= \operatorname{argmax}_{\omega \in \Omega} p_{\omega} p(\cap_{s \in R} I(s) \cap \cap_{s \notin R} I(s) | c(R) = \omega). \end{aligned}$$

Given that the class of a region has no effect on the observations of pixels outside the region, that is, $p(\cap_{s \notin R} I(s) | c(R) = \omega) = p(\cap_{s \notin R} I(s))$, this factor does not affect the result $\operatorname{argmax}_{\omega \in \Omega}$, so we can write

$$\begin{aligned} &= \operatorname{argmax}_{\omega \in \Omega} p_{\omega} p(\cap_{s \in R} I(s) | c(R) = \omega) \\ &= \operatorname{argmax}_{\omega \in \Omega} p_{\omega} \int_{q \in Q_{\omega}} p(\cap_{s \in R} I(s) | c(R) = \omega \cap q) v_{\omega}(q) dq \\ &= \operatorname{argmax}_{\omega \in \Omega} p_{\omega} \int_{q \in Q_{\omega}} \prod_{s \in R} p(I(s) | c(R) = \omega \cap q) v_{\omega}(q) dq \end{aligned}$$

and finally

$$\hat{c}(R) = \operatorname{argmax}_{\omega \in \Omega} p_{\omega} \int_{q \in Q_{\omega}} v_{\omega}(q) \prod_{s \in R} \xi_{\omega}(q)(I(s)) dq. \quad (5.45)$$

Again, a similar derivation can be done for a maximum likelihood approach (ML). In that case we start from

$$\hat{c} = \operatorname{argmax}_{c \in \Omega^R} p(I | c), \quad (5.46)$$

which yields

$$\hat{c}(R) = \operatorname{argmax}_{\omega \in \Omega} \int_{q \in Q_{\omega}} v_{\omega}(q) \prod_{s \in R} \xi_{\omega}(q)(I(s)) dq. \quad (5.47)$$

Estimating the certainty of a classification

Once a region R has been classified as belonging to the class $\omega = \hat{c}(R)$, we can estimate how well the pixels in the region fit the probability law for the class to obtain a confidence coefficient $\hat{\kappa}(R)$. We can set a threshold on this confidence value and have the algorithm refuse to classify certain regions—those with low classification confidence—. A probable cause of bad classification would be that the region is not actually homogeneous; in that case the region should be split into homogeneous subregions to be reclassified.

As for the MAP and ML approaches to per-region classification with flat models of section 5.2.2, the MAP or ML probability of a region belonging to the selected class—except, in the MAP

case, for the constant factor $p(I)$ — can be used as a confidence coefficient. For MAP, this would be

$$\hat{\kappa}(R) = \log p(\hat{c}(R) | I) p(I) = \log \max_{\omega \in \Omega} p_{\omega} \int_{q \in Q_{\omega}} v_{\omega}(q) \prod_{s \in R} \xi_{\omega}(q)(I(s)) \, dq, \quad (5.48)$$

and for ML

$$\hat{\kappa}(R) = \log p(I | \hat{c}(R)) = \log \max_{\omega \in \Omega} \int_{q \in Q_{\omega}} v_{\omega}(q) \prod_{s \in R} \xi_{\omega}(q)(I(s)) \, dq. \quad (5.49)$$

Alternately, the following region-size-independent confidence coefficients can be used:

$$\hat{\kappa}(R) = \log p(\hat{c}(R) | I) p(I) = \log \max_{\omega \in \Omega} p_{\omega} \int_{q \in Q_{\omega}} v_{\omega}(q) \left(\prod_{s \in R} \xi_{\omega}(q)(I(s)) \right)^{1/|R|} \, dq, \quad (\text{MAP}) \quad (5.50)$$

$$\hat{\kappa}(R) = \log p(I | \hat{c}(R)) = \log \max_{\omega \in \Omega} \int_{q \in Q_{\omega}} v_{\omega}(q) \left(\prod_{s \in R} \xi_{\omega}(q)(I(s)) \right)^{1/|R|} \, dq. \quad (\text{ML}) \quad (5.51)$$

For the MAP case, it must be noted that since we discarded the term $p(I)$, confidence values for different regions will be comparable only within the same image.

In the alternative notation, if we define

$$K_{C:\text{MAP}}(\omega, R) = p_{\omega} \int_{q \in Q_{\omega}} v_{\omega}(q) \prod_{s \in R} \xi_{\omega}(q)(I(s)) \, dq, \quad (5.52)$$

$$K_{C:\text{ML}}(\omega, R) = \int_{q \in Q_{\omega}} v_{\omega}(q) \prod_{s \in R} \xi_{\omega}(q)(I(s)) \, dq, \quad (5.53)$$

$$K_{C:\text{MAPn}}(\omega, R) = p_{\omega} \int_{q \in Q_{\omega}} v_{\omega}(q) \left(\prod_{s \in R} \xi_{\omega}(q)(I(s)) \right)^{1/|R|} \, dq, \quad (5.54)$$

$$K_{C:\text{MLn}}(\omega, R) = \int_{q \in Q_{\omega}} v_{\omega}(q) \left(\prod_{s \in R} \xi_{\omega}(q)(I(s)) \right)^{1/|R|} \, dq, \quad (5.55)$$

we can note the different combinations of \hat{c} and $\hat{\kappa}$ that we have described as

$$\hat{c}(R) = \operatorname{argmax}_{\omega \in \Omega} K_{C:\text{MAP}}(\omega, R), \quad \hat{\kappa}(R) = \log \max_{\omega \in \Omega} K_{C:\text{MAP}}(\omega, R), \quad (5.56)$$

$$\hat{c}(R) = \operatorname{argmax}_{\omega \in \Omega} K_{C:\text{ML}}(\omega, R), \quad \hat{\kappa}(R) = \log \max_{\omega \in \Omega} K_{C:\text{ML}}(\omega, R), \quad (5.57)$$

$$\hat{c}(R) = \operatorname{argmax}_{\omega \in \Omega} K_{C:\text{MAPn}}(\omega, R), \quad \hat{\kappa}(R) = \log \max_{\omega \in \Omega} K_{C:\text{MAPn}}(\omega, R), \quad (5.58)$$

$$\hat{c}(R) = \operatorname{argmax}_{\omega \in \Omega} K_{C:\text{MLn}}(\omega, R), \quad \hat{\kappa}(R) = \log \max_{\omega \in \Omega} K_{C:\text{MLn}}(\omega, R), \quad (5.59)$$

and, for completeness, add these two combinations:

$$\hat{c}(R) = \operatorname{argmax}_{\omega \in \Omega} K_{C:\text{MAPn}}(\omega, R), \quad \hat{\kappa}(R) = \log \max_{\omega \in \Omega} K_{C:\text{MAPn}}(\omega, R), \quad (5.60)$$

$$\hat{c}(R) = \operatorname{argmax}_{\omega \in \Omega} K_{C:\text{MLn}}(\omega, R), \quad \hat{\kappa}(R) = \log \max_{\omega \in \Omega} K_{C:\text{MLn}}(\omega, R). \quad (5.61)$$

5.4.2 Assuming regions are homogeneous and of a modelled class

For nested models too, we can assume that the class of a region is one of the classes in Ω to obtain similar but slightly different results. Again, this assumption can be broken down into

two parts: That regions are indeed homogeneous, and that there is no terrain class not included in Ω . Then, we can calculate the probability that a region R is of class $c(R)$ given the image I , and given that we *know* that this class is in $c(R) \in \Omega$, as follows:

$$p(c(R) = \omega | I \cap c(R) \in \Omega) = \frac{p(c(R) = \omega \cap c(R) \in \Omega | I)}{p(c(R) \in \Omega | I)}$$

which, since $c(R) = \omega \Rightarrow c(R) \in \Omega$, gives

$$= \frac{p(c(R) = \omega | I)}{\sum_{\pi \in \Omega} p(c(R) = \pi | I)}$$

which, by Bayes' rule, can be written as

$$= \frac{p(I | c(R) = \omega) p(c(R) = \omega) p(I)^{-1}}{\sum_{\pi \in \Omega} p(I | c(R) = \pi) p(c(R) = \pi) p(I)^{-1}}.$$

Separating the image I into the region of interest R and the rest R^c , assuming the independence of pixel values in R and those in R^c , conditional on the class of R , and noting that pixel values in R^c are independent on the class of R , we obtain

$$\begin{aligned} &= \frac{p(R | c(R) = \omega) p(R^c | c(R) = \omega) p(c(R) = \omega) p(I)^{-1}}{\sum_{\pi \in \Omega} p(R | c(R) = \pi) p(R^c | c(R) = \pi) p(c(R) = \pi) p(I)^{-1}} \\ &= \frac{p(R | c(R) = \omega) p(R^c) p(c(R) = \omega) p(I)^{-1}}{\sum_{\pi \in \Omega} p(R | c(R) = \pi) p(R^c) p(c(R) = \pi) p(I)^{-1}} \\ &= \frac{p(R | c(R) = \omega) p(c(R) = \omega)}{\sum_{\pi \in \Omega} p(R | c(R) = \pi) p(c(R) = \pi)} \end{aligned}$$

and, as before, if we assume that pixel values in a region are independent, conditional on the region's class,

$$\begin{aligned} &= \frac{p(c(R) = \omega) p(\cap_{s \in R} I(s) | c(R) = \omega)}{\sum_{\pi \in \Omega} p(c(R) = \pi) p(\cap_{s \in R} I(s) | c(R) = \pi)} \\ &= \frac{p(c(R) = \omega) \int_{q \in Q_\omega} p(q | c(R) = \omega) p(\cap_{s \in R} I(s) | q \cap c(R) = \omega) dq}{\sum_{\pi \in \Omega} p(c(R) = \pi) \int_{q \in Q_\pi} p(q | c(R) = \pi) p(\cap_{s \in R} I(s) | q \cap c(R) = \pi) dq} \\ &= \frac{p_\omega \int_{q \in Q_\omega} v_\omega(q) \prod_{s \in R} p(I(s) | q \cap c(R) = \omega) dq}{\sum_{\pi \in \Omega} p_\pi \int_{q \in Q_\pi} v_\pi(q) \prod_{s \in R} p(I(s) | q \cap c(R) = \pi) dq} \\ &= \frac{p_\omega \int_{q \in Q_\omega} v_\omega(q) \prod_{s \in R} \xi_\omega(q)(I(s)) dq}{\sum_{\pi \in \Omega} p_\pi \int_{q \in Q_\pi} v_\pi(q) \prod_{s \in R} \xi_\pi(q)(I(s)) dq}. \end{aligned} \tag{5.62}$$

We can then choose the most probable class for a region

$$\hat{c}(R) = \operatorname{argmax}_{\omega \in \Omega} p(c(R) = \omega | I \cap c(R) \in \Omega) = \operatorname{argmax}_{\omega \in \Omega} \frac{p_\omega \int_{q \in Q_\omega} v_\omega(q) \prod_{s \in R} \xi_\omega(q)(I(s)) dq}{\sum_{\pi \in \Omega} p_\pi \int_{q \in Q_\pi} v_\pi(q) \prod_{s \in R} \xi_\pi(q)(I(s)) dq}. \tag{5.63}$$

And we can use as confidence measure the probability of the region being of the selected class, assuming again that the real class is in Ω . This gives

$$\hat{c}(R) = \log \max_{\omega \in \Omega} p(c(R) = \omega | I \cap c(R) \in \Omega) = \log \max_{\omega \in \Omega} \frac{p_\omega \int_{q \in Q_\omega} v_\omega(q) \prod_{s \in R} \xi_\omega(q)(I(s)) dq}{\sum_{\pi \in \Omega} p_\pi \int_{q \in Q_\pi} v_\pi(q) \prod_{s \in R} \xi_\pi(q)(I(s)) dq}. \quad (5.64)$$

That is,

$$\hat{c}(R) = \operatorname{argmax}_{\omega \in \Omega} K_{\text{C:MAPo}}(\omega, R), \quad (5.65)$$

$$\hat{c}(R) = \log \max_{\omega \in \Omega} K_{\text{C:MAPo}}(\omega, R), \quad (5.66)$$

$$K_{\text{C:MAPo}}(\omega, R) = \frac{p_\omega \int_{q \in Q_\omega} v_\omega(q) \prod_{s \in R} \xi_\omega(q)(I(s)) dq}{\sum_{\pi \in \Omega} p_\pi \int_{q \in Q_\pi} v_\pi(q) \prod_{s \in R} \xi_\pi(q)(I(s)) dq}. \quad (5.67)$$

Note that a similar derivation in a maximum likelihood approach gives the same result as equation (5.47), since, given that $c(R) = \omega \Rightarrow c(R) \in \Omega$,

$$p(I | c(R) = \omega \cap c(R) \in \Omega) = p(I | c(R) = \omega). \quad (5.68)$$

The probability itself can be calculated as

$$\begin{aligned} p(I | c(R) = \omega \cap c(R) \in \Omega) &= p(I | c(R) = \omega) \\ &= p(R | c(R) = \omega) p(R^c | c(R) = \omega) \\ &= p(R^c) p(\cap_{s \in R} I(s) | c(R) = \omega) \\ &= p(R^c) \int_{q \in Q_\omega} p(q | c(R) = \omega) p(\cap_{s \in R} I(s) | q \cap c(R) = \omega) dq \\ &= p(R^c) \int_{q \in Q_\omega} v_\omega(q) \prod_{s \in R} p(I(s) | q \cap c(R) = \omega) dq \\ &= p(R^c) \int_{q \in Q_\omega} v_\omega(q) \prod_{s \in R} \xi_\omega(q)(I(s)) dq. \end{aligned} \quad (5.69)$$

The probability $p(R^c)$ is not needed to determine the most probable class for a region

$$\hat{c}(R) = \operatorname{argmax}_{\omega \in \Omega} p(I | c(R) = \omega \cap c(R) \in \Omega) = \operatorname{argmax}_{\omega \in \Omega} \int_{q \in Q_\omega} v_\omega(q) \prod_{s \in R} \xi_\omega(q)(I(s)) dq, \quad (5.70)$$

which is the same as equation (5.47).

5.4.3 Yuille's confidence measure

Yuille, Coughlan, and others propose in [YC00, YCWZ01] to use as a confidence measure the difference of log-probabilities of the most probable class to the second most probable, as follows:

Let us sort the classes in Ω , in decreasing order of the probability that the region R belongs to each class, given that the class is in Ω . The i -th class in this sequence is $\omega(R, i) \in \Omega$ for $i = \{1, \dots, |\Omega|\}$, that is,

$$p(c(R) = \omega(R, i) | I \cap c(R) \in \Omega) \geq p(c(R) = \omega(R, i+1) | I \cap c(R) \in \Omega), \quad \text{for } i \in \{1, \dots, |\Omega| - 1\}. \quad (5.71)$$

Then, they suggest using as confidence measure

$$\hat{\kappa}(R) = \log \frac{\mathbb{p}(c(R) = \omega(R, 1) | I \cap c(R) \in \Omega)}{\mathbb{p}(c(R) = \omega(R, 2) | I \cap c(R) \in \Omega)}. \quad (5.72)$$

For classification methods where the class of a region is determined by an expression like

$$\hat{c}(R) = \operatorname{argmax}_{\omega \in \Omega} K(\omega, R), \quad (5.73)$$

we can often take $\alpha K(\omega, R)$ as being the probability of R being of class ω , with α a constant factor. In that case, Yuille's confidence measure is

$$\hat{\kappa}(R) = \log \frac{K(\omega_1, R)}{K(\omega_2, R)}, \quad (5.74)$$

where ω_1 and ω_2 are the most probable classes,

$$\omega_1 = \operatorname{argmax}_{\omega \in \Omega} K(\omega, R), \quad (5.75)$$

$$\omega_2 = \operatorname{argmax}_{\omega \in \Omega \setminus \{\omega_1\}} K(\omega, R), \quad (5.76)$$

and the constant factor α is not used.

We have used this confidence measure as a variant in many classification algorithms, in the evaluation sections, sections 5.7–5.9. We have named these algorithm variants with a suffix “d”. For clarity of notation in the evaluation sections, for a function $X(\omega)$, we define a functional $\mathbf{Y}(X)$ as

$$\mathbf{Y}(X) = \log \frac{X(\omega_1)}{X(\omega_2)}, \quad (5.77)$$

where ω_1 and ω_2 are the most probable classes,

$$\omega_1 = \operatorname{argmax}_{\omega \in \Omega} X(\omega), \quad (5.78)$$

$$\omega_2 = \operatorname{argmax}_{\omega \in \Omega \setminus \{\omega_1\}} X(\omega). \quad (5.79)$$

Given a function f taking two arguments, $f(\omega, y)$, we shall use the notation $f(\cdot, y)$ for the function taking one argument, such that

$$f(\cdot, y)(\omega) \mapsto f(\omega, y). \quad (5.80)$$

5.4.4 Classification by goodness of fit

In the probabilistic classification approach of section 5.4.1 we calculate the probability that each pixel in a region is a realization of the region's law $\Xi(q)$. As we did for flat models, we can instead check how similar the distribution of pixel values in the region is to the distribution of $\Xi(q)$. This is explored in this section.

For a random variable $\Psi \in \mathcal{V}_{\mathbf{D}}$, and a sequence of pixel values $X = \{x_1, \dots, x_N\}$, $x_i \in \mathbf{D}$, let $d(X, \Psi)$ be an indicator of the dissimilarity between the distribution of X and Ψ . The Kullback-Leibler divergence of equation (5.29) and the χ^2 statistic of equation (5.33) are suitable indicators.

For a region R , we can then calculate, for each $\omega \in \Omega$, the values

$$\hat{q}_\omega = \operatorname{argmin}_{q \in Q_\omega} d(I(R), \Xi_\omega(q)),$$

$$\hat{d}_\omega = \min_{q \in Q_\omega} d(I(R), \Xi_\omega(q)),$$

and then choose as classification for region R the one with lowest \hat{d} ,

$$\hat{c}_{\text{dist}}(R) = \operatorname{argmin}_{\omega \in \Omega} \min_{q \in Q_\omega} d(I(R), \Xi_\omega(q)). \quad (5.81)$$

We suggest taking a confidence value based on the dissimilarity to the optimal distribution. By defining, for the Kullback-Leibler divergence,

$$K_{C:\text{KL}}(\omega, R) = - \min_{q \in Q_\omega} d_{\text{KL}}(I(R), \Xi_\omega(q)), \quad (5.82)$$

and, for the χ^2 statistic,

$$K_{C:\chi^2}(\omega, R) = - \min_{q \in Q_\omega} \log \chi^2(I(R), \Xi_\omega(q)), \quad \text{and} \quad (5.83)$$

$$K_{C:\chi^2_n}(\omega, R) = - \min_{q \in Q_\omega} \log \frac{\chi^2(I(R), \Xi_\omega(q))}{|R|}, \quad (5.84)$$

(where as in equation (5.35) the $1/|R|$ makes the confidence coefficient independent of region size), we can use the following combinations

$$\hat{c}(R) = \operatorname{argmax}_{\omega \in \Omega} K_{C:\text{KL}}(\omega, R), \quad \hat{r}(R) = \max_{\omega \in \Omega} K_{C:\text{KL}}(\omega, R), \quad (5.85)$$

$$\hat{c}(R) = \operatorname{argmax}_{\omega \in \Omega} K_{C:\chi^2}(\omega, R), \quad \hat{r}(R) = \max_{\omega \in \Omega} K_{C:\chi^2_n}(\omega, R). \quad (5.86)$$

Some variants have also been tested in the evaluation. First, a variant for the χ^2 statistic without the $1/|R|$ correcting exponent,

$$\hat{c}(R) = \operatorname{argmax}_{\omega \in \Omega} K_{C:\chi^2}(\omega, R), \quad \hat{r}(R) = \max_{\omega \in \Omega} K_{C:\chi^2}(\omega, R); \quad (5.87)$$

second, variants with the symmetrical version of the Kullback-Leibler divergence,

$$\hat{c}(R) = \operatorname{argmax}_{\omega \in \Omega} K_{C:\text{KL}_s}(\omega, R), \quad \hat{r}(R) = \max_{\omega \in \Omega} K_{C:\text{KL}_s}(\omega, R), \quad (5.88)$$

where

$$K_{C:\text{KL}_s}(\omega, R) = - \min_{q \in Q_\omega} d_{\text{KL}_s}(I(R), \Xi_\omega(q)); \quad (5.89)$$

and, finally, variants where we add a $v_\omega(q)$ term in an attempt to incorporate prior probabilities into that formalism,

$$\hat{c}(R) = \operatorname{argmax}_{\omega \in \Omega} K_{C:\text{KL}_v}(\omega, R), \quad \hat{r}(R) = \max_{\omega \in \Omega} K_{C:\text{KL}_v}(\omega, R), \quad (5.90)$$

$$\hat{c}(R) = \operatorname{argmax}_{\omega \in \Omega} K_{C:\text{KL}_{sv}}(\omega, R), \quad \hat{r}(R) = \max_{\omega \in \Omega} K_{C:\text{KL}_{sv}}(\omega, R), \quad (5.91)$$

$$\hat{c}(R) = \operatorname{argmax}_{\omega \in \Omega} K_{C:\chi^2_v}(\omega, R), \quad \hat{r}(R) = \max_{\omega \in \Omega} K_{C:\chi^2_v}(\omega, R), \quad (5.92)$$

$$\hat{c}(R) = \operatorname{argmax}_{\omega \in \Omega} K_{C:\chi^2_{nv}}(\omega, R), \quad \hat{r}(R) = \max_{\omega \in \Omega} K_{C:\chi^2_{nv}}(\omega, R), \quad (5.93)$$

where

$$K_{C:KL_V}(\omega, R) = -\min_{q \in Q_\omega} (d_{KL}(I(R), \Xi_\omega(q)) - \log v_\omega(q)), \quad (5.94)$$

$$K_{C:KL_{SV}}(\omega, R) = -\min_{q \in Q_\omega} (d_{KL_S}(I(R), \Xi_\omega(q)) - \log v_\omega(q)), \quad (5.95)$$

$$K_{C:\chi^2_V}(\omega, R) = -\min_{q \in Q_\omega} \log(v_\omega(q)^{-1} \chi^2(I(R), \Xi_\omega(q))), \quad (5.96)$$

$$K_{C:\chi^2_{NV}}(\omega, R) = -\min_{q \in Q_\omega} \log \frac{\chi^2(I(R), \Xi_\omega(q))}{v_\omega(q) |R|}. \quad (5.97)$$

In this approach, we are evaluating how well the distribution for a region fits an instance of a random variable class. This is not the same as testing how well the distribution for a set of regions fits a random variable class, which will be considered, indirectly, in section 5.5.4 for the purposes of model selection.

5.4.5 Classification by maximising over q

It would seem, intuitively, that there is another approach to solving the classification problem in this nested model setting. The idea is to follow the *maximum likelihood* approach of section 5.4.1 but instead of, for each class, integrating for all values of q , we would take into account only its “optimal” value.

This means deciding that region R belongs to class

$$\hat{c}(R) = \operatorname{argmax}_{\omega \in \Omega} \max_{q \in Q_\omega} p(R | c(R) = \omega \cap q) \quad (5.98)$$

$$= \operatorname{argmax}_{\omega \in \Omega} \max_{q \in Q_\omega} \prod_{s \in R} \xi_\omega(q)(I(s)). \quad (5.99)$$

This approach is justified only by intuition, with no probabilistic backing. In the evaluation section it is compared with the more mathematically-backed MAP and ML approaches of equations (5.45) and (5.47).

In parallel with the standard ML approach of equation (5.47), we can define the following

$$K_{C:qML}(\omega, R) = \max_{q \in Q_\omega} \prod_{s \in R} \xi_\omega(q)(I(s)), \quad (5.100)$$

$$K_{C:qMLn}(\omega, R) = \max_{q \in Q_\omega} (\prod_{s \in R} \xi_\omega(q)(I(s)))^{1/|R|}. \quad (5.101)$$

and use these combinations of classes and confidence coefficients, without or with an exponent $1/|R|$ for region size independence:

$$\hat{c}(R) = \operatorname{argmax}_{\omega \in \Omega} K_{C:qML}(\omega, R), \quad \hat{\kappa}(R) = \log \max_{\omega \in \Omega} K_{C:qML}(\omega, R), \quad (5.102)$$

$$\hat{c}(R) = \operatorname{argmax}_{\omega \in \Omega} K_{C:qMLn}(\omega, R), \quad \hat{\kappa}(R) = \log \max_{\omega \in \Omega} K_{C:qMLn}(\omega, R). \quad (5.103)$$

In addition, the following variants have been evaluated:

$$\hat{c}(R) = \operatorname{argmax}_{\omega \in \Omega} K_{C:\text{qMLn}}(\omega, R), \quad \hat{\kappa}(R) = \log \max_{\omega \in \Omega} K_{C:\text{qMLn}}(\omega, R), \quad (5.104)$$

$$\hat{c}(R) = \operatorname{argmax}_{\omega \in \Omega} K_{C:\text{qMLv}}(\omega, R), \quad \hat{\kappa}(R) = \log \max_{\omega \in \Omega} K_{C:\text{qMLv}}(\omega, R), \quad (5.105)$$

$$\hat{c}(R) = \operatorname{argmax}_{\omega \in \Omega} K_{C:\text{qMLv}}(\omega, R), \quad \hat{\kappa}(R) = \log \max_{\omega \in \Omega} K_{C:\text{qMLnv}}(\omega, R), \quad (5.106)$$

$$\hat{c}(R) = \operatorname{argmax}_{\omega \in \Omega} K_{C:\text{qMLnv}}(\omega, R), \quad \hat{\kappa}(R) = \log \max_{\omega \in \Omega} K_{C:\text{qMLnv}}(\omega, R), \quad (5.107)$$

where

$$K_{C:\text{qMLv}} = \max_{q \in Q_\omega} v_\omega(q) \left(\prod_{s \in R} \xi_\omega(q)(I(s)) \right), \quad (5.108)$$

$$K_{C:\text{qMLnv}} = \max_{q \in Q_\omega} v_\omega(q) \left(\prod_{s \in R} \xi_\omega(q)(I(s)) \right)^{1/|R|}. \quad (5.109)$$

The corresponding variant for a *maximum a posteriori* approach is meaningless. That would mean choosing as classification for region R the class

$$\hat{c}(R) = \operatorname{argmax}_{\omega \in \Omega} \max_{q \in Q_\omega} p(c(R) = \omega | R \cap q), \quad (5.110)$$

which gives

$$\begin{aligned} \hat{c}(R) &= \operatorname{argmax}_{\omega \in \Omega} \max_{q \in Q_\omega} p(c(R) = \omega | R \cap q) \\ &= \operatorname{argmax}_{\omega \in \Omega} \max_{q \in Q_\omega} \frac{p(c(R) = \omega \cap R \cap q)}{p(R \cap q)} \\ &= \operatorname{argmax}_{\omega \in \Omega} \max_{q \in Q_\omega} \frac{p(R | q \cap c(R) = \omega) p(q | c(R) = \omega) p(c(R) = \omega)}{\sum_{\omega' \in \Omega} p(c(R) = \omega' \cap R \cap q)} \\ &= \operatorname{argmax}_{\omega \in \Omega} \max_{q \in Q_\omega} \frac{p(c(R) = \omega) v_\omega(q) \prod_{s \in R} \xi_\omega(q)(I(s))}{\sum_{\omega' \in \Omega} p(c(R) = \omega') v_{\omega'}(q) \prod_{s \in R} \xi_{\omega'}(q)(I(s))} \\ &= \operatorname{argmax}_{\omega \in \Omega} \max_{q \in Q_\omega} \frac{p_\omega v_\omega(q) \prod_{s \in R} \xi_\omega(q)(I(s))}{\sum_{\omega' \in \Omega} p_{\omega'} v_{\omega'}(q) \prod_{s \in R} \xi_{\omega'}(q)(I(s))}, \end{aligned} \quad (5.111)$$

but in $\sum_{\omega' \in \Omega} p(c(R) = \omega' \cap R \cap q)$ we would use $q \in Q_\omega$ to find probabilities of $c(R) = \omega'$, thus requiring $v_{\omega'}(q)$. But q may not be in the domain of $v_{\omega'}$, which is $Q_{\omega'}$. Rewriting the expression by maximising not for $q \in Q_\omega$ but for $q \in \cup_{x \in \Omega} Q_x$ and extending $v_{\omega'}(q) = 0$ if $q \notin Q_{\omega'}$ would create divisions by zero and is, thus, not helpful. The reason is, possibly, the lack of meaning, from a probability point of view, of equation (5.110).

5.4.6 Classification by finding the optimal q

The chosen q in equation (5.98) is optimal in the sense of maximizing the probability of the class or of the observations, but it is not itself the most probable value of q . Using the most probable value of q is much more interesting from an practical point of view because the calculation of $\hat{c}(R)$ may require two separate loops, instead of two nested loops.

The value of q that has most likely produced the pixels in region R , assuming they belong to class ω ,

$$\hat{q}_\omega(R) = \operatorname{argmax}_{q \in Q_\omega} p(q | R, c(R) = \omega), \quad (5.112)$$

can, in some cases, be found very easily. For example, for a random variable class Ξ whose instances $\Xi(q)$ are the random variables $M+q$, for some base random variable M (see section 5.5.2), the most likely q can be found as

$$\hat{q} = \mu_R - E(M), \quad (5.113)$$

where μ_R is the mean of the pixel values in region R . For the case described in section 5.5.2, where the instances $\Xi(q_a, q_b)$ are the random variables $Mq_a + q_b$ for some base random variable M , the most likely q_a and q_b can be found as

$$\hat{q}_b = \mu_R - E(M), \quad (5.114)$$

$$\hat{q}_a^{2d} = \frac{\det \Sigma_R}{\det \Sigma_M}, \quad (5.115)$$

where μ_R is the mean of the pixel values in region R , Σ_R is its sample covariance matrix, Σ_M is M 's covariance matrix, and $\Xi \in \mathcal{K}_{\mathbb{R}^{d+1} \rightarrow \mathbb{R}^d}$.

Once $\hat{q}_\omega(R)$ has been found, we can then adapt the methods already presented as follows.

For the Bayesian approaches (MAP and ML), we define

$$K_{C:eMAP}(\omega, R) = \frac{p_\omega v_\omega(\hat{q}_\omega(R)) \prod_{s \in R} \xi_\omega(\hat{q}_\omega(R))(I(s))}{\sum_{\omega' \in \Omega} p_{\omega'} v_{\omega'}(\hat{q}_\omega(R)) \prod_{s \in R} \xi_{\omega'}(\hat{q}_\omega(R))(I(s))}, \quad (5.116)$$

$$K_{C:eMAPn}(\omega, R) = \frac{p_\omega v_\omega(\hat{q}_\omega(R)) (\prod_{s \in R} \xi_\omega(\hat{q}_\omega(R))(I(s)))^{1/|R|}}{\sum_{\omega' \in \Omega} p_{\omega'} v_{\omega'}(\hat{q}_\omega(R)) (\prod_{s \in R} \xi_{\omega'}(\hat{q}_\omega(R))(I(s)))^{1/|R|}}, \quad (5.117)$$

$$K_{C:eML}(\omega, R) = \prod_{s \in R} \xi_\omega(\hat{q}_\omega(R))(I(s)), \quad (5.118)$$

and use these for the following combinations

$$\begin{aligned} \hat{c}(R) &= \operatorname{argmax}_{\omega \in \Omega} p(c(R) = \omega | R, \hat{q}_\omega(R)) & \hat{r}(R) &= \log \max_{\omega \in \Omega} K_{C:eMAP}(\omega, R), \\ &= \operatorname{argmax}_{\omega \in \Omega} K_{C:eMAP}(\omega, R), \end{aligned} \quad (5.119)$$

$$\hat{c}(R) = \operatorname{argmax}_{\omega \in \Omega} K_{C:eMAP}(\omega, R), \quad \hat{r}(R) = \log \max_{\omega \in \Omega} K_{C:eMAPn}(\omega, R), \quad (5.120)$$

$$\hat{c}(R) = \operatorname{argmax}_{\omega \in \Omega} K_{C:eMAPn}(\omega, R), \quad \hat{r}(R) = \log \max_{\omega \in \Omega} K_{C:eMAPn}(\omega, R), \quad (5.121)$$

$$\begin{aligned} \hat{c}(R) &= \operatorname{argmax}_{\omega \in \Omega} p(R | \hat{q}_\omega(R), c(R) = \omega) & \hat{r}(R) &= \log \max_{\omega \in \Omega} K_{C:eML}(\omega, R), \\ &= \operatorname{argmax}_{\omega \in \Omega} K_{C:eML}(\omega, R), \end{aligned} \quad (5.122)$$

$$\hat{c}(R) = \operatorname{argmax}_{\omega \in \Omega} K_{C:eML}(\omega, R), \quad \hat{r}(R) = \log \max_{\omega \in \Omega} (K_{C:eML}(\omega, R))^{1/|R|}, \quad (5.123)$$

$$\hat{c}(R) = \operatorname{argmax}_{\omega \in \Omega} (K_{C:eML}(\omega, R))^{1/|R|}, \quad \hat{r}(R) = \log \max_{\omega \in \Omega} (K_{C:eML}(\omega, R))^{1/|R|}, \quad (5.124)$$

$$\hat{c}(R) = \operatorname{argmax}_{\omega \in \Omega} v_{\omega}(\hat{q}_{\omega}(R)) \left(K_{C:eML}(\omega, R) \right)^{1/|R|}, \quad \hat{r}(R) = \log \max_{\omega \in \Omega} v_{\omega}(\hat{q}_{\omega}(R)) \left(K_{C:eML}(\omega, R) \right)^{1/|R|}, \quad (5.125)$$

$$\begin{aligned} \hat{c}(R) &= \operatorname{argmax}_{\omega \in \Omega} p(c(R) = \omega, \hat{q}_{\omega}(R) | R) & \hat{r}(R) &= \log \max_{\omega \in \Omega} p_{\omega} v_{\omega}(\hat{q}_{\omega}(R)) K_{C:eML}(\omega, R), \\ &= \operatorname{argmax}_{\omega \in \Omega} p_{\omega} v_{\omega}(\hat{q}_{\omega}(R)) K_{C:eML}(\omega, R), & & \end{aligned} \quad (5.126)$$

$$\hat{c}(R) = \operatorname{argmax}_{\omega \in \Omega} p_{\omega} v_{\omega}(\hat{q}_{\omega}(R)) K_{C:eML}(\omega, R), \quad \hat{r}(R) = \log \max_{\omega \in \Omega} p_{\omega} v_{\omega}(\hat{q}_{\omega}(R)) \left(K_{C:eML}(\omega, R) \right)^{1/|R|}, \quad (5.127)$$

$$\hat{c}(R) = \operatorname{argmax}_{\omega \in \Omega} p_{\omega} v_{\omega}(\hat{q}_{\omega}(R)) \left(K_{C:eML}(\omega, R) \right)^{1/|R|}, \quad \hat{r}(R) = \log \max_{\omega \in \Omega} p_{\omega} v_{\omega}(\hat{q}_{\omega}(R)) \left(K_{C:eML}(\omega, R) \right)^{1/|R|}. \quad (5.128)$$

For the goodness-of-fit approaches (Kullback-Leibler divergence and χ^2 statistic) we define

$$K_{C:eKL}(\omega, R) = -d_{KL}(I(R), \Xi_{\omega}(\hat{q}_{\omega}(R))), \quad (5.129)$$

$$K_{C:eKLs}(\omega, R) = -d_{KLs}(I(R), \Xi_{\omega}(\hat{q}_{\omega}(R))), \quad (5.130)$$

$$K_{C:e\chi^2}(\omega, R) = -\log \chi^2(I(R), \Xi_{\omega}(\hat{q}_{\omega}(R))), \quad (5.131)$$

$$K_{C:e\chi^2n}(\omega, R) = -\log \left(\chi^2(I(R), \Xi_{\omega}(\hat{q}_{\omega}(R))) / |R| \right), \quad (5.132)$$

$$K_{C:e\chi^2v}(\omega, R) = \log \left(\chi^2(I(R), \Xi_{\omega}(\hat{q}_{\omega}(R)))^{-1} v_{\omega}(\hat{q}_{\omega}(R)) \right), \quad (5.133)$$

$$K_{C:e\chi^2nv}(\omega, R) = \log \left(\chi^2(I(R), \Xi_{\omega}(\hat{q}_{\omega}(R)))^{-1} |R| v_{\omega}(\hat{q}_{\omega}(R)) \right), \quad (5.134)$$

$$K_{C:e\chi^2vo}(\omega, R) = \log \left(\chi^2(I(R), \Xi_{\omega}(\hat{q}_{\omega}(R)))^{-1} \tilde{v}_{\omega}(R) \right), \quad (5.135)$$

$$K_{C:e\chi^2nvo}(\omega, R) = \log \left(\chi^2(I(R), \Xi_{\omega}(\hat{q}_{\omega}(R)))^{-1} |R| \tilde{v}_{\omega}(R) \right), \quad (5.136)$$

where

$$\tilde{v}_{\omega}(R) = \frac{v_{\omega}(\hat{q}_{\omega}(R))}{\max_{q \in Q_{\omega}} v_{\omega}(q)}, \quad (5.137)$$

and use these for the following combinations

$$\hat{c}(R) = \operatorname{argmax}_{\omega \in \Omega} K_{C:eKL}(\omega, R), \quad \hat{r}(R) = \max_{\omega \in \Omega} K_{C:eKL}(\omega, R), \quad (5.138)$$

$$\hat{c}(R) = \operatorname{argmax}_{\omega \in \Omega} K_{C:eKLs}(\omega, R), \quad \hat{r}(R) = \max_{\omega \in \Omega} K_{C:eKLs}(\omega, R), \quad (5.139)$$

$$\hat{c}(R) = \operatorname{argmax}_{\omega \in \Omega} v_{\omega}(\hat{q}_{\omega}(R)) K_{C:eKL}(\omega, R), \quad \hat{r}(R) = \max_{\omega \in \Omega} v_{\omega}(\hat{q}_{\omega}(R)) K_{C:eKL}(\omega, R), \quad (5.140)$$

$$\hat{c}(R) = \operatorname{argmax}_{\omega \in \Omega} \tilde{v}_{\omega}(R) K_{C:eKL}(\omega, R), \quad \hat{r}(R) = \max_{\omega \in \Omega} \tilde{v}_{\omega}(R) K_{C:eKL}(\omega, R), \quad (5.141)$$

$$\hat{c}(R) = \operatorname{argmax}_{\omega \in \Omega} v_{\omega}(\hat{q}_{\omega}(R)) K_{C:eKLs}(\omega, R), \quad \hat{r}(R) = \max_{\omega \in \Omega} v_{\omega}(\hat{q}_{\omega}(R)) K_{C:eKLs}(\omega, R), \quad (5.142)$$

$$\hat{c}(R) = \operatorname{argmax}_{\omega \in \Omega} \tilde{v}_{\omega}(R) K_{C:eKLs}(\omega, R), \quad \hat{r}(R) = \max_{\omega \in \Omega} \tilde{v}_{\omega}(R) K_{C:eKLs}(\omega, R), \quad (5.143)$$

$$\hat{c}(R) = \operatorname{argmax}_{\omega \in \Omega} K_{C:e\chi^2}(\omega, R), \quad \hat{\kappa}(R) = \max_{\omega \in \Omega} K_{C:e\chi^2}(\omega, R), \quad (5.144)$$

$$\hat{c}(R) = \operatorname{argmax}_{\omega \in \Omega} K_{C:e\chi^2}(\omega, R), \quad \hat{\kappa}(R) = \max_{\omega \in \Omega} K_{C:e\chi^2_n}(\omega, R), \quad (5.145)$$

$$\hat{c}(R) = \operatorname{argmax}_{\omega \in \Omega} K_{C:e\chi^2_n}(\omega, R), \quad \hat{\kappa}(R) = \max_{\omega \in \Omega} K_{C:e\chi^2_n}(\omega, R), \quad (5.146)$$

$$\hat{c}(R) = \operatorname{argmax}_{\omega \in \Omega} K_{C:e\chi^2_v}(\omega, R), \quad \hat{\kappa}(R) = \max_{\omega \in \Omega} K_{C:e\chi^2_v}(\omega, R), \quad (5.147)$$

$$\hat{c}(R) = \operatorname{argmax}_{\omega \in \Omega} K_{C:e\chi^2_v}(\omega, R), \quad \hat{\kappa}(R) = \max_{\omega \in \Omega} K_{C:e\chi^2_{nv}}(\omega, R), \quad (5.148)$$

$$\hat{c}(R) = \operatorname{argmax}_{\omega \in \Omega} K_{C:e\chi^2_{nv}}(\omega, R), \quad \hat{\kappa}(R) = \max_{\omega \in \Omega} K_{C:e\chi^2_{nv}}(\omega, R), \quad (5.149)$$

$$\hat{c}(R) = \operatorname{argmax}_{\omega \in \Omega} K_{C:e\chi^2_{v_0}}(\omega, R), \quad \hat{\kappa}(R) = \max_{\omega \in \Omega} K_{C:e\chi^2_{v_0}}(\omega, R), \quad (5.150)$$

$$\hat{c}(R) = \operatorname{argmax}_{\omega \in \Omega} K_{C:e\chi^2_{v_0}}(\omega, R), \quad \hat{\kappa}(R) = \max_{\omega \in \Omega} K_{C:e\chi^2_{nv_0}}(\omega, R), \quad (5.151)$$

$$\hat{c}(R) = \operatorname{argmax}_{\omega \in \Omega} K_{C:e\chi^2_{nv_0}}(\omega, R), \quad \hat{\kappa}(R) = \max_{\omega \in \Omega} K_{C:e\chi^2_{nv_0}}(\omega, R). \quad (5.152)$$

In practice, and to avoid problems with the finiteness of the data domain \mathbf{D} , the means and covariances of the pixel values in a region, and those of the random variables, are calculated from histograms, after emptying the histogram bins that have a relative frequency lower than a certain threshold relative to the maximum relative frequency.

5.5 Model estimation

In order to run these classification algorithms, we need the *a priori* probabilities $p(\omega)$ for the terrain classes, the probability density functions v_ω and, for compound random variable classification, the region probability density functions ξ_ω .

These can be obtained from training data, that is, from an image I , a set of regions \mathbf{R} , and a known classification for regions in that set. The set \mathbf{R} does not need to cover the whole image.

The class prior probability for class ω_i , $p(\omega_i)$, can be estimated by the sample frequency of regions of class ω_i , that is

$$\hat{p}(\omega_i) = \frac{\operatorname{card}\{R \in \mathbf{R} : c(R) = \omega_i\}}{\operatorname{card} \mathbf{R}}. \quad (5.153)$$

We cannot simply “estimate” an arbitrary random variable. Instead, a certain type of random variable is selected—for example, Gaussian—and its *parameters* are estimated—if Gaussian was selected, the parameters are its mean and covariance matrix. Given a number of random laws available in an implementation, we can estimate the parameters for each law, and then select the one that best fits the training data using the methods described in section 5.5.1. In the following sections we propose estimators for the parameters of several random laws. Visual inspection of the data suggests these random laws are sufficient to model our data, but it could be the case that a different random law would be more appropriate for other applications.

Note that these estimations are for random variables with values in \mathbb{R}^d , whereas we will be dealing with values in $\{0, \dots, 255\}^d$. In practice, the quantization of $[0, 255]^d$ by $\{0, \dots, 255\}^d$ is however fine enough to avoid any particular problem.

Figure 5.5 shows a flowchart of steps involved in estimating a probability model in the proposed system.

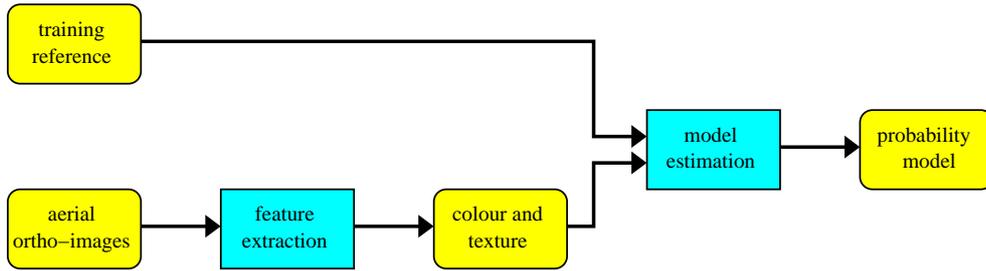


Figure 5.5: Flowchart depicting the processing steps (in blue rectangles) and intermediate results (in yellow rounded boxes) for the estimation of a probability model for classification in the proposed system.

We will start by discussing how to estimate parameters for some typical random variables, and how to select the best estimation. Then, methods for estimating parameters for some specific random variable classes, for use in nested classification, will be presented.

5.5.1 Estimation of simple random variables

The probability density function for class ω_i for the flat model setting (section 5.2) can be obtained from the set of pixels whose class is ω_i .

Let $X = (x_1, \dots, x_N)$ be the a sequence of values from which a random variable is to be estimated. That is, we wish to obtain the parameters of a random variable such that X is a likely sample of that random variable. For a d -channel image, $x_i \in \mathbf{D} \subset \mathbb{R}^d$. Let x_{ij} denote the j -th component of x_i . Let $\bar{\mu} \in \mathbb{R}^d$ be the sample mean of X , and $\bar{\mathbf{S}}$ its sample covariance matrix. Let \bar{s}_{nm} be the elements of $\bar{\mathbf{S}}$. In estimating a model for class ω_i , X would be the sequence of values of pixels in training polygons which are labeled to ω_i , taken in arbitrary order and allowing duplicate pixel values.

The first step is to use this training data X to estimate a model for each of the probability laws proposed in an implementation. In this chapter several laws are proposed, such as Gaussian, Laplacian, or uniform, but specific applications may require the definition of additional models. For each law, the parameters of the model parameters are estimated from X by maximum likelihood estimation. This gives, for each law, the probability distribution most closely following the distribution of X .

The second step is to select one of these models. As explained in section 5.5.1, the Bayes Information Criterion is used to select the best model, balancing the complexity of the model with the quality of its fit to the data.

In the implementation used for the evaluation of sections 5.7–5.9, we provide, in addition to a standard d -dimensional Gaussian model, probability models for Laplacian random variables, rectangular uniform random variables, non-parametric random variables, and for kernel density estimation.

Laplacian random variable

A 1-dimensional Laplacian random variable V of parameters λ and m has probability density function $v(x) = \frac{1}{2\lambda} e^{-|x-m|/\lambda}$. The higher-dimensional version is not easily found in the literature; we can define one as follows: Let $A = (A_1, \dots, A_d)$ be a vector of independent 1-dimensional Laplacian random variables of mean zero ($m = 0$) and $\lambda = 1$ ($\sigma^2 = 2$). The random variable

$W = \mathbf{M} \cdot A + \mu$, where \mathbf{M} is an invertible $d \times d$ matrix and $\mu \in \mathbb{R}^d$, is a d -dimensional Laplacian random variable. Its probability density function $w(x)$ is

$$w(x) = \frac{1}{2^d |\det \mathbf{M}|} \cdot e^{-\|\mathbf{M}^{-1}(x-\mu)\|_1}, \quad (5.154)$$

where $\|\cdot\|_1$ is the L_1 norm, $\|(x_1, x_2, \dots, x_n)\|_1 = \sum_{i=1}^n |x_i|$. Its mean is $E(W) = \mu$ and its covariance matrix is $\text{cov } W = 2 \cdot \mathbf{M} \cdot \mathbf{M}^T$, where \mathbf{M}^T denotes matrix transposition. Estimates $\hat{\mu}$ and $\hat{\mathbf{M}}$ for its parameters μ and \mathbf{M} can be calculated from the sample's mean and covariance—using Cholesky decomposition—with

$$\hat{\mu} = \bar{\mu}, \quad 2 \cdot \hat{\mathbf{M}} \cdot \hat{\mathbf{M}}^T = \bar{\mathbf{S}}. \quad (5.155)$$

Rectangular uniform random variable

A d -dimensional random variable $V = (V_1, \dots, V_d)$ with constant probability over a rectangular support $V_S = [a_1, b_1] \times [a_2, b_2] \times \dots \times [a_d, b_d] \subset \mathbb{R}^d$ has a probability density function $v(x)$

$$v(x) = \begin{cases} \prod_{j=1}^d \frac{1}{b_j - a_j} & \text{if } x \in V_S, \\ 0 & \text{otherwise.} \end{cases} \quad (5.156)$$

Its a and b parameters are related to its mean and covariance as $E(V_j) = (b_j - a_j)/2$ and $E((V_j - E(V_j))^2) = ((b_j - a_j)^2)/12$, and we can therefore give estimates \hat{a} and \hat{b} for parameters a and b from the sample X as

$$\hat{a} = \bar{\mu} - w/2, \quad \hat{b} = \bar{\mu} + w/2, \quad \text{where } w_i = \sqrt{12 \bar{s}_{ii}}, \quad w = (w_1, w_2, \dots, w_d). \quad (5.157)$$

Raw non-parametric model

We can also estimate a “non-parametric” random variable whose probability density function follows the distribution histogram of the sample X . Let us partition the data domain \mathbf{D} into equally-shaped histogram bins, $\{D_i\}_i$, with $\cup_i D_i = \mathbf{D}$ and $\forall i \neq j. D_i \cap D_j = \emptyset$. Let us construct a relative frequency histogram h from X by counting the number of values in X that belong to the i -th histogram bin, for each i : $h_i = \text{card}\{j : x_j \in D_i\}/N$. We have $\sum_i h_i = 1$. We can define a random variable V with values in \mathbf{D} with probability density function $v(x)$

$$v(x) = h_i \quad \text{with } i \text{ such that } x \in D_i. \quad (5.158)$$

In this implementation, several such models are estimated for different partitions of the data domain.

Kernel density estimation

Raw non-parametric estimation as presented in the previous paragraph has an important drawback: even if the underlying density is known to be smooth, a large sample size is necessary in order to obtain an estimation which has high resolution both in the domain (a high number of histogram bins) and in the frequency (a high count in each histogram bin). With small sample sizes we obtain histograms which are either jagged or have a low domain resolution.

If the underlying density is known to be smooth, a technique known as *kernel density estimation* [Ros56, Sil86, Par62, Epa69] can be applied to obtain smooth histograms with high domain

resolution. It consists in calculating a histogram h_i as in last section, with a high domain resolution, and then convolving it by a suitable d -dimensional kernel. In this implementation, several such models are estimated using Gaussian kernels of different variance.

An improvement, known as *variable kernel density estimation*, uses Gaussian kernels of varying variances, using larger variances in the regions of higher uncertainty, that is, in the regions with fewer samples. This was not implemented in this thesis.

Model selection

We have discussed in the previous sections how to choose the parameters for a random variable so that it optimally fits a data sample. One probability model must eventually be selected from those estimated in these sections. We cannot simply select the model that best fits the sample, because this will tend to select models with a high number of degrees of freedom, and cause *overfitting*. Overfitting occurs when a model not only extracts the characteristics of the process having produced the training sample, but also the peculiarities of that precise sample; when this occurs in excess, the model will fit the training sample very well, but will perform poorly on other samples drawn from the same process—that is, *generalization* will be poor. This problem is also typical of neural networks and clustering algorithms. It is usually solved by selecting the model which maximizes a value which balances the fit to the data and a measure of the model complexity. One of these values is the *Bayes Information Criterion*, or BIC [Sch78], defined as

$$BIC = -2 \ln p(X|T) + p \ln N, \quad (5.159)$$

where $p(X|T)$ is the probability of the observations given the model (the likelihood of the observations, and thus the goodness of fit), p is the number of parameters in the model (the complexity), and N is the number of observations (the sample size).

Among the several random variable models for a given land cover class, the one with the lowest value of BIC is selected.

It may be interesting to study whether, if a non-Bayesian classification method is used, the probability models used should correspondingly be selected with something other than BIC—which seems to correspond to a MAP approach. This is not addressed in this thesis. On a related topic, it is possible to select the best model not by using a model fit/complexity criterion such as BIC, but by actually testing how well each of the estimated models classifies data. This requires defining a second, non-overlapping, training set, and the question remains of which classification algorithm to use for this selection.

5.5.2 Estimation of nested random variables

In this section I present estimation methods for a certain number of random variable classes, to be used in the nested model setup of section 5.3 for the nested random variable classification algorithms of section 5.4. As in section 5.5.1, only a limited number of random variable classes are presented, and it is possible that additional kinds of random variable classes would be more appropriate to other applications.

For the remainder of this section, to estimate the random laws that define class ω we will be working with a set $\mathbf{R} = \{R_1, \dots, R_N\}$ of regions of class ω . Let us call X_i the sequence of pixel values of region R_i , in any arbitrary order, but allowing duplicate values, $X_i = (x_{i1}, x_{i2}, \dots, x_{im_i}), x_{ij} \in \mathbf{D}$. Let us call X the aggregate sequence of pixel values or all regions in \mathbf{R} , also in arbitrary order and allowing duplicate values.

Compound estimation should not be done if N is small.

Random translation of random variable

Let M be a random variable with $E(M) = 0$, with values in \mathbb{R}^d and probability density function f_M .

Let us consider the random variable class Ξ , as defined in section 5.3, with values in \mathbb{R}^d , parametrized by a parameter in \mathbb{R}^d which is drawn from the random variable V , such that the density of the instance $\Xi(q)$ is

$$\xi_\omega(q)(x) = f_M(x - q). \quad (5.160)$$

That is, M defines the “shape” of the instances, and q defines the position of this basic shape. The random variable from which a sample X_i is drawn is $q_i + M$, where q_i is the value of q corresponding to the region R_i ; the random variable from which X is drawn is $V + M$ (this addition is component-by-component vector addition). Let the probability density function of V be f_V .

We can estimate M and V in the following way:

For each sample $X_i = (x_{ij})_j$, we calculate its mean $\hat{\mu}_i$ as

$$\hat{\mu}_i = \frac{1}{n_i} \sum_{j=1}^{n_i} x_{ij} \quad (5.161)$$

Given that $E(\hat{\mu}_i) = q_i$, we assume the sequence of values $Q = (\hat{\mu}_1, \hat{\mu}_2, \dots, \hat{\mu}_N)$ to have been drawn from random variable V , which we estimate from Q using one of the techniques described in section 5.5.1 and elsewhere. This gives us an estimation for the probability density function $f_V(v)$.

We can then estimate M as follows: From each sample $X_i = (x_{i1}, x_{i2}, \dots, x_{in_i})$ and its mean $\hat{\mu}_i$ we calculate the centred sample $X'_i = (x_{i1} - \hat{\mu}_i, x_{i2} - \hat{\mu}_i, \dots, x_{in_i} - \hat{\mu}_i)$. We aggregate the centred samples X'_i into a single sample X' (by taking all the elements in each X'_i , in an arbitrary order, with duplicates allowed). Then, using techniques such as those in section 5.5.1 we estimate M from the sample X' .

An alternate method for estimating M , which I have not implemented, relies on the fact that the probability density functions of X , V and M , respectively f_X , f_V and f_M , satisfy

$$f_X(x) = f_V(v) * f_M(m). \quad (5.162)$$

We then estimate X (that is, f_X) from the aggregated sample set X using section 5.5.1, and deconvolve by Fourier transform to obtain

$$f_M = \mathcal{F}^{-1} \left(\frac{\mathcal{F} f_X}{\mathcal{F} f_V} \right) \quad (5.163)$$

(where we indicate the direct and inverse Fourier transform operators by \mathcal{F} and \mathcal{F}^{-1} respectively).

Random scaling and translation of random variable

Let M be a random variable with $E(M) = 0$ and covariance matrix Σ , with $\det \Sigma = 1$ (or another constant value), with values in \mathbb{R}^d and probability density function f_M .

Let us consider the random variable class Ξ , with values in \mathbb{R}^d , parametrized by a parameter in \mathbb{R} which is drawn from the random variable V_a , and a parameter in \mathbb{R}^d which is drawn from V_b , such that the density of the instance $\Xi(q)$ is

$$\xi_\omega(q_a, q_b)(x) = f_M \left(\frac{x - q_b}{q_a} \right). \quad (5.164)$$

That is, M defines the “shape” of the instances, q_a (drawn from V_a) defines their size, and q_b (drawn from V_b) their position. The random variable from which a sample X_i is drawn is $q_{ai}M + q_{bi}$, where q_{ai} and q_{bi} are the values of q_a and q_b corresponding to the region r_i ; the random variable from which X is drawn is $V_A M + V_B$ (the case of V_A having $d \times d$ matrices as values will not be treated here). Let f_{V_a} be the density of V_a , and f_{V_b} the density of V_b .

We can estimate M , V_A , and V_B in the following way.

For each sample $X_i = (x_{i1}, x_{i2}, \dots, x_{in_i})$, we calculate its mean c_i and covariance matrix $S_i = (S_{i,ab})$ as

$$c_i = \frac{1}{n_i} \sum_{n=1}^{n_i} x_{in} \quad (5.165)$$

$$S_{i,ab} = \frac{1}{n_i - 1} \sum_{n=1}^{n_i} (x_{ina} - \mu_i)(x_{inb} - \mu_j), \quad (5.166)$$

where x_{ina} is the a -th component of x_{in} , the n -th pixel in region i .

Given that X_i is drawn from $q_{ai}M + q_{bi}$, the expectation μ_i of its sample mean c_i is

$$\mu_i = E(q_{ai}M + q_{bi}) = q_{ai} E(M) + q_{bi} = q_{bi},$$

and the expectation $\Sigma_i = (\sigma_{i,uv}^2)$ of its sample covariance matrix S_i is

$$\begin{aligned} \sigma_{i,uv}^2 &= E((q_{ai}M_u + q_{biu} - \mu_{iu})(q_{ai}M_v + q_{biv} - \mu_{iv})) \\ &= E(q_{ai}q_{ai}M_uM_v) + E(q_{ai}q_{biv}M_u) - E(\mu_{iv}q_{ai}M_u) \\ &\quad + E(q_{ai}q_{biu}M_v) + E(q_{biu}q_{biv}) - E(\mu_{iv}q_{biu}) \\ &\quad - E(\mu_{iu}q_{ai}M_v) - E(\mu_{iu}q_{biv}) + E(\mu_{iu}\mu_{iv}) \\ &= q_{ai}q_{ai} E(M_uM_v) + q_{ai}(q_{biv} - \mu_{iv}) E(M_u) + q_{ai}(q_{biu} - \mu_{iu}) E(M_v) + (q_{biu} - \mu_{iu})(q_{biv} - \mu_{iv}) \\ &= q_{ai}q_{ai} E(M_uM_v), \end{aligned}$$

where q_{biu} and μ_{iu} are the u -th components of vectors q_{bi} and μ_i respectively; and, since $E(M) = 0$,

$$\Sigma_i = q_{ai}^2 \Sigma,$$

and therefore

$$\det \Sigma_i = q_{ai}^{2d} \det \Sigma = q_{ai}^{2d}. \quad (5.167)$$

From the sequence of values $C = (c_1, c_2, \dots, c_N)$ we estimate the parameters for random variable V_b .

The sequence of values $(\det S_1, \det S_2, \dots, \det S_N)$ follows the random variable V_a^{2d} , since we had set $\det \Sigma = 1$. We can estimate V_a from the values $((\det S_i)^{1/2d})_i$.

Finally, we can then estimate M as follows: From each sample $X_i = (x_{ij})_j$, we calculate the normalized, centred sample

$$X'_i = \left(\frac{x_{ij} - \hat{\mu}_i}{(\det S_i)^{1/2d}} \right)_j, \quad (5.168)$$

where $\hat{\mu}_i$ is the mean of X_i and S_i is its covariance matrix. We aggregate the centred samples X'_i into a single sample X' (by taking all the elements in each X'_i , in an arbitrary order, with duplicates allowed). Using techniques such as those in section 5.5.1 we estimate M from the sample X' .

5.5.3 Flat random variables in a nested framework

Flat models are a subset of nested models. Some terrain classes are adequately modelled by flat models, and it is therefore necessary to be able to integrate such flat models into a nested model framework.

Let M be a random variable with values in \mathbb{R}^d and probability density function f_M .

Let us consider the random variable class Ξ , with values in \mathbb{R}^d , parametrized by a “parameter” such that the density of the instance $\Xi(q)$ is

$$\xi_\omega(q)(x) = f_M(x). \quad (5.169)$$

That is, the actual value of the parameter is ignored.

The random variable M can be estimated from the aggregate sample X as in section 5.5.1. Any random variable can be taken as the corresponding V variable—that which produces values of the parameter q —and need not be estimated. For practical reasons we suggest taking V as a “random” variable with values in $\{0\}$ and probability $p(V = 0) = 1$, $p(V = x) = 0$ for $x \neq 0$.

5.5.4 Model selection

As we did in section 5.5.1 for flat models, we have discussed in sections 5.5.2–5.5.3 how to choose the parameters for a random variable class model that make it optimally fit training data. The question remains, of which of the several random variable class models available to a particular implementation should be selected. As before, we shall use the Bayes Information Criterion to select the best model, balancing model complexity with goodness of fit:

$$BIC = -2 \ln p(X|T) + p \ln N,$$

where $p(X|T)$ is the probability of the observations given the model (the likelihood of the observations), p is the number of parameters in the model, and N is the number of observations. For a nested model (p, Ξ, V) , the number of parameters is that of Ξ , plus that of V , plus one to account for the selection of the one kind of random variable among many for V .

Among several random variable class models, the one with the lowest value of BIC will be selected.

The likelihood of the observations can be found, as in equation (5.47), as

$$p(R|c(R) = \omega) = \int_{q \in Q_\omega} v_\omega(q) \prod_{s \in R} \xi_\omega(q)(I(s)) dq. \quad (5.170)$$

Note that model selection for nested models is a two-level process. In one level, nested models are estimated using the random variable classes provided in the implementation, such as a translated random variable (section 5.5.2) or a scaled-and-translated random variable (section 5.5.2). In turn, in order to obtain each of these nested models, models for several random variables need to be estimated (for example, the M and V variables in section 5.5.2). Each of these random variables is estimated as in section 5.5.1, and every time the best random variable model is selected by BIC as in section 5.5.1. Afterwards, another model selection by BIC is among the nested models.

5.6 Implementation details

The direct computation of \hat{c} and \hat{r} following the equations given in section 5.4 is too slow for practical application. Fortunately, these equations can be transformed to make them easier to

compute.

5.6.1 Discretisation of the integral over Q_ω

Equations (5.45)–(5.55), (5.67), and (5.170) contain an integration over Q_ω , the domain of the parameter of a random variable class. In practice, Q_ω will be a finite set, often a subset of \mathbb{Z}^a for some a . In that case v_ω is a discrete probability function (with $\sum_{q \in Q_\omega} v_\omega(q) = 1$).

The integral in equations (5.45)–(5.49), (5.52), (5.53), (5.67), and (5.170) then takes the discrete form

$$\sum_{q \in Q_\omega} v_\omega(q) \prod_{s \in R} \xi_\omega(q)(I(s)) \quad (5.171)$$

instead of $\int_{q \in Q_\omega} v_\omega(q) \prod_{s \in R} \xi_\omega(q)(I(s)) dq$. Similarly, the integral in equations (5.50), (5.51), (5.54), and (5.55) takes the discrete form

$$\sum_{q \in Q_\omega} v_\omega(q) \left(\prod_{s \in R} \xi_\omega(q)(I(s)) \right)^{1/|R|} \quad (5.172)$$

instead of $\int_{q \in Q_\omega} v_\omega(q) \left(\prod_{s \in R} \xi_\omega(q)(I(s)) \right)^{1/|R|} dq$.

5.6.2 Discarding small values of $v_\omega(q)$

In addition, since for most values of q the variable $v_\omega(q)$ will take very small values, we can approximate equation (5.171) by the following, much faster to calculate

$$\sum_{q \in Q'_\omega} \alpha_\omega v_\omega(q) \prod_{s \in R} \xi_\omega(q)(I(s)), \quad (5.173)$$

and equation (5.172) by

$$\sum_{q \in Q'_\omega} \alpha_\omega v_\omega(q) \left(\prod_{s \in R} \xi_\omega(q)(I(s)) \right)^{1/|R|}, \quad (5.174)$$

where

$$Q'_\omega = \{q \in Q_\omega : v_\omega(q) > k_\omega\}, \quad (5.175)$$

$$k_\omega = \epsilon \cdot \max_{q \in Q_\omega} v_\omega(q) \quad (\text{a threshold}), \quad (5.176)$$

$$\alpha_\omega = \frac{\sum_{q \in Q_\omega} v_\omega(q)}{\sum_{q \in Q'_\omega} v_\omega(q)}, \quad (5.177)$$

$$\epsilon = 1/10 \quad (\text{or another small value}). \quad (5.178)$$

5.6.3 Scanning across \mathbf{S} instead of R

The term $p(R|\omega \cap q) = \prod_{s \in R} \xi_\omega(q)(I(s))$ is used in many equations in this chapter, including (5.52)–(5.55), (5.67), (5.100), (5.101), (5.108), (5.109), and (5.116)–(5.118), as well as (5.170), and their modified versions (5.171), (5.172), (5.173), and (5.174). If the number of pixels in the region to classify, $|R|$, is larger than the number of bins in a reasonably fine partition of the coordinate domain \mathbf{S} , it may be convenient to iterate across the coordinate domain instead of across the set of pixels in the region, using the following alternate calculation.

Let us construct a histogram of the pixel values $I(R) = (x_1, \dots, x_N)$, by partitioning the pixel value domain \mathbf{D} into a set of M equally-shaped bins, $\{D_1, \dots, D_M\}$, with $\cup_i D_i = \mathbf{D}$ and $\forall i \neq j, D_i \cap D_j = \emptyset$. The bin count for bin D_i is $\bar{d}_i = \text{card}\{j : x_j \in D_i\}$. Let p_i be a representing point for the bin D_i , for example its centre, and P the set of such points, $P = \{p_i : i \in \{1, \dots, M\}\}$.

We can then replace the term $\prod_{s \in R} \xi_\omega(q)(I(s))$ in these equations

$$p(R | \omega \cap q) = \prod_{s \in R} \xi_\omega(q)(I(s)) \simeq \prod_i \xi_\omega(q)(p_i)^{\bar{d}_i} = \exp \left(\sum_{i=1}^M \bar{d}_i \log \xi_\omega(q)(p_i) \right). \quad (5.179)$$

5.6.4 Calculating $p(R | \omega \cap q)$ using a Fourier transform

The approximation for $p(R | \omega \cap q)$ given by equation (5.179) can, for specific types of random variable classes, be calculated in a more efficient way using Fourier transforms. I propose alternate calculations for translated random variable classes and scaled-translated random variable classes (section 5.5.2).

Translated random variable classes

In this case, as defined by equation (5.160), we have $\xi_\omega(q)(x) = f_M(x - q)$. We define $\bar{d}(p_i)$ as a function of radiometry values as $\bar{d}(p_i) := \bar{d}_i = \text{card}\{j : x_j \in D_i\}$, and we also define

$$\tilde{m}(x) := \log f_M(-x). \quad (5.180)$$

Then, the sum in equation (5.179), which we will note $\eta(q)$, becomes

$$\eta(q) = \sum_{i=1}^M \bar{d}_i \log \xi_\omega(q)(p_i) = \sum_{i=1}^M \bar{d}_i \log f_M(p_i - q) = \sum_{x \in P} \bar{d}(x) \tilde{m}(q - x)$$

which can be expressed as the convolution

$$\eta(q) = (\bar{d} * \tilde{m})(q). \quad (5.181)$$

By the properties of the convolution and the Fourier transform (denoted by \mathcal{F}) we have

$$\eta = \bar{d} * \tilde{m} = \mathcal{F}^{-1}(\mathcal{F} \bar{d} \cdot \mathcal{F} \tilde{m}),$$

and we can then express equation (5.179) as

$$\exp \left(\sum_{i=1}^M \bar{d}_i \log \xi_\omega(q)(p_i) \right) = \exp \eta(q). \quad (5.182)$$

The advantage is that it is fast to calculate $\eta(q)$ for all values of q in a single step, by using direct and inverse fast Fourier transform (FFT), which can be a substantial improvement for equations (5.52)–(5.55), (5.67), (5.100), (5.101), (5.108), (5.109), and (5.116)–(5.118), as well as (5.170), and their modified versions (5.171), (5.172), (5.173), and (5.174), when used in the transformed form given by section 5.6.3.

Scaled and translated random variable classes

In this case, as defined by equation (5.164), we have $\xi_\omega(q_a, q_b)(x) = f_M((x - q_b)/q_a)$. We again define $\bar{d}(p_i)$ as a function of radiometry values as $\bar{d}(p_i) := \bar{d}_i$, and we also define

$$\tilde{m}_{q_a}(x) := \log f_M\left(-\frac{x}{q_a}\right). \quad (5.183)$$

Then, the sum in equation (5.179), which we will note $\eta(q_a, q_b)$, becomes

$$\eta(q_a, q_b) = \sum_{i=1}^M \bar{d}_i \log \xi_\omega(q)(p_i) = \sum_{i=1}^M \bar{d}_i \log f_M\left(\frac{p_i - q_b}{q_a}\right) = \sum_{x \in P} \bar{d}(x) \tilde{m}_{q_a}(q_b - x)$$

which can be expressed as the convolution

$$\eta(q_a, q_b) = (\bar{d} * \tilde{m}_{q_a})(q_b). \quad (5.184)$$

By the properties of the convolution and the Fourier transform we have

$$\eta(q_a, q_b) = (\bar{d} * \tilde{m}_{q_a})(q_b) = (\mathcal{F}^{-1}(\mathcal{F} \bar{d} \cdot \mathcal{F} \tilde{m}_{q_a}))(q_b), \quad (5.185)$$

and we can then express equation (5.179) as

$$\exp\left(\sum_{i=1}^M \bar{d}_i \log \xi_\omega(q)(p_i)\right) = \exp \eta(q_a, q_b). \quad (5.186)$$

Unlike equation (5.182), the convolution by Fourier transform of equation (5.185) provides $\eta(q_a, q_b)$ for all values of q_b , but for a single value of q_a only. Still, this can be an important improvement.

5.6.5 Minimising the Kullback-Leibler divergence using a Fourier transform

In equations (5.82) and (5.94) the term $\min_{q \in \mathcal{Q}_\omega} d_{\text{KL}}(I(R), \Xi_\omega(q))$ is very slow to calculate. The idea of section 5.6.4 can also be applied to this case to render its calculation fast enough for practical applications. Again, I propose alternate calculations for translated random variable classes (section 5.5.2) and scaled-translated random variable classes (section 5.5.2).

As in sections 5.6.3 and 5.6.4, the pixel value domain \mathbf{D} is partitioned into a set of M equally-shaped bins, $\{D_1, \dots, D_M\}$, with $\cup_i D_i = \mathbf{D}$ and $\forall i \neq j. D_i \cap D_j = \emptyset$; p_i is a representing point for the bin D_i , for example its center, and P the set of such points, $P = \{p_i : i \in \{1, \dots, M\}\}$. The bin count for bin D_i is $\bar{d}_i = \text{card}\{j : x_j \in D_i\}$.

Let a be the measure of a bin (they all have the same shape and, therefore, measure), that is, since $\mathbf{D} \subset \mathbb{Z}^d$

$$a = \text{card } D_i. \quad (5.187)$$

We define

$$h(p_i) := \frac{\bar{d}_i}{N}, \quad (5.188)$$

where N is the number of pixels in the region being classified, and we note that

$$p(\Xi_\omega(q) \in D_i) = \sum_{z \in D_i} \xi_\omega(q)(z) \simeq a \xi_\omega(q)(p_i). \quad (5.189)$$

We can then express the minimisation of the Kullback-Leibler divergence of equations (5.82) and (5.94) as

$$\min_{q \in Q_\omega} d_{\text{KL}}(I(R), \Xi_\omega(q)) \simeq \min_{q \in Q_\omega} 2 \sum_{x \in P} \left(h(x) \log h(x) - h(x) \log a \xi_\omega(q)(x) \right). \quad (5.190)$$

Translated random variable classes

In this case we have $\xi_\omega(q)(x) = f_M(x - q)$. Defining

$$\tilde{m}(x) := \log f_M(-x), \quad (5.191)$$

we can rewrite equation (5.190) as

$$\min_{q \in Q_\omega} d_{\text{KL}}(I(R), \Xi_\omega(q)) \simeq 2 \sum_{x \in P} h(x) \log h(x) - \max_{q \in Q_\omega} 2 \log a \sum_{x \in P} h(x) \tilde{m}(q - x),$$

which, as in section 5.6.4 can be expressed as a convolution,

$$= 2 \sum_{x \in P} h(x) \log h(x) - 2 \log a \max_{q \in Q_\omega} (h * \tilde{m})(q). \quad (5.192)$$

The convolution $h * \tilde{m}$ can be calculated by Fourier transform as

$$h * \tilde{m} = \mathcal{F}^{-1}(\mathcal{F}h \cdot \mathcal{F}\tilde{m}), \quad (5.193)$$

which allows for fast computation of $(h * \tilde{m})(q)$ for all values of q in a single step, thereby speeding up the calculation of equations (5.82) and (5.94).

Scaled and translated random variable classes

In this case we have $\xi_\omega(q_a, q_b)(x) = f_M((x - q_b)/q_a)$. Defining

$$\tilde{m}_{q_a}(x) := \log f_M\left(-\frac{x}{q_a}\right), \quad (5.194)$$

we can rewrite equation (5.190) as

$$\min_{(q_a, q_b) \in Q_\omega} d_{\text{KL}}(I(R), \Xi_\omega(q_a, q_b)) \simeq 2 \sum_{x \in P} h(x) \log h(x) - \max_{(q_a, q_b) \in Q_\omega} 2 \log a \sum_{x \in P} h(x) \tilde{m}_{q_a}(q_b - x),$$

which, as in section 5.6.4 can be expressed as a convolution,

$$= 2 \sum_{x \in P} h(x) \log h(x) - 2 \log a \max_{(q_a, q_b) \in Q_\omega} (h * \tilde{m}_{q_a})(q_b). \quad (5.195)$$

Again, the convolution $h * \tilde{m}_{q_a}$ can be calculated by Fourier transform as

$$h * \tilde{m}_{q_a} = \mathcal{F}^{-1}(\mathcal{F}h \cdot \mathcal{F}\tilde{m}_{q_a}), \quad (5.196)$$

which allows for fast computation of $(h * \tilde{m}_{q_a})(q_b)$ for all values of q_b in a single step. A separate step is still needed for each value of q_a .

5.6.6 Minimising the symmetric Kullback-Leibler divergence by Fourier transform

A similar technique can be used to quickly minimise the term $\min_{q \in Q_\omega} d_{\text{KLS}}(I(R), \Xi_\omega(q))$ of equations (5.89) and (5.95), which uses the symmetric Kullback-Leibler divergence.

With the same notation as in section 5.6.5, we can then express the minimisation of the symmetric Kullback-Leibler divergence as

$$\begin{aligned} \min_{q \in Q_\omega} d_{\text{KLS}}(I(R), \Xi_\omega(q)) \\ \simeq \min_{q \in Q_\omega} \sum_{x \in P} \left(h(x) \log h(x) - h(x) \log a \xi_\omega(q)(x) + a \xi_\omega(q)(x) \log(a \xi_\omega(q)(x)) - a \xi_\omega(q)(x) \log h(x) \right). \end{aligned} \quad (5.197)$$

For translated random variable classes, we have $\xi_\omega(q)(x) = f_M(x - q)$. We can therefore express equation (5.198) using convolutions and function compositions ($a \circ b$) as

$$\sum_{x \in P} h(x) \log h(x) + a \log a \sum_{x \in P} f_M(x) \log f_M(x) - \max_{q \in Q_\omega} \left((\log a) \cdot h * (\log \circ f_M) + a \cdot (\log \circ h) * m \right)(q), \quad (5.198)$$

and solve it as in section 5.6.5. As we have shown in that section, this can also be extended to scaled and translated random variable classes.

5.6.7 Minimising the χ^2 statistic

Equations (5.83), (5.84), (5.96), and (5.97) are also very slow to calculate, but their calculation could be significantly speeded up if the term

$$\min_{q \in Q_\omega} \chi^2(I(R), \Xi_\omega(q)) \quad (5.199)$$

could be calculated in a simpler way.

In this section I propose faster alternate calculations for translated random variable classes and for scaled-translated random variable classes (section 5.5.2).

We keep the definitions from sections 5.6.3, 5.6.4, and 5.6.5 for D_i , p_i , P , a , N , and $h(x)$, and we recall the approximation of equation (5.189).

We can then express the minimisation of the χ^2 statistic of equation (5.199) as

$$\begin{aligned} \min_{q \in Q_\omega} \chi^2(I(R), \Xi_\omega(q)) &\simeq \min_{q \in Q_\omega} N \sum_{x \in P} \frac{(h(x) - a \xi_\omega(q)(x))^2}{a \xi_\omega(q)(x)} \\ &= N \min_{q \in Q_\omega} \left(\frac{1}{a} \sum_{x \in P} \frac{h(x)^2}{\xi_\omega(q)(x)} + a \sum_{x \in P} \xi_\omega(q)(x) \right) - 2N. \end{aligned} \quad (5.200)$$

First term

To calculate the term $\sum_{x \in P} \frac{h(x)^2}{\xi_\omega(q)(x)}$ quickly enough we may again use the convolution technique of section 5.6.4.

Let us define

$$\tilde{h}(x) := h(x)^2. \quad (5.201)$$

For a translated random variable class, we have $\xi_\omega(q)(x) = f_M(x - q)$. We define

$$\tilde{m}(x) := \frac{1}{f_M(-x)} \quad (5.202)$$

and rewrite

$$\sum_{x \in P} \frac{h(x)^2}{\xi_\omega(q)(x)} = \sum_{x \in P} \tilde{h}(x) \tilde{m}(q - x) = (\tilde{h} * \tilde{m})(q) = (\mathcal{F}^{-1}(\mathcal{F}\tilde{h} \cdot \mathcal{F}\tilde{m}))(q). \quad (5.203)$$

The term $\sum_{x \in P} \frac{h(x)^2}{\xi_\omega(q)(x)}$ can thus be calculated for all values of q in a single, fast, step.

For a scaled and translated random variable class, we have $\xi_\omega(q_a, q_b)(x) = f_M((x - q_b)/q_a)$. We define

$$\tilde{m}_{q_a}(x) := \frac{1}{f_M(-x/q_a)} \quad (5.204)$$

and rewrite

$$\sum_{x \in P} \frac{h(x)^2}{\xi_\omega(q)(x)} = \sum_{x \in P} \tilde{h}(x) \tilde{m}_{q_a}(q_b - x) = (\tilde{h} * \tilde{m}_{q_a})(q_b) = (\mathcal{F}^{-1}(\mathcal{F}\tilde{h} \cdot \mathcal{F}\tilde{m}_{q_a}))(q_b). \quad (5.205)$$

The term $\sum_{x \in P} \frac{h(x)^2}{\xi_\omega(q_a, q_b)(x)}$ can thus be calculated for all values of q_b in a single step. As before, a separate step is still needed for each value of q_a .

Second term

To calculate the term $\sum_{x \in P} \xi_\omega(q)(x)$ faster, we need to use a different approach.

For a translated random variable class, we have $\xi_\omega(q)(x) = f_M(x - q)$. Therefore,

$$\sum_{x \in P} \xi_\omega(q)(x) = \sum_{x \in P} f_M(x - q) = \sum_{y \in P - q} f_M(y), \quad (5.206)$$

where by the notation $P - q$ we mean the translation of the points in P by an offset q , that is, the set

$$P - q := \{x - q : x \in P\}. \quad (5.207)$$

In our implementation, the points in P are laid out in a regular grid. The spacing between adjacent points is δ_i in the i -th dimension, and we define an "origin" $(x_{o1}, x_{o2}, \dots, x_{od})$. Then, each point $p_i \in P$ is $(x_{o1} + c_{i1}\delta_1, x_{o2} + c_{i2}\delta_2, \dots, x_{od} + c_{id}\delta_d)$, for some values of $c_i = (c_{i1}, c_{i2}, \dots, c_{id}) \in \mathbb{N}^d$. Let us note, in this section,

$$\pi(n_1, n_2, \dots, n_d) = (x_{o1} + n_1\delta_1, x_{o2} + n_2\delta_2, \dots, x_{od} + n_d\delta_d), \quad (5.208)$$

and π^{-1} its inverse function.

Let us then define the cumulative sum

$$\bar{m}(n_1, n_2, \dots, n_d) := \sum_{u_1 \leq n_1 \wedge u_2 \leq n_2 \wedge \dots \wedge u_d \leq n_d} f_M(\pi(u_1, u_2, \dots, u_d)). \quad (5.209)$$

This cumulative sum \bar{m} can be easily pre-calculated in d steps by separately performing sums

along each dimension as follows:

$$\begin{aligned}
\mu_0(n_1, n_2, \dots, n_d) &= f_M(\pi(n_1, n_2, \dots, n_d)), \\
\mu_1(0, n_2, \dots, n_d) &= \mu_0(0, n_2, \dots, n_d), \\
\mu_1(1 + b, n_2, \dots, n_d) &= \mu_0(1 + b, n_2, \dots, n_d) + \mu_1(b, n_2, \dots, n_d), \\
\mu_2(n_1, 0, \dots, n_d) &= \mu_1(n_1, 0, \dots, n_d), \\
\mu_2(n_1, 1 + b, \dots, n_d) &= \mu_1(n_1, 1 + b, \dots, n_d) + \mu_2(n_1, b, \dots, n_d), \\
&\dots \\
\mu_d(n_1, n_2, \dots, 0) &= \mu_{d-1}(n_1, n_2, \dots, 0), \\
\mu_d(n_1, n_2, \dots, 1 + b) &= \mu_{d-1}(n_1, n_2, \dots, 1 + b) + \mu_d(n_1, n_2, \dots, b), \\
\bar{m}(n_1, n_2, \dots, n_d) &= \mu_d(n_1, n_2, \dots, n_d).
\end{aligned} \tag{5.210}$$

In our implementation, values of q are quantized in such a way that $\pi^{-1}(x - q)$ is defined for each $x \in P$. Let $C = \pi^{-1}(P - q)$ be the image of $P - q$ by π^{-1} . We have $C = \{c_1, \dots, c_1 + w_1\} \times \{c_2, \dots, c_2 + w_2\} \times \dots \times \{c_d, \dots, c_d + w_d\} \subset \mathbb{N}^d$. Equation (5.206) can then be rewritten as

$$\sum_{x \in P} f_M(x - q) = \sum_{y \in P - q} \mu_0(\pi^{-1}(y)) = \sum_{(n_1, n_2, \dots, n_d) \in C} \mu_0(n_1, n_2, \dots, n_d), \tag{5.211}$$

which can be obtained from the pre-calculated values of \bar{m} as

$$\sum_{(n_1, n_2, \dots, n_d) \in C} \mu_0(n_1, n_2, \dots, n_d) = \sum_{(i_1, i_2, \dots, i_d) \in \{0, 1\}^d} s(i_1, i_2, \dots, i_d) \bar{m}(c_1 + i_1 w_1, c_2 + i_2 w_2, \dots, c_d + i_d w_d), \tag{5.212}$$

$$s(i_1, i_2, \dots, i_d) = \begin{cases} +1 & \text{if } d - \sum_k i_k \text{ is even,} \\ -1 & \text{if } d - \sum_k i_k \text{ is odd.} \end{cases} \tag{5.213}$$

For a scaled and translated random variable class, we have $\xi_{\omega}(q_a, q_b)(x) = f_M((x - q_b)/q_a)$. Therefore,

$$\sum_{x \in P} \xi_{\omega}(q_a, q_b)(x) = \sum_{x \in P} f_M\left(\frac{x - q_b}{q_a}\right) = \sum_{y \in (P - q_b)/q_a} f_M(y), \tag{5.214}$$

where by the notation $(P - q_b)/q_a$ we mean the set $\{(x - q_b)/q_a : x \in P\}$. In our implementation, q_a and q_b are quantized in such a way that $\pi^{-1}((x - q_b)/q_a)$ is defined for each $x \in P$. We can define $C = \pi^{-1}((P - q_b)/q_a)$ and use the same technique as for translated random variable classes.

5.7 Evaluation of the model estimation

In previous sections the new nested probability model for land cover classification has been presented. After shortly reviewing traditional per-pixel Bayesian classification, we have discussed classification algorithms using non-nested and nested per-region classification, and we have studied methods for estimating nested models from training data.

In this and the next two sections we will put these methods into practical use. In this section we describe how we used training data on our available test sites to estimate random models. These models will be used in section 5.9 to classify the terrain in these test sites. Section 5.8 reproduces the results of [TSB05], which show a more limited use of non-nested per-region classification, but with results which are very interesting for cartography applications.

We have estimated different probability models from the available training data depending on a number of factors: training set, input channels, and type of models allowed.

First, we have estimated different models depending on the training set. Two sets are available. The first one, *Saint-Léger*, contains images at 50 cm per pixel resolution, with red, green, and blue channels. These images were taken with an analog camera, scanned at a resolution of 80 cm per pixel, and resampled to 50 cm. This test set corresponds to 5.7 km² of relatively simple terrain, with forests, fields, and some orchards. The second data set, *Toulouse*, contains images at 50 cm per pixel, with red, green, blue, and near-infrared channels, taken with a digital camera. This test set corresponds to a 37.6 km² area of more complex terrain, which includes forest, fields, and orchards, but also rivers, ponds, vineyards, quarries, and bare soil. Only for 7.0 km² of this set was cadastre data available, which, for the evaluation of the classification algorithms, means that results for this site using the registered cadastre as input regions is less significant. Images in *Toulouse* had undergone a radiometry correction to remove the *hot spot*.

Training and test polygons are defined on both data sets, for the following terrain classes:

Field Wild or planted grass, or other low or middle-height crop. Does not include trees — wild or planted— nor bushes. Includes both active vegetation with green colour, and brown-coloured vegetation if it is apparent that vegetation is indeed present.

Forest Wild forest, consisting in spontaneously growing trees in a random layout. Includes evergreen forests, and deciduous forests both with and without leaves. It also includes hedges and shrubs, given the difficulty of discriminating between these classes and the fact that this discrimination is of little interest to us.

Orchard Trees laid out in a regular grid. Includes fruit tree plantations, timber plantations, and also areas, reforested for ecological purposes, with a regular layout. Trees in orchards can be on a green background —undergrowth, typical of timber and reforested areas— or on a brown background ground —ground, typical of fruit trees.

Quarry Large open-air rock area. In spite of the name, the system will not distinguish between quarries and natural rocky areas.

Lake Lakes, reservoirs, swimming pools, and other areas with still water.

River Rivers, channels, and other areas with flowing water.

Tillable Homogeneous large brownish area, which does not contain any vegetation but shows bare soil which appears suitable for tilling and growing crops. Because we established terrain references from old topographic maps and from the original images only, without on-the-ground surveying, it is sometimes difficult to distinguish between the *tillable* class and *fields* containing brown plants, the main distinctive feature then being the presence or absence of tills. Because of this, in the evaluation phase of section 5.9 these two classes are merged into one.

Sand Wet or dry sand —typically on a river bank or beach— or bare soil which does not appear suitable for tilling and growing crops.

Vineyard A vineyard, a plantation of vines. This looks the same as an orchard on brown ground, except that the plant size and the spacing between adjacent rows of plants is smaller in vineyards than in orchards. In chapter 6 we present a method specifically aimed at distinguishing these two classes.

In addition, I defined two more classes in the reference: *built areas* (roofs) and *roads* (including parking lots). However, I have not estimated models for them, the classifiers cannot give them

as output classes, and they are not taken into account for the evaluations of section 5.9. This is because, thanks to the cadastre data, the system knows beforehand the position of roads and buildings. Since the system is anyway not supposed to be used in urban areas, I think that using it for these purposes does not give any added value, while possibly decreasing the system's performance on its target context, rural areas. The special *cadastre built* class is given to pixels that the cadastre marks as belonging or being within a certain distance of a building, regardless of their actual radiometry and texture.

In addition to which data set is used, it is also necessary to decide which channels will be used for classification. Not only the raw radiometry channels are available, but also all the transformed colour channels and texture features of appendix A. Following suggestions in [Gif04], for Saint-Léger I use the *pix-ent-c* channel of section A.2.4, the raw red channel, and, optionally, the second Karhunen-Loève channel *kl2* of equation (A.13) in section A.1.6. For Toulouse I use the same *pix-ent-c* channel, the Normalized Difference Vegetation Index (NDVI) of equation (A.17) in section A.1.9, and, optionally, the *pix-ent-lg-h* channel of section A.2.4. In addition, we may take the data domain \mathbf{D} to be $\mathbf{D} = \{0, \dots, 255\}^d$, as expected, or an extended domain $\mathbf{D} = \{-64, \dots, 320\}^d$. I tried the latter because I suspected some estimation problems with parametric models having a support which significantly falls outside the data domain.

These combinations, including the name by which they will be referenced in the remainder of this chapter, are listed in table 5.1. Note that not all possible combinations will be evaluated.

name	data set	channels	data domain
stl2	Saint-Léger	<i>pix-ent-c, red</i>	$\mathbf{D} = \{0, \dots, 255\}^2$
stl3	Saint-Léger	<i>pix-ent-c, red, kl2</i>	$\mathbf{D} = \{0, \dots, 255\}^3$
stl2x	Saint-Léger	<i>pix-ent-c, red</i>	$\mathbf{D} = \{-64, \dots, 320\}^2$
stl3x	Saint-Léger	<i>pix-ent-c, red, kl2</i>	$\mathbf{D} = \{-64, \dots, 320\}^3$
tor2	Toulouse	<i>ndvi, pix-ent-c</i>	$\mathbf{D} = \{0, \dots, 255\}^2$
tor3	Toulouse	<i>ndvi, pix-ent-c, pix-ent-lg-h</i>	$\mathbf{D} = \{0, \dots, 255\}^3$
tor2x	Toulouse	<i>ndvi, pix-ent-c</i>	$\mathbf{D} = \{-64, \dots, 320\}^2$
tor3x	Toulouse	<i>ndvi, pix-ent-c, pix-ent-lg-h</i>	$\mathbf{D} = \{-64, \dots, 320\}^3$

Table 5.1: Combinations of test sets and input channels for model estimation.

Figures 5.6 to 5.13 show these test sites. We give the red-green-blue image for each of the sites, the near-infrared image for the Toulouse site, and the three-channel composite corresponding to the *stl3* and *tor3* configurations. The Toulouse site is shown split in two halves.

Figures 5.14 to 5.17 show some examples of the distribution of pixel values for the Toulouse site. Figures 5.14 and 5.15 are two-variable histograms, showing the relative frequency of each combination of pixel values for the *tor2* configuration. The x axis (to the right of the page) is the *ndvi* transformed colour channel, and the y axis (perpendicular to the page) is the *pix-ent-c* texture descriptor. Relative frequency is given by the height of the bars. The figures show the distributions of some terrain classes separately, and for all classes together.

Figures 5.16 and 5.17 are three-variable histograms for the *tor3* configuration. The x and y axes are as before, and the z axis (to the top of the page) is the *pix-ent-lg-h* texture descriptor. Relative frequency is given by the size of the cubes.

Note that, although in many remote sensing classification applications probability distributions are assumed to be Gaussian, for some terrain types the distributions shown in figures 5.14 to 5.17 certainly are not.

Another factor which affects model estimation is which kind of submodels are used for the estimation of simple random variables (i.e., the Gaussian, Laplacian, rectangular, non-parametric, and KDE models of section 5.5.1), whether we should estimate flat or nested models, and



Figure 5.6: Saint-Léger test site. True-colour image, showing the red, green, and blue channels in the original image as red, green, and blue in this document. North is to the left of the page.

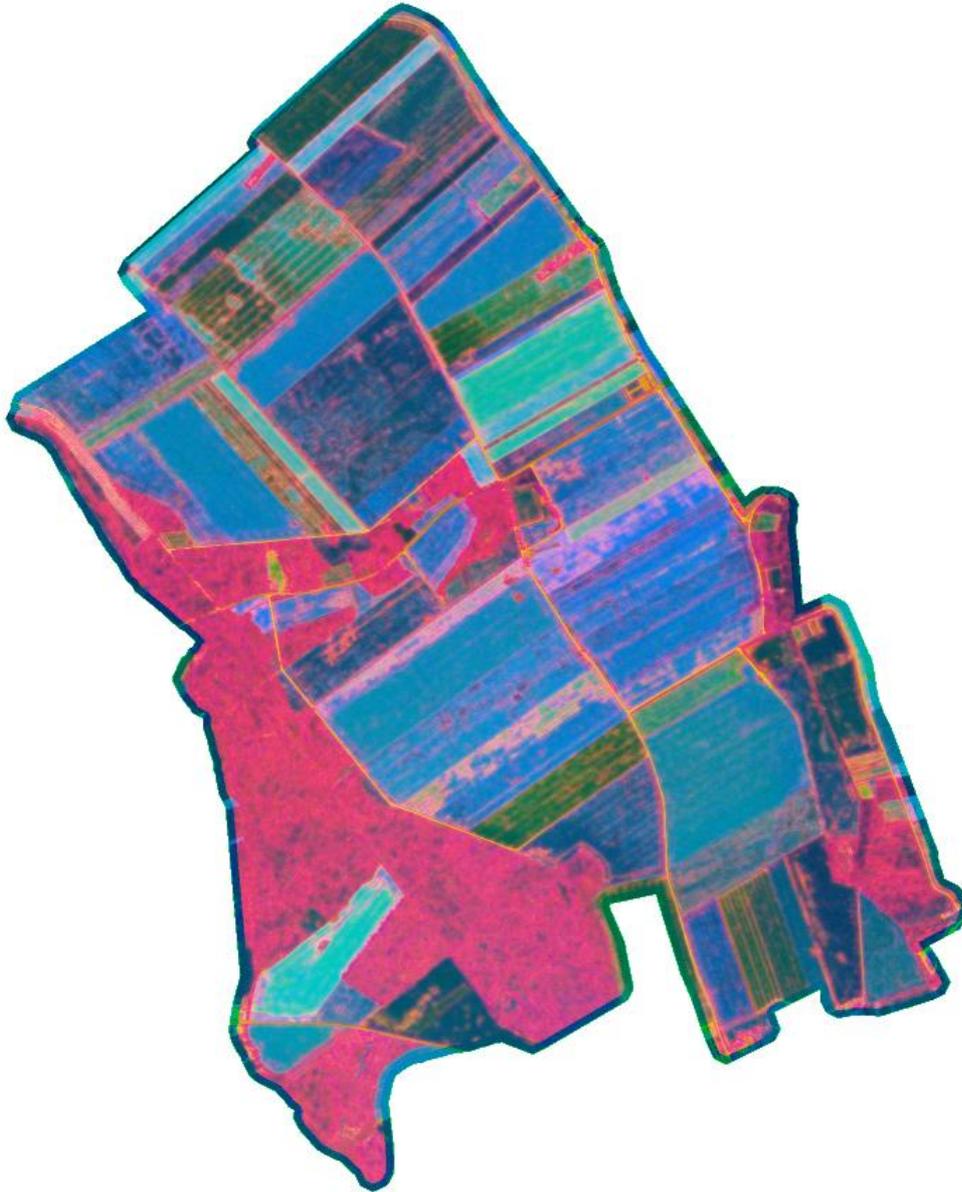


Figure 5.7: Saint-Léger test site. False-colour image for the *stl3* channel set, with the *pix-ent-c*, *red*, and *kl2* channels respectively shown as red, green, and blue in this document. North is to the left of the page.



Figure 5.8: Western half of the Toulouse test site. True-colour image, showing the red, green, and blue channels in the original image as red, green, and blue in this document. North is to the left of the page.

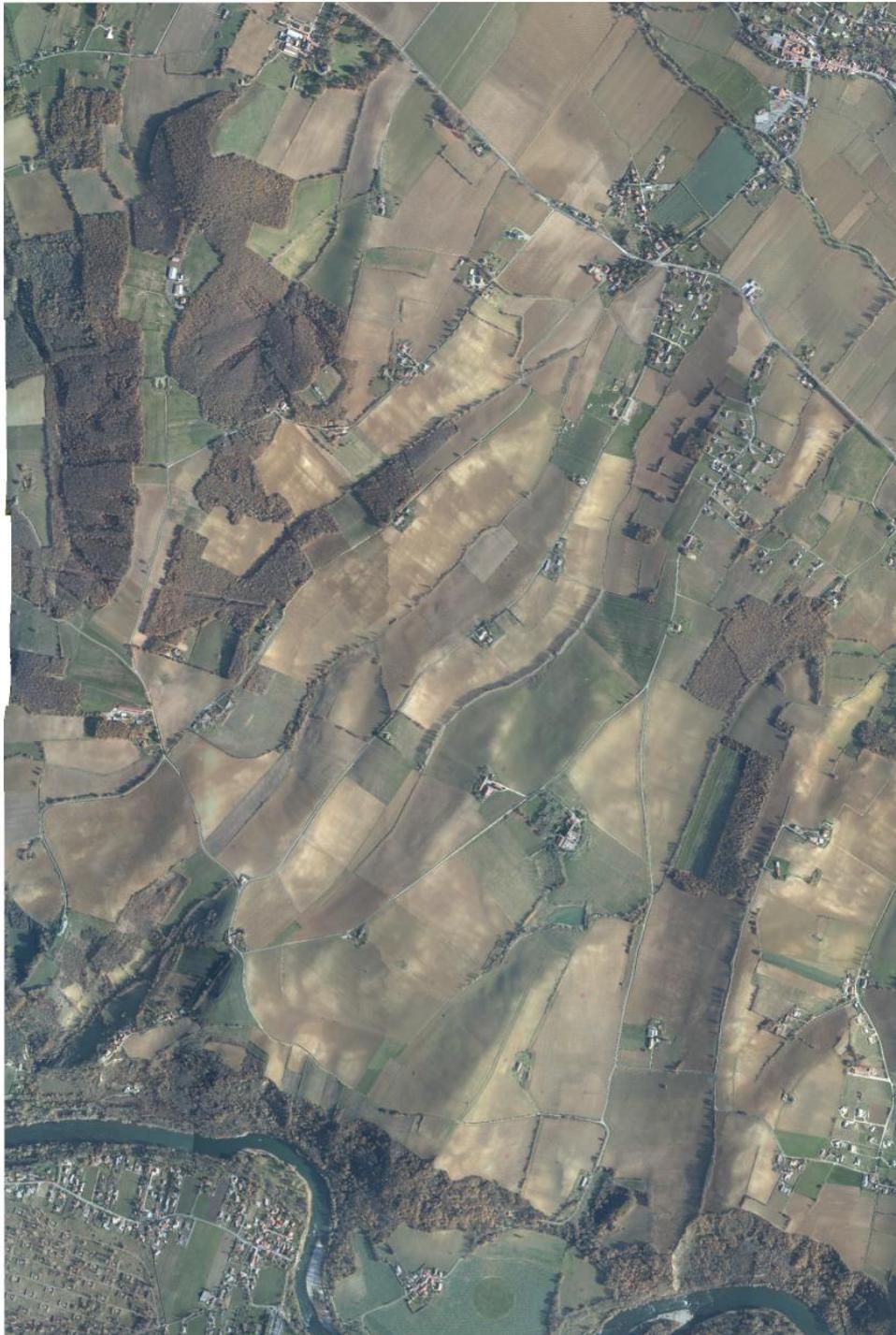


Figure 5.9: Eastern half of the Toulouse test site. True-colour image, showing the red, green, and blue channels in the original image as red, green, and blue in this document. North is to the left of the page.



Figure 5.10: Western half of the Toulouse test site. Near-infrared channel. North is to the left of the page.



Figure 5.11: Eastern half of the Toulouse test site. Near-infrared channel. North is to the left of the page.

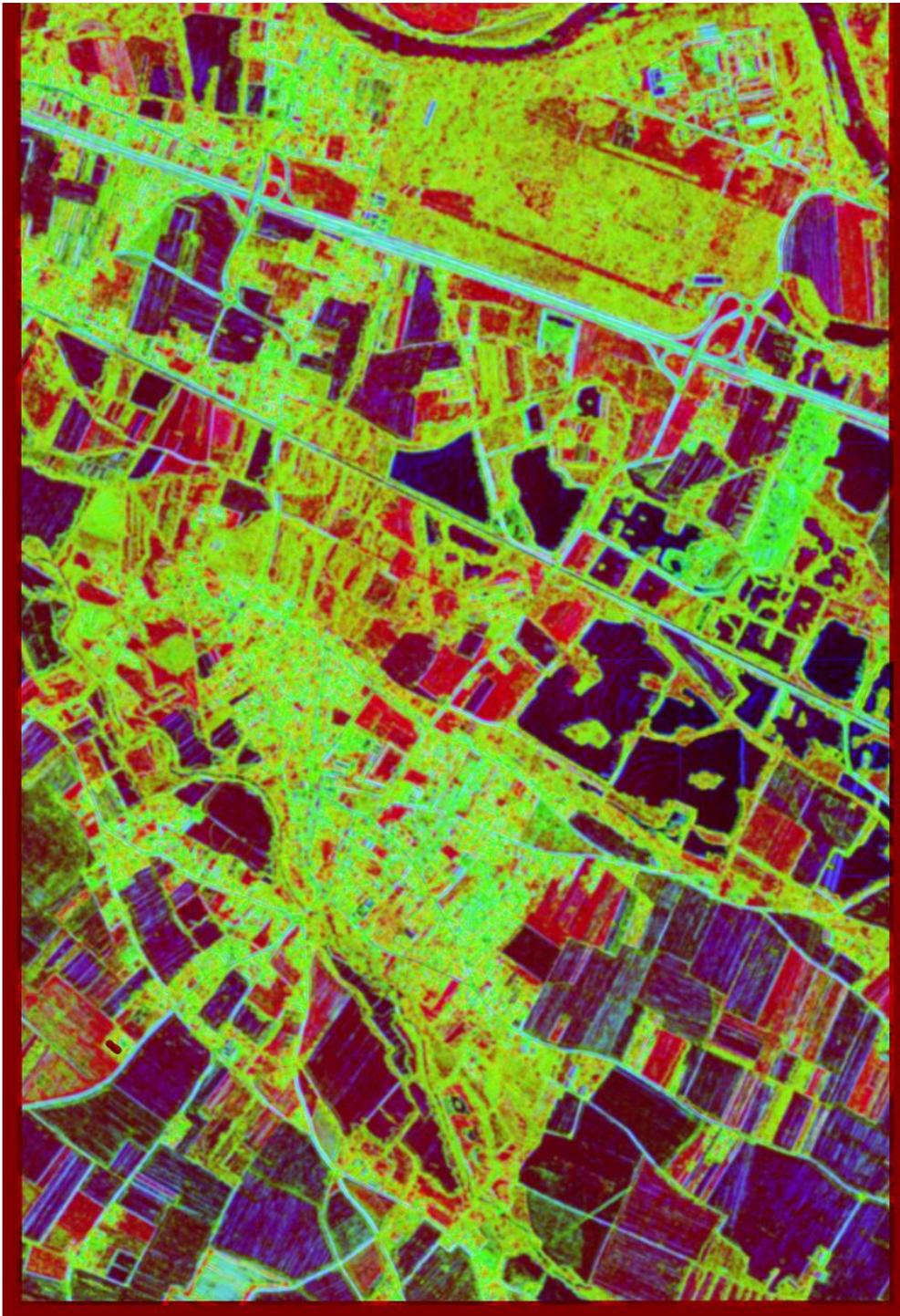


Figure 5.12: Western half of the Toulouse test site. False-colour image for the *tor3* channel set, with the *ndvi*, *pix-ent-c*, and *pix-ent-lg-h* channels respectively shown as red, green, and blue in this document. North is to the left of the page.

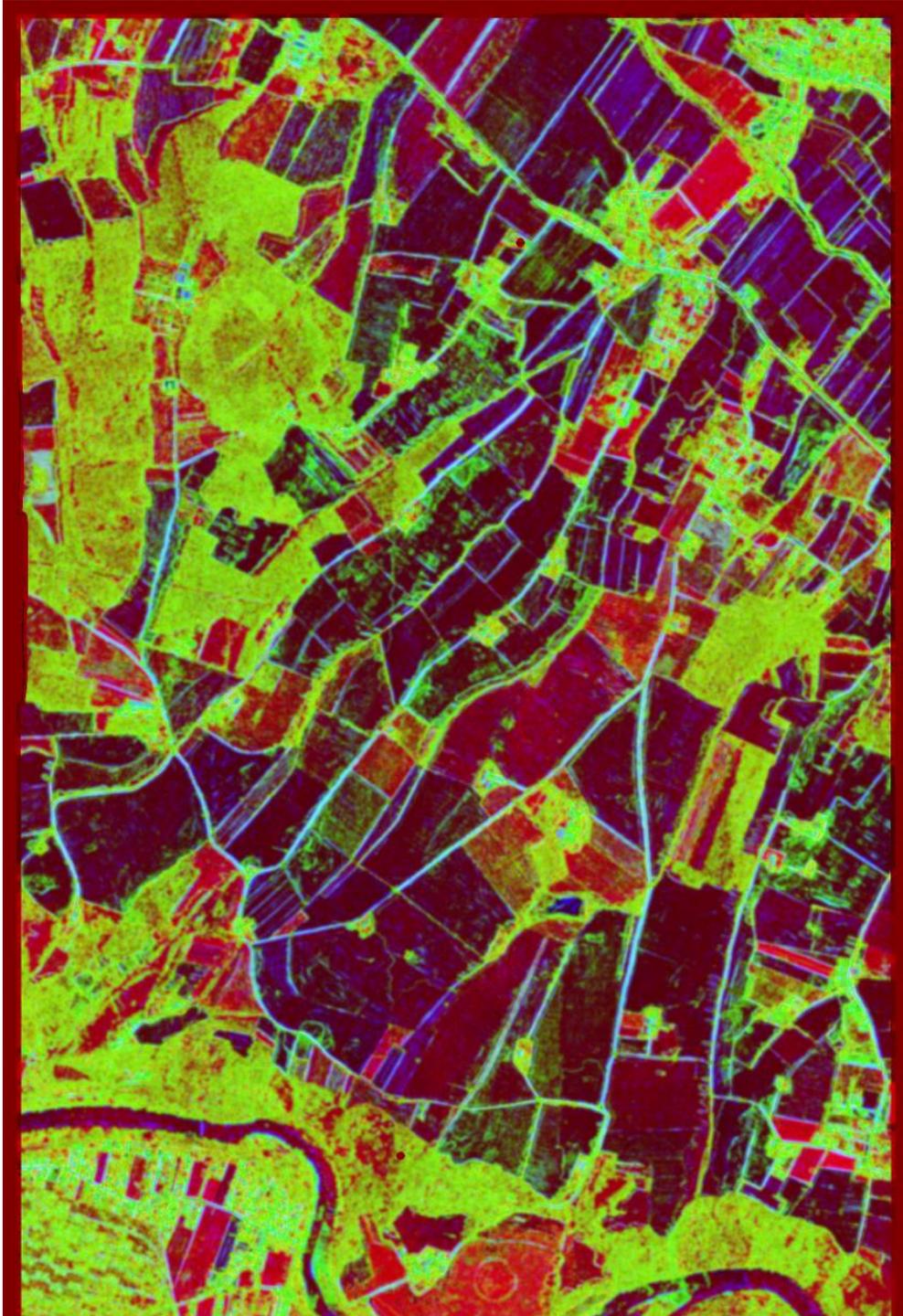


Figure 5.13: Eastern half of the Toulouse test site. False-colour image for the *tor3* channel set, with the *ndvi*, *pix-ent-c*, and *pix-ent-lg-h* channels respectively shown as red, green, and blue in this document. North is to the left of the page.

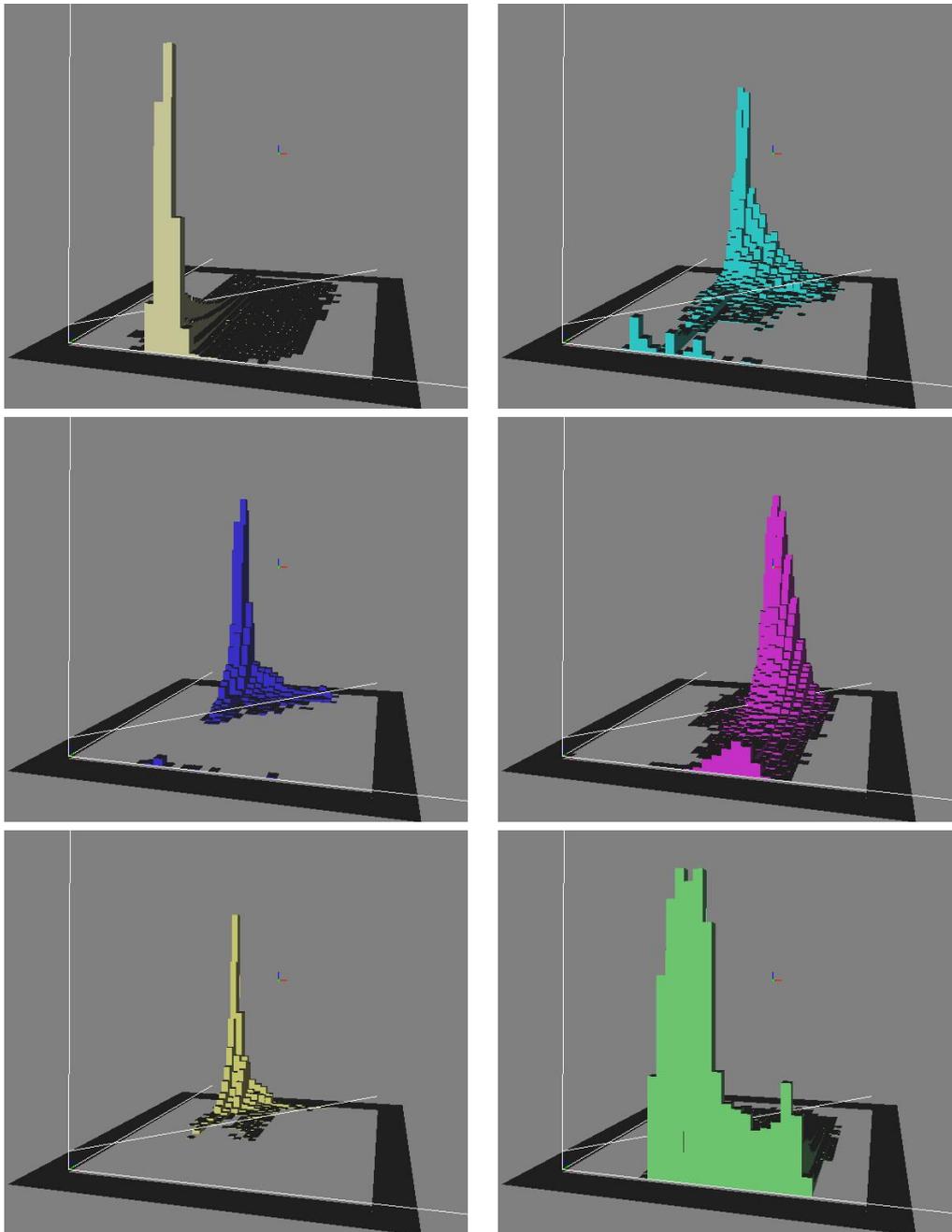


Figure 5.14: Distribution of pixel values for some classes of the Toulouse site. The x axis (to the right of the page) is the $ndvi$ channel, and the y axis (perpendicular to the page) is the $pix-ent-c$ channel. From left to right and top to bottom: *tillable*, *built area*; *road*, *forest*; *quarry*, and *field* terrain classes.

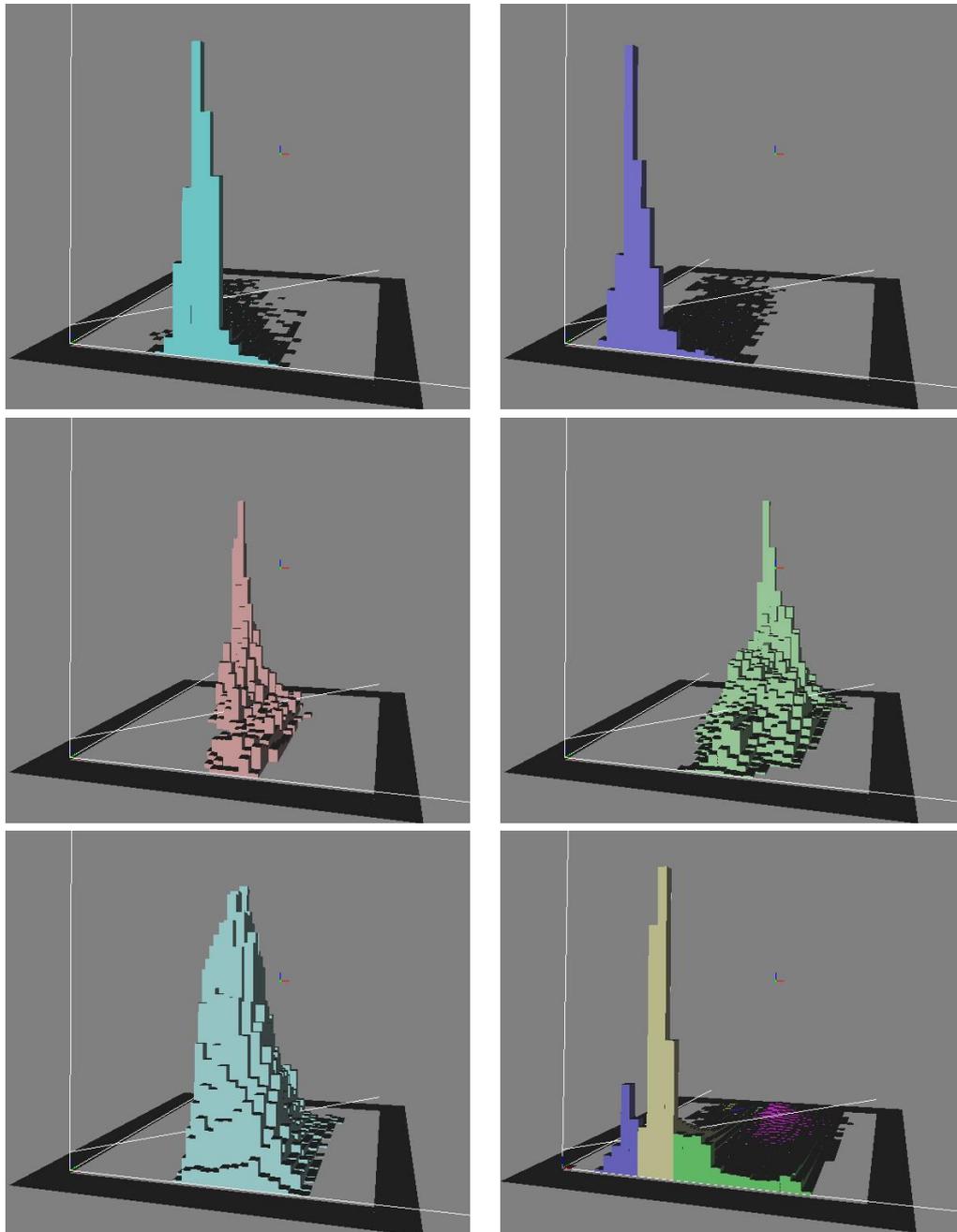


Figure 5.15: Distribution of pixel values for some classes of the Toulouse site. The x axis (to the right of the page) is the *ndvi* channel, and the y axis (perpendicular to the page) is the *pix-ent-c* channel. From left to right and top to bottom: *river*, *lake*; *sand*, *orchard*; *vineyard*, and all terrain classes together, rescaled so that the vertical axis is the global relative frequency instead of the per-class relative frequency.

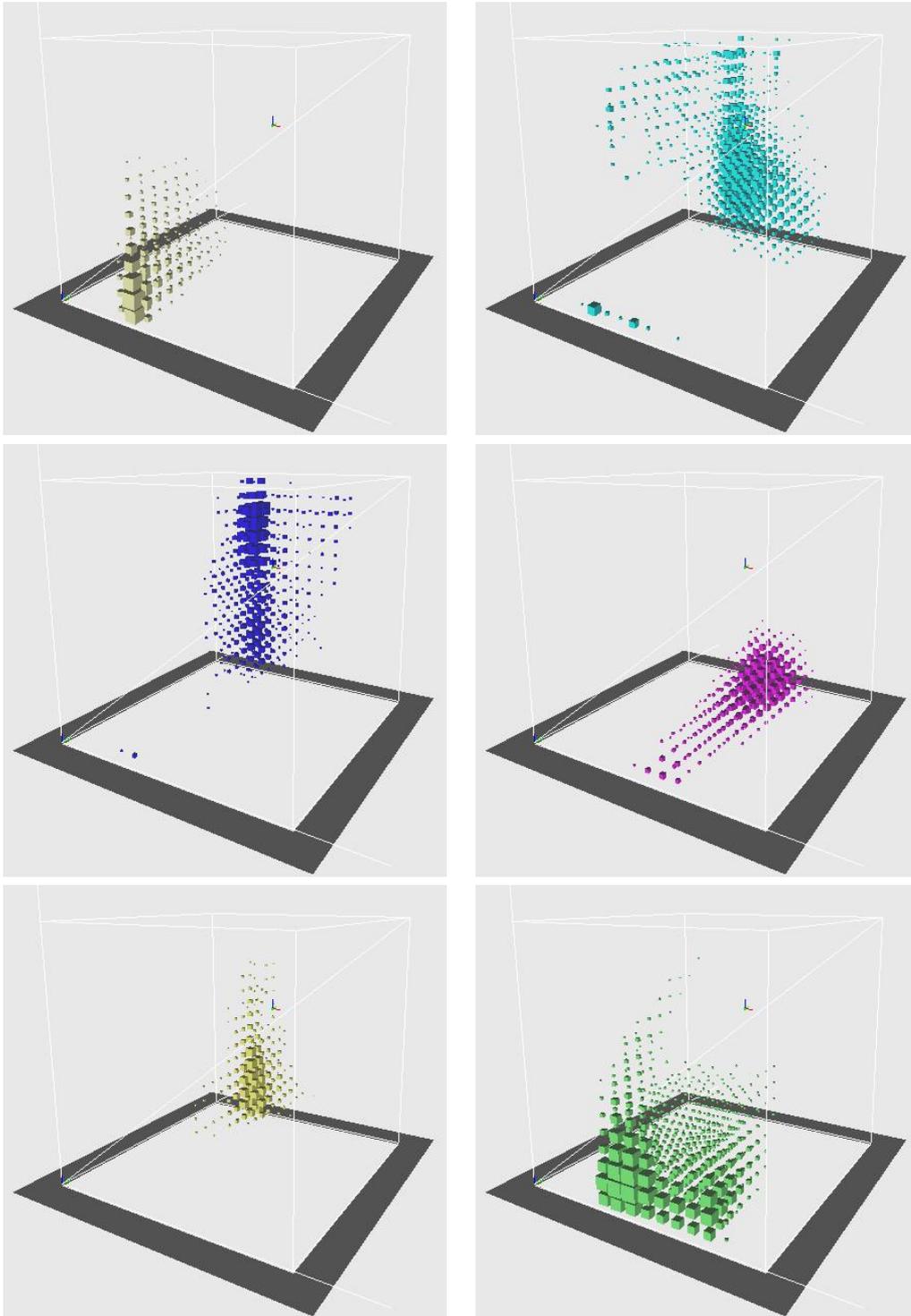


Figure 5.16: Distribution of pixel values for some classes of the Toulouse site. The x axis (to the right of the page) is the $ndvi$ channel, the y axis (perpendicular to the page) is the $pix-ent-c$ channel, and the z axis (to the top of the page) is the $pix-ent-lg-h$ channel. From left to right and top to bottom: *tillable*, *built area*; *road*, *forest*; *quarry*, and *field terrain* classes.

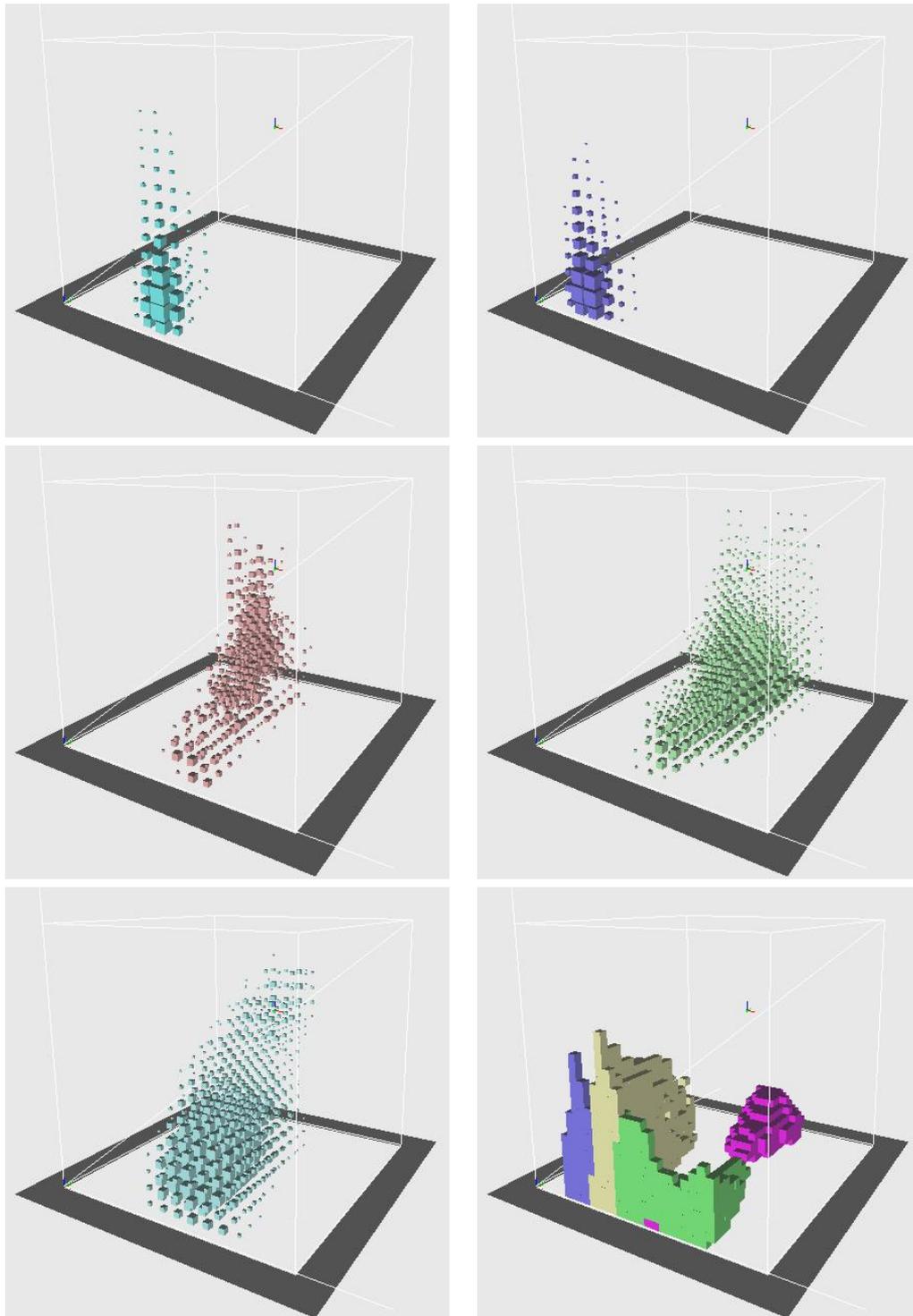


Figure 5.17: Distribution of pixel values for some classes of the Toulouse site. The x axis (to the right of the page) is the $ndvi$ channel, the y axis (perpendicular to the page) is the $pix-ent-c$ channel, and the z axis (to the top of the page) is the $pix-ent-lg-h$ channel. From left to right and top to bottom: *river*, *lake*; *sand*, *orchard*; *vineyard*, and a graph showing the most frequent terrain class for each pixel value.

in the latter case, which of the submodels are used (nested version of a flat model, random translation, random translation and scaling).

These combinations, including the name by which they will be referenced in the remainder of this chapter, are listed in table 5.2. Note that not all possible combinations are listed, and not all listed combinations will be studied in section 5.9. In this table, the *type* column indicates whether this is a flat (section 5.2) or a nested model (section 5.3). In the case of nested models, the column *nested submodels* indicates which of section 5.5.2 are used: *flat* means the embedding of flat models into the nested framework, as in section 5.5.3; *mean* means the *random translation of random variables*; and *mean and scale* means the *random scaling and translation of random variables*. Finally, the *flat submodels* indicates which of the models of section 5.5.1 are used, either for the main variable for flat models, or as the subsidiary variables for the nested models; *nonparm.* indicates the *raw non-parametric model*, *KDE* indicates the *kernel density estimation*, and *parm.* indicates the parametric models, that is, Gaussian, Laplacian, and rectangular uniform.

name	type	nested submodels	flat submodels
02	flat	—	nonparm.
08	flat	—	KDE
09	flat	—	KDE, parm.
32	nested	flat, mean	nonparm.
38	nested	flat, mean	KDE
39	nested	flat, mean	KDE, parm.
72	nested	flat, mean, mean and scale	nonparm.
78	nested	flat, mean, mean and scale	KDE
79	nested	flat, mean, mean and scale	KDE, parm.

Table 5.2: Combinations of submodels used for model estimation.

Computation time required for estimating probability models varies significantly, from less than a minute to about 7 hours. Of all intervening factors, the most influential are the number of input channels used for the estimation, and whether flat or nested models are being estimated. Figure 5.18 shows histograms for the required computation time, for, on one hand, models with three input channels (in column *channels* in table 5.1) with nested estimation (column *type* in table 5.2), and, on the other hand, all other estimation types. We can see that the estimation of more complex models (with more channels and nested estimation) takes more time than that of simpler models.

An important question that needs to be addressed when, as we have done, a new probability model is proposed, is whether it actually improves upon existing models. This can be answered in two ways. First, we can test whether, for typical data relevant to the application, the new model fits the data better than existing models do. Second, we can run a classification algorithm using the new model, and compare its performance to that of a classification using the existing models. We shall now give the first answer; in section 5.9 we give an answer to the second part.

Using the configurations of table 5.1, and the configurations in table 5.2 where we let the estimator choose between a flat and a both kinds of nested estimation—that is, the lines starting with “7. . .”—we have estimated 379 different class models. Remember, from section 5.5.4, that for each class we estimate the optimal flat model, the optimal nested model with translation (section 5.5.2) and the optimal nested model with translation and scaling (section 5.5.2), and then the best of these three, the one with lowest value of the Bayes Information Criterion, is selected.

Of these 379 configurations, in 27 cases the flat model was the optimal one. In 236 cases the

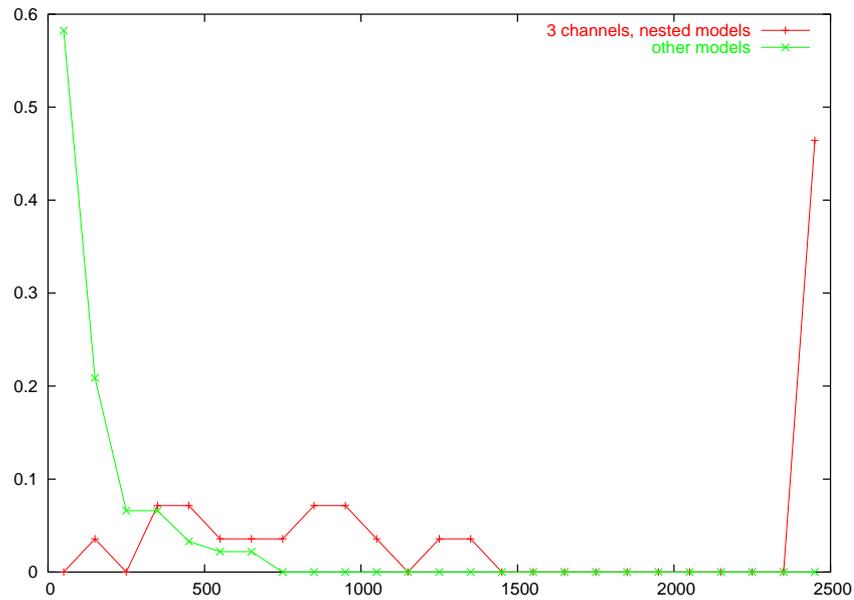


Figure 5.18: Histogram of computation time (in seconds) for estimating probability models, drawn separately for nested models with three channels, and for all other models. The right-most bin contains also all samples larger than 2500s.

best one was the nested model with translation, and in 24 cases it was the nested model with translation and scaling. In 92 cases nested models could not be estimated because too few training regions were available, which makes the estimation of the per-region random variable V impossible, so a flat model was selected (we require at least 6 regions for nested estimation). This is summarized in table 5.3.

selected model	number of cases
flat model	27 (7%)
nested with translation	236 (62%)
nested with translation and scaling	24 (7%)
nested estimation not possible	92 (24%)
total	379

Table 5.3: Optimal model selected in the estimation of probability models for 379 terrain classes for different configurations. The line *nested estimation not possible* refers to cases where too few training regions were available in order to reliably estimate a nested model.

For illustration, we give in table 5.4 the actual values of the BIC for all terrain classes with the “tor3” test set (Toulouse with 3 input channels and small radiometry domain) and the “79” submodel (maximum freedom to choose among different models).

It is clear from these values of the BIC and from table 5.3 that nested models describe this kind of data—radiometry and textural features of high-resolution aerial imagery of rural areas—better than the standard flat models, in most cases. Among the two implemented nested models—with translation only, and with translation and scaling—the nested model with translation only also wins overwhelmingly, suggesting that for this application there is

terrain class	Bayes Information Criterion			best model
	flat	translation	tran. & scaling	
Field	$2.29978 \cdot 10^6$	$1.56147 \cdot 10^6$	$2.22426 \cdot 10^6$	translated
Forest	$2.24723 \cdot 10^6$	$1.86998 \cdot 10^6$	$1.91402 \cdot 10^6$	translated
Orchard	$3.97104 \cdot 10^5$	$3.59704 \cdot 10^5$	$2.09092 \cdot 10^5$	translated and scaled
Lake	$6.5032 \cdot 10^5$	$5.49936 \cdot 10^5$	$8.97664 \cdot 10^5$	translated
River	$2.03592 \cdot 10^5$	$2.39132 \cdot 10^5$	$2.17792 \cdot 10^5$	flat
Quarry	$8.9538 \cdot 10^4$	n/c	n/c	flat
Tillable	$4.78938 \cdot 10^6$	$3.98573 \cdot 10^6$	$7.3943 \cdot 10^6$	translated
Sand	$1.79515 \cdot 10^4$	n/c	n/c	flat
Vineyard	$3.71944 \cdot 10^5$	$3.01984 \cdot 10^5$	$2.54888 \cdot 10^5$	translated and scaled

Table 5.4: Values of the Bayes Information Criterion for all terrain classes of the “tor3 - 79” configuration, separately for the flat estimation, nested estimation with translation, and nested estimation with translation and scaling. “N/c” indicates that a nested model could not be estimated because not enough training regions were available.

no need to develop more complex nested models. Finally, the fact that still in some cases the flat model was selected even when nested estimation was possible reassures us about the correctness of implementation.

That the BIC indicates that nested models are often better than flat models at describing training data is encouraging, but is no guarantee that nested models will actually perform better in terms of classification accuracy. We shall examine whether they do or not in section 5.9.

These nested and flat estimations also involve the estimation of underlying random models. As described in section 5.5.1, given a data distribution, the estimation procedure will estimate the best-fitting Gaussian variable, the best-fitting Laplacian, Rayleigh, rectangular uniform, and will also try a Kernel Density Estimation (KDE). The one model among these that actually best fits the data will be selected. This will be the final model in the case of flat estimation, or will be used as one of the component random variables for nested estimation. It may be interesting to see what the actual best-fitting variable types are.

The estimation of the 379 class models described above required the estimation of 1880 underlying “flat” models. The types of the best-fitting random models are given in table 5.5.

It is interesting to note that, contrary to popular assumptions, in many cases the data does not follow a Gaussian model, although not all of these estimated variables correspond to pixel values. Also, see again figures 5.14 to 5.17, where many non-Gaussian pixel value distributions can be found.

selected model	number of cases
Gaussian	158 (8%)
Laplacian	303 (16%)
Rayleigh	504 (27%)
rect. uniform	239 (13%)
KDE	676 (36%)
total	1880

Table 5.5: Optimal model selected —among Gaussian, Laplacian, Rayleigh, rectangular uniform, and KDE— in the estimation of the 1880 underlying flat probability models involved in the estimation of the 379 class models for different configurations.

5.8 Classification into forest and non-forest

In [TSB05] we presented some intermediate results that we obtained during the development of the per-region flat classification algorithms, applied with very good results to the classification regions in a rural area into forest and non-forest. Although these tests cover only a part of this chapter, because of their practical interest—forest discrimination is an important task in automatic map making—the results will be recalled in this section.

Classification involves three steps. First, a partition of the image to be classified is obtained. The cadastre, which partitions terrain into cadastre regions, can be used. The edges of cadastre regions, however, are geometrically imprecise and do not correspond to actual edges in a land cover class map because, among others, farmers are free not to follow cadastre edges when growing their crops. The cadastre can however be registered to the image to obtain a better image partition (see [TS04a] or [TSPD04]). Alternately—for example, when cadastre data is not available—a partition can be obtained by simply running a segmentation algorithm configured to give a “fine” segmentation, one with small regions; there is a risk of obtaining biased results in the classification confidence measures if the segmentation uses the same image channels as the classification, which is why using the cadastre is a better option. Note that some cadastre regions are not homogeneous, but contain more than one land cover class. In that case, confidence will be very low for whatever single class is given for the region; a post-processing step should attempt to divide such regions into smaller homogeneous regions.

Next, probability models are estimated for each terrain class. This involves manually defining, in a training image, a set of polygons for each terrain class, constructing a histogram for the radiometries—or other input features such as texture or derived colour channels—of pixels in a class, selecting a probability model for each class, and estimating the parameters for these models, using the radiometry histograms. Generalization occurs at this point, by preferring a worse-fitting but simpler model over a more complex, better-fitting one. Models were estimated using the techniques described in section 5.5.1 (recall that nested models, described in section 5.5.2, were not used for this experiment).

Finally, each region in the image to be classified is processed in turn. The histogram of the region is constructed, and compared to the probability models for each terrain class. The most probable class, or that with the probability model closest to the region’s histogram, is selected for that region. For this we used some of the algorithms presented in section 5.2.2, as well as, in order to define a baseline, the per-pixel algorithms of section 5.2.1. In particular, the combinations of classifiers and confidence measures listed in table 5.6 were used (the \hat{c} and \hat{c} are those defined in sections 5.2 and 5.4):

type	name	$\hat{c}(R)$	$\hat{c}(R)$
Pixel	MAP	$\operatorname{argmax}_{\omega} K_{P:MAP}(\omega, R)$	$\log \max_{\omega} K_{P:MAP}(\omega, R)$
Pixel	ML	$\operatorname{argmax}_{\omega} K_{P:ML}(\omega, R)$	$\log \max_{\omega} K_{P:ML}(\omega, R)$
Majority	MAP	$\hat{c}_{Maj}(R)$	$\hat{c}_{Maj}(R)$
Flat	MAPn	$\operatorname{argmax}_{\omega} K_{S:MAP}(\omega, R)$	$\log \max_{\omega} K_{S:MAPn}(\omega, R)$
Flat	MLn	$\operatorname{argmax}_{\omega} K_{S:ML}(\omega, R)$	$\log \max_{\omega} K_{S:MLn}(\omega, R)$
Flat	KL	$\operatorname{argmax}_{\omega} (-d_{KL}(I(R), V_{\omega}))$	$\max_{\omega} (-d_{KL}(I(R), V_{\omega}))$
Flat	Chi2	$\operatorname{argmax}_{\omega} (-\log \chi^2(I(R), V_{\omega}))$	$\max_{\omega} (-\log \chi^2(I(R), V_{\omega}))$
Flat	Chi2n	$\operatorname{argmax}_{\omega} (-\log \chi^2(I(R), V_{\omega}))$	$\max_{\omega} (-\log(\chi^2(I(R), V_{\omega})/ R))$

Table 5.6: Classification methods and confidence measures tested for the forest/non-forest classification.

Using the estimation and classification methods listed in the previous section, we present results for two sites. In the first, *Saint-Léger*, we use aerial images at 80cm resolution, resampled to 50cm, with red, green, and blue channels. In the second, *Toulouse*, we use digital aerial images at 20cm resolution, also resampled to 50cm, with red, green, blue, and near-infrared channels. Training and evaluation polygons are defined on both sites. Classification is not only performed on the raw radiometric channels but also on derived colour and textural channels—the latter because few channels are available but spatial resolution is very high. Following suggestions in [Gif04], for Saint-Léger we use the raw green channel, the second Karhunen-Loève transformed channel (from [VdWSLVD99], $k_2 = (red - blue)/2$), and the local proportion of pixels whose gradient module is over a threshold [Bai97]. For Toulouse we use the green channel, the same gradient-based measure, and the normalized difference vegetation index (NDVI).

Using the training polygons, probability models are obtained, for each site, for terrain classes “field”, “forest”, “road”, “bare soil”, “orchard”, “quarry”, “untilled field”, “other vegetation”, “water”, “sand”, “vineyard”, and “shrubs”. For each class, several models are computed as described in section 5.5.1 and the best fit—according to the Bayes Information Criterion—is selected. Regions are obtained either from a fine segmentation of the images [GLMC03b] or by registering geometrically imprecise cadastral maps onto the images [TSPD04, TS04a]. Per-region classification is then performed using the methods described before. Finally, all classes except the “forest” class are merged into a single “non-forest” class. This is done after classification, and not at the modelling stage, because non-forest classes have quite distinct signatures which would be more difficult to model jointly.

Evaluation results are summarized in table 5.7. This shows the ratio of correctly classified pixels at 100% coverage,—accepting all classifications regardless of their confidence value— for the Saint-Léger and Toulouse sites using the presented classification algorithms, and using either an image oversegmentation or registered cadastral data as input regions to the per-region classifiers. The Saint-Léger test site has an area of 5.7 km². The Toulouse test site has an area of 37.6 km². However, since only 7.0 km² of cadastral data is available for the Toulouse site, results for that site using registered cadastral edges as regions are less significant; results under “Toulouse cadastral” in table 5.7 correspond to the subset of the Toulouse site where cadastral data is available. Using regions extracted from cadastral maps instead of an oversegmentation of the source image gives slightly worse accuracy, but output regions tend to have higher geometrical quality. Execution time is approximately 40 s/km² on a 2.4GHz Intel Pentium 4 processor for a single algorithm.

Method		Saint-Léger		Toulouse	
		segmentation	cadastre	segmentation	cadastre
Flat	KL	99.4%	97.0%	93.5%	85.7%
Flat	Chi2	99.4%	97.6%	87.6%	63.3%
Flat	Chi2n	99.4%	97.6%	87.6%	63.1%
Flat	MAPn	99.4%	97.0%	95.1%	90.1%
Flat	MLn	99.4%	97.0%	95.0%	90.1%
Pixel	MAP	98.2%		92.8%	
Pixel	ML	97.6%		90.0%	
Majority	MAP	99.4%	97.8%	93.1%	84.8%

Table 5.7: Classification accuracies—correctly classified pixels in the evaluation set—for full coverage, for both the Saint-Léger and Toulouse test sites, and both using an image oversegmentation (*segmentation*) or registered cadastral data (*cadastre*) as input regions to the per-region classifiers.

For the Saint-Léger site, all algorithms except per-pixel MAP and ML achieve better than 99% pixel-wise accuracy —accuracy is defined as the ratio of correctly classified pixels in the evaluation set over the total number of pixels in the evaluation set. For the much more diverse and complex Toulouse site, the best algorithm, per-region flat MAPn, achieves an accuracy of over 95%; this is slightly better than the per-pixel MAP and ML methods, and has the advantage of not having the dotted noise typical of per-pixel classification. It is also slightly better than a majority vote of a per-pixel MAP classification. For comparison, recent papers—which often use per-pixel classification on images of lower spatial resolution but higher spectral resolution—report accuracies around 85%–90% [THC03, HEBF04, CM03].

In some cases, accuracy increases with decreasing coverage—as a larger fraction of the least-confidently-classified regions is rejected, accuracy in the accepted regions increases. This indicates meaningful confidence values. In that case we can obtain higher accuracies at the cost of decreased coverage. In other cases, however, the calculated confidence measure appears not to be indicative of the quality of classification. In figure 5.19 we show several confidence measure graphs, some showing the desired behaviour (increasing accuracy as coverage decreases) and some not. For example, in the Toulouse site, using regions obtained from an over-segmentation, and the per-region flat MAPn algorithm, the accuracy of the “forest” class can be increased from 87.9% at 100% coverage to 96.0% at 81% coverage (figure 5.19 E).

Figures 5.20 and 5.21 show, for the Saint-Léger site and a portion of the Toulouse site respectively, the source image and the class map using segmentation-derived regions and the per-region flat Chi2n and MAPn algorithms. For Saint-Léger, we also show the classification accuracy as a function of coverage, and the confidence map; accuracy indeed increases with decreasing coverage, indicating that the confidence measure is meaningful there. For the Toulouse site, global confidence measures—shown in figure 5.19 C—are not meaningful and cannot be used to obtain higher accuracy at lower coverage; confidence classes for certain classes, however, behave correctly—such as that for the “forest” class shown in figure 5.19 E.

In this section we have presented results on applying some per-region classification methods using flat models to the problem of locating forest areas in high-resolution colour and color-infrared aerial images with very high reliability, achieving more than 95% accuracy. Because few channels are available, but spatial resolution is high, we use texture features in addition to raw radiometry and derived colour channels as classification input. Region boundaries are obtained from a multi-scale hierarchical segmentation or from a registration of cadastral maps—the latter gives lower classification accuracies but better spatial precision. When using segmentation regions, the presented per-region flat MAPn method outperforms the per-pixel baseline methods also evaluated. We obtain slightly more accurate classifications than other studies, while, thanks to the use of textural features, being able to use higher spatial resolution images—and therefore having much improved spatial precision.

In addition, we have tested some classification confidence measures that should indicate those classifications that the classifier is unsure about; this would allow a trade-off between classification accuracy and coverage—leaving some areas unclassified, to be processed by a human operator, in exchange for higher accuracy in the classified regions. Evaluation shows that these confidence measures behave as expected for some cases but not all: further research is needed on the subject.

5.9 Evaluation

In this section, I present the results of several experiments that were run in order to evaluate the performances of the proposed classification algorithms, under a variety of conditions.

I will start by presenting the factors that can be controlled for such experiments, and then

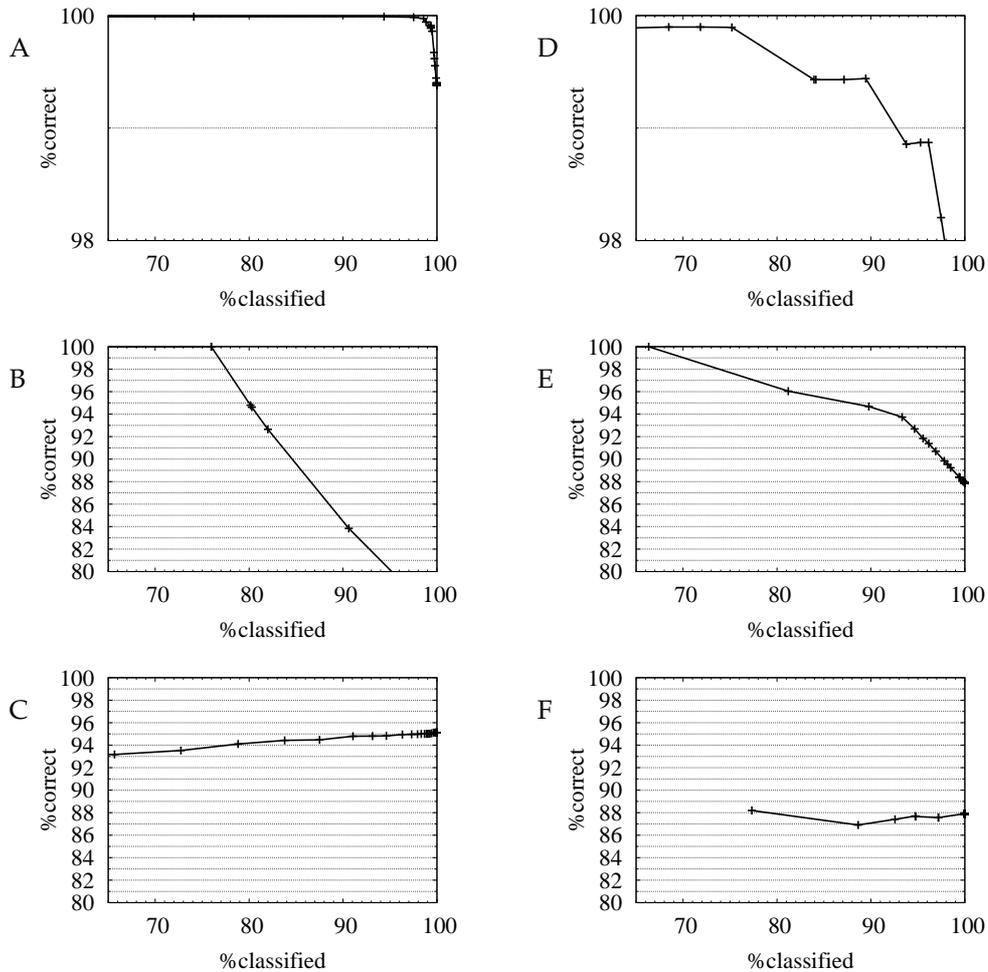


Figure 5.19: Several graphs of classification accuracy as a function of coverage showing excellent (A, D), correct (B, E), and undesired behaviours (C, F). A: Saint-Léger, segmentation regions, per-region flat MAPn, "forest" class. B: Toulouse, cadastre regions, per-region flat MAPn, "forest" class. C: Toulouse, segmentation, per-region flat MAPn, global accuracy. D: Saint-Léger, cadastre, per-region flat KL, global accuracy. E: Toulouse, segmentation, per-region flat MAPn, "forest" class. F: Toulouse, segmentation, per region flat MAP ($\hat{c}(R) = \operatorname{argmax}_{\omega} K_{S:\text{MAP}}(\omega, R)$, $\hat{\kappa}(R) = \log \max_{\omega} K_{S:\text{MAP}}(\omega, R)$), "forest" class.

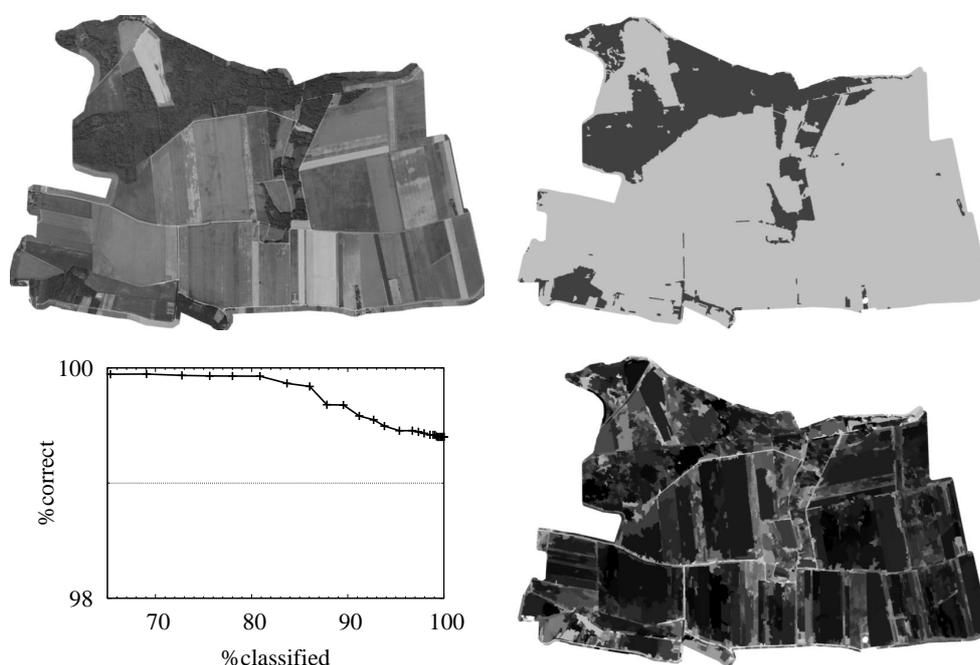


Figure 5.20: Saint-Léger site, classified using the per-region flat Chi2n method and segmentation regions. Top left: input image. Top right: class map (white: unclassified; dark grey: forest; light grey: non-forest). Bottom right: confidence values (darker is higher confidence). Bottom left: classification accuracy (% correctly classified pixels) as a function of coverage (% classified pixels, depending on a threshold on the classification confidence value).



Figure 5.21: A portion of the Toulouse site, classified using the per-region flat MAPn method and segmentation regions. Left: input image. Right: class map (white: unclassified; dark grey: forest; light grey: non-forest). Graphs of accuracy versus coverage are given in figure 5.19 C and E.

analyse separately the effects of each factor —choice of input channels; source of the partition of the image into regions, for per-region classification; use of not of Yuille’s confidence measure; and choice of nested estimation method—. Then the performances of different classification algorithms will be shown, and finally the results using the best-performing algorithm will be studied in detail. The section closes with an analysis of required classification time.

5.9.1 Test parameters

There are several factors which we can control for such experiments.

First, there is the test site which is to be classified. I presented the *Saint-Léger* and *Toulouse* test sites in section 5.7. Then, there is the set of input channels to be used for the classification, and the data domain \mathbf{D} . In addition, there is the specific probability model —obtained from training data— to be used. These three factors also concern the estimation of probability models, as described in section 5.7. Although it would be possible, when classifying images from one site, to use a model estimated using training data from another site, I have not tried to do it, because I consider these two sites to be too much different for this cross-site experiment to have any value. Indeed, the expected way that system should be used is that new models would be estimated for each significantly different landscape type, so as to capture the specific characteristics of that zone, as well as characteristics dependent on the specific time of the year and the specific sensor used for acquiring the images. Production sites are so much larger than the available test sites that it makes sense even to manually define a small ground truth for each site and use it to classify the rest of the site. Nonetheless, in this thesis, the regions used for training and the regions used for evaluating classifications, although taken from the same site, do not overlap, of course.

Another factor is which partition of the image into regions is to be used, for the region-based classification algorithms. There are three options: We can use a segmentation of the image into small regions, as given by the transition, in a two-phase hierarchical segmentation, between the first and second phases (section 3.2.2). This has the advantage that we can do this for all kinds of images, even if cadastre data is not available, and the disadvantage that, in order to be sure that these regions are homogeneous, they have to be small. With small regions, the advantage of per-region classification compared to per-pixel classification is smaller. The second option is to use the registered cadastre, which gives large homogeneous regions; however, cadastre data is not always available. In particular, for the Toulouse data set, cadastre data is available for only 7.0 km^2 of a total of 37.6 km^2 . Results will be less significant, and furthermore they will not be comparable with results using a fine segmentation as region input because they do not cover the same area. The third option is to use a fine segmentation as region input, but only for the area for which cadastre data is also available; this allows a more meaningful comparison between both region sources, although the fact that such a small area is covered, in the Toulouse case, would make the results less interesting. In this section I give results for the Saint-Léger site with either a registered cadastre or a fine segmentation covering the same area, and for the Toulouse site with a fine segmentation covering the whole image.

The final factor affecting a classification is which classification algorithm to use —including which confidence measure is chosen. The chosen probability model determines the possible choices, in that only with nested models can nested classification algorithms be used, and vice versa. In table 5.8 we list the algorithms which we have evaluated —including the name by which they shall be referenced in the remainder of this chapter. For the definition of \mathbf{Y} , recall equation 5.77.

Note that not all combinations of algorithms, region sources, test sites, and estimation models, have been tested.

type	name	$\hat{c}(R)$	$\hat{\kappa}(R)$
Pixel	MAP	$\operatorname{argmax}_{\omega} K_{P:MAP}$	$\log \max_{\omega} K_{P:MAP}$
Pixel	MAPd	$\operatorname{argmax}_{\omega} K_{P:MAP}$	$Y(K_{P:MAP})$
Pixel	ML	$\operatorname{argmax}_{\omega} K_{P:ML}$	$\log \max_{\omega} K_{P:ML}$
Pixel	MLd	$\operatorname{argmax}_{\omega} K_{P:ML}$	$Y(K_{P:ML})$
Majority	MAP	$\hat{c}_{Maj}(R)$	$\hat{\kappa}_{Maj}(R)$
Majority	MAPd	$\hat{c}_{Maj}(R)$	$Y(\sum_{s \in R} K_{P:MAP}(\cdot, s))$
Flat	MAP	$\operatorname{argmax}_{\omega} K_{S:MAP}$	$\log \max_{\omega} K_{S:MAP}$
Flat	MAPn	$\operatorname{argmax}_{\omega} K_{S:MAP}$	$\log \max_{\omega} K_{S:MAPn}$
Flat	MAPnn	$\operatorname{argmax}_{\omega} K_{S:MAPn}$	$\log \max_{\omega} K_{S:MAPn}$
Flat	MAPo	$\operatorname{argmax}_{\omega} K_{S:MAPo}$	$\log \max_{\omega} K_{S:MAPo}$
Flat	ML	$\operatorname{argmax}_{\omega} K_{S:ML}$	$\log \max_{\omega} K_{S:ML}$
Flat	MLn	$\operatorname{argmax}_{\omega} K_{S:ML}$	$\log \max_{\omega} K_{S:MLn}$
Flat	KL	$\operatorname{argmax}_{\omega} (-d_{KL}(I(R), V_{\omega}))$	$\max_{\omega} (-d_{KL}(I(R), V_{\omega}))$
Flat	KLs	$\operatorname{argmax}_{\omega} (-d_{KLs}(I(R), V_{\omega}))$	$\max_{\omega} (-d_{KLs}(I(R), V_{\omega}))$
Flat	Chi2	$\operatorname{argmax}_{\omega} (-\log \chi^2(I(R), V_{\omega}))$	$\max_{\omega} (-\log \chi^2(I(R), V_{\omega}))$
Flat	Chi2n	$\operatorname{argmax}_{\omega} (-\log \chi^2(I(R), V_{\omega}))$	$\max_{\omega} (-\log(\chi^2(I(R), V_{\omega})/ R))$
Flat	MAPd	$\operatorname{argmax}_{\omega} K_{S:MAP}$	$Y(K_{S:MAP})$
Flat	MAPnnd	$\operatorname{argmax}_{\omega} K_{S:MAPn}$	$Y(K_{S:MAPn})$
Flat	MAPod	$\operatorname{argmax}_{\omega} K_{S:MAPo}$	$Y(K_{S:MAPo})$
Flat	MLd	$\operatorname{argmax}_{\omega} K_{S:ML}$	$Y(K_{S:ML})$
Flat	MLnnd	$\operatorname{argmax}_{\omega} K_{S:MLn}$	$Y(K_{S:MLn})$
Flat	KLd	$\operatorname{argmax}_{\omega} (-d_{KL}(I(R), V_{\omega}))$	$Y(-d_{KL}(I(R), V_{\omega}))$
Flat	KLsd	$\operatorname{argmax}_{\omega} (-d_{KLs}(I(R), V_{\omega}))$	$Y(-d_{KLs}(I(R), V_{\omega}))$
Flat	Chi2d	$\operatorname{argmax}_{\omega} (-\log \chi^2(I(R), V_{\omega}))$	$Y(1/\chi^2(I(R), V_{\omega}))$
Flat	Chi2nnd	$\operatorname{argmax}_{\omega} (-\log \chi^2(I(R), V_{\omega}))$	$Y(R /\chi^2(I(R), V_{\omega}))$
Nested	MAP	$\operatorname{argmax}_{\omega} K_{C:MAP}$	$\log \max_{\omega} K_{C:MAP}$
Nested	MAPn	$\operatorname{argmax}_{\omega} K_{C:MAP}$	$\log \max_{\omega} K_{C:MAPn}$
Nested	MAPnn	$\operatorname{argmax}_{\omega} K_{C:MAPn}$	$\log \max_{\omega} K_{C:MAPn}$
Nested	MAPo	$\operatorname{argmax}_{\omega} K_{C:MAPo}$	$\log \max_{\omega} K_{C:MAPo}$
Nested	ML	$\operatorname{argmax}_{\omega} K_{C:ML}$	$\log \max_{\omega} K_{C:ML}$
Nested	MLn	$\operatorname{argmax}_{\omega} K_{C:ML}$	$\log \max_{\omega} K_{C:MLn}$
Nested	MLnn	$\operatorname{argmax}_{\omega} K_{C:MLn}$	$\log \max_{\omega} K_{C:MLn}$
Nested	KL	$\operatorname{argmax}_{\omega} K_{C:KL}$	$\max_{\omega} K_{C:KL}$
Nested	KLs	$\operatorname{argmax}_{\omega} K_{C:KLs}$	$\max_{\omega} K_{C:KLs}$
Nested	KLv	$\operatorname{argmax}_{\omega} K_{C:KLv}$	$\max_{\omega} K_{C:KLv}$
Nested	KLsv	$\operatorname{argmax}_{\omega} K_{C:KLsv}$	$\max_{\omega} K_{C:KLsv}$
Nested	Chi2	$\operatorname{argmax}_{\omega} K_{C:\chi^2}$	$\max_{\omega} K_{C:\chi^2}$
Nested	Chi2n	$\operatorname{argmax}_{\omega} K_{C:\chi^2}$	$\max_{\omega} K_{C:\chi^2n}$
Nested	Chi2v	$\operatorname{argmax}_{\omega} K_{C:\chi^2v}$	$\max_{\omega} K_{C:\chi^2v}$
Nested	Chi2nv	$\operatorname{argmax}_{\omega} K_{C:\chi^2v}$	$\max_{\omega} K_{C:\chi^2nv}$
Nested	qML	$\operatorname{argmax}_{\omega} K_{C:qML}$	$\max_{\omega} K_{C:qML}$
Nested	qMLn	$\operatorname{argmax}_{\omega} K_{C:qML}$	$\max_{\omega} K_{C:qMLn}$
Nested	qMLv	$\operatorname{argmax}_{\omega} K_{C:qMLv}$	$\max_{\omega} K_{C:qMLv}$
Nested	qMLnv	$\operatorname{argmax}_{\omega} K_{C:qMLv}$	$\max_{\omega} K_{C:qMLnv}$
Nested	qMLnnv	$\operatorname{argmax}_{\omega} K_{C:qMLnv}$	$\max_{\omega} K_{C:qMLnv}$

Table 5.8: List of classification functions (\hat{c}) and confidence measures ($\hat{\kappa}$) tested in the evaluation. The \hat{c} and $\hat{\kappa}$ are those defined in sections 5.2 and 5.4. For clarity, we have used \hat{v}_{ω} as a shorthand for $v_{\omega}(\hat{q}_{\omega}(R))$, we have used \hat{v} as a shorthand for $v(\hat{q}(R))$, and the K terms should be taken as having an implicit (ω, R) argument—or an (\cdot, R) argument when the K itself is an argument to the Y functional.

type	name	$\hat{c}(R)$	$\hat{\kappa}(R)$
Nested	eqMAP	$\operatorname{argmax}_{\omega \in \Omega} K_{C:\text{eMAP}}$	$\log \max_{\omega \in \Omega} K_{C:\text{eMAP}}$
Nested	eqMAPn	$\operatorname{argmax}_{\omega \in \Omega} K_{C:\text{eMAP}}$	$\log \max_{\omega \in \Omega} K_{C:\text{eMAPn}}$
Nested	eqMAPnn	$\operatorname{argmax}_{\omega \in \Omega} K_{C:\text{eMAPn}}$	$\log \max_{\omega \in \Omega} K_{C:\text{eMAPn}}$
Nested	eqML	$\operatorname{argmax}_{\omega \in \Omega} K_{C:\text{eML}}$	$\log \max_{\omega \in \Omega} K_{C:\text{eML}}$
Nested	eqMLn	$\operatorname{argmax}_{\omega \in \Omega} K_{C:\text{eML}}$	$\log \max_{\omega \in \Omega} (K_{C:\text{eML}})^{1/ R }$
Nested	eqMLnn	$\operatorname{argmax}_{\omega \in \Omega} (K_{C:\text{eML}})^{1/ R }$	$\log \max_{\omega \in \Omega} (K_{C:\text{eML}})^{1/ R }$
Nested	eqMLnnv	$\operatorname{argmax}_{\omega \in \Omega} \hat{v}_{\omega}(K_{C:\text{eML}})^{1/ R }$	$\log \max_{\omega \in \Omega} \hat{v}_{\omega}(K_{C:\text{eML}})^{1/ R }$
Nested	eqMIX	$\operatorname{argmax}_{\omega \in \Omega} p_{\omega} \hat{v}_{\omega} K_{C:\text{eML}}$	$\log \max_{\omega \in \Omega} p_{\omega} \hat{v}_{\omega} K_{C:\text{eML}}$
Nested	eqMIXn	$\operatorname{argmax}_{\omega \in \Omega} p_{\omega} \hat{v}_{\omega} K_{C:\text{eML}}$	$\log \max_{\omega \in \Omega} p_{\omega} \hat{v}_{\omega} (K_{C:\text{eML}})^{1/ R }$
Nested	eqMIXnn	$\operatorname{argmax}_{\omega \in \Omega} p_{\omega} \hat{v}_{\omega} (K_{C:\text{eML}})^{1/ R }$	$\log \max_{\omega \in \Omega} p_{\omega} \hat{v}_{\omega} (K_{C:\text{eML}})^{1/ R }$
Nested	eqKL	$\operatorname{argmax}_{\omega \in \Omega} K_{C:\text{eKL}}$	$\max_{\omega \in \Omega} K_{C:\text{eKL}}$
Nested	eqKLs	$\operatorname{argmax}_{\omega \in \Omega} K_{C:\text{eKLS}}$	$\max_{\omega \in \Omega} K_{C:\text{eKLS}}$
Nested	eqKLv	$\operatorname{argmax}_{\omega \in \Omega} \hat{v}_{\omega} K_{C:\text{eKL}}$	$\max_{\omega \in \Omega} \hat{v}_{\omega} K_{C:\text{eKL}}$
Nested	eqKLvo	$\operatorname{argmax}_{\omega \in \Omega} \tilde{v}_{\omega}(R) K_{C:\text{eKL}}$	$\max_{\omega \in \Omega} \tilde{v}_{\omega}(R) K_{C:\text{eKL}}$
Nested	eqKLsv	$\operatorname{argmax}_{\omega \in \Omega} \hat{v}_{\omega} K_{C:\text{eKLS}}$	$\max_{\omega \in \Omega} \hat{v}_{\omega} K_{C:\text{eKLS}}$
Nested	eqKLsvo	$\operatorname{argmax}_{\omega \in \Omega} \tilde{v}_{\omega}(R) K_{C:\text{eKLS}}$	$\max_{\omega \in \Omega} \tilde{v}_{\omega}(R) K_{C:\text{eKLS}}$
Nested	eqChi2	$\operatorname{argmax}_{\omega \in \Omega} K_{C:\text{e}\chi^2}$	$\max_{\omega \in \Omega} K_{C:\text{e}\chi^2}$
Nested	eqChi2n	$\operatorname{argmax}_{\omega \in \Omega} K_{C:\text{e}\chi^2}$	$\max_{\omega \in \Omega} K_{C:\text{e}\chi^2\text{n}}$
Nested	eqChi2nn	$\operatorname{argmax}_{\omega \in \Omega} K_{C:\text{e}\chi^2\text{n}}$	$\max_{\omega \in \Omega} K_{C:\text{e}\chi^2\text{n}}$
Nested	eqChi2v	$\operatorname{argmax}_{\omega \in \Omega} K_{C:\text{e}\chi^2\text{v}}$	$\max_{\omega \in \Omega} K_{C:\text{e}\chi^2\text{v}}$
Nested	eqChi2nv	$\operatorname{argmax}_{\omega \in \Omega} K_{C:\text{e}\chi^2\text{v}}$	$\max_{\omega \in \Omega} K_{C:\text{e}\chi^2\text{nv}}$
Nested	eqChi2nnv	$\operatorname{argmax}_{\omega \in \Omega} K_{C:\text{e}\chi^2\text{nv}}$	$\max_{\omega \in \Omega} K_{C:\text{e}\chi^2\text{nv}}$
Nested	eqChi2vo	$\operatorname{argmax}_{\omega \in \Omega} K_{C:\text{e}\chi^2\text{vo}}$	$\max_{\omega \in \Omega} K_{C:\text{e}\chi^2\text{vo}}$
Nested	eqChi2nvo	$\operatorname{argmax}_{\omega \in \Omega} K_{C:\text{e}\chi^2\text{vo}}$	$\max_{\omega \in \Omega} K_{C:\text{e}\chi^2\text{nvo}}$
Nested	eqChi2nnvo	$\operatorname{argmax}_{\omega \in \Omega} K_{C:\text{e}\chi^2\text{nvo}}$	$\max_{\omega \in \Omega} K_{C:\text{e}\chi^2\text{nvo}}$
Nested	MAPd	$\operatorname{argmax}_{\omega} K_{C:\text{MAP}}$	$Y(K_{C:\text{MAP}})$
Nested	MAPnd	$\operatorname{argmax}_{\omega} K_{C:\text{MAPn}}$	$Y(K_{C:\text{MAPn}})$
Nested	MAPod	$\operatorname{argmax}_{\omega} K_{C:\text{MAPo}}$	$Y(K_{C:\text{MAPo}})$
Nested	MLd	$\operatorname{argmax}_{\omega} K_{C:\text{ML}}$	$Y(K_{C:\text{ML}})$
Nested	MLnd	$\operatorname{argmax}_{\omega} K_{C:\text{MLn}}$	$Y(K_{C:\text{MLn}})$
Nested	KLd	$\operatorname{argmax}_{\omega} K_{C:\text{KL}}$	$Y(K_{C:\text{KL}})$
Nested	KLsd	$\operatorname{argmax}_{\omega} K_{C:\text{KLS}}$	$Y(K_{C:\text{KLS}})$
Nested	KLvd	$\operatorname{argmax}_{\omega} K_{C:\text{KLV}}$	$Y(K_{C:\text{KLV}})$
Nested	KLsvd	$\operatorname{argmax}_{\omega} K_{C:\text{KLSv}}$	$Y(K_{C:\text{KLSv}})$
Nested	Chi2d	$\operatorname{argmax}_{\omega} K_{C:\chi^2}$	$Y(K_{C:\chi^2})$
Nested	Chi2nd	$\operatorname{argmax}_{\omega} K_{C:\chi^2\text{n}}$	$Y(K_{C:\chi^2\text{n}})$
Nested	Chi2vd	$\operatorname{argmax}_{\omega} K_{C:\chi^2\text{v}}$	$Y(K_{C:\chi^2\text{v}})$
Nested	Chi2nnvd	$\operatorname{argmax}_{\omega} K_{C:\chi^2\text{nv}}$	$Y(K_{C:\chi^2\text{nv}})$
Nested	qMLd	$\operatorname{argmax}_{\omega} K_{C:\text{qML}}$	$Y(K_{C:\text{qML}})$
Nested	qMLnd	$\operatorname{argmax}_{\omega} K_{C:\text{qMLn}}$	$Y(K_{C:\text{qMLn}})$
Nested	qMLvd	$\operatorname{argmax}_{\omega} K_{C:\text{qMLv}}$	$Y(K_{C:\text{qMLv}})$
Nested	qMLnnvd	$\operatorname{argmax}_{\omega} K_{C:\text{qMLnv}}$	$Y(K_{C:\text{qMLnv}})$
Nested	eqMAPd	$\operatorname{argmax}_{\omega \in \Omega} K_{C:\text{eMAP}}$	$Y(K_{C:\text{eMAP}})$
Nested	eqMAPnd	$\operatorname{argmax}_{\omega \in \Omega} K_{C:\text{eMAPn}}$	$Y(K_{C:\text{eMAPn}})$
Nested	eqMLd	$\operatorname{argmax}_{\omega \in \Omega} K_{C:\text{eML}}$	$Y(K_{C:\text{eML}})$
Nested	eqMLnd	$\operatorname{argmax}_{\omega \in \Omega} (K_{C:\text{eML}})^{1/ R }$	$Y((K_{C:\text{eML}})^{1/ R })$
Nested	eqMLnnvd	$\operatorname{argmax}_{\omega \in \Omega} \hat{v}_{\omega}(K_{C:\text{eML}})^{1/ R }$	$Y(\hat{v}_{\omega}(K_{C:\text{eML}})^{1/ R })$
Nested	eqMIXd	$\operatorname{argmax}_{\omega \in \Omega} p_{\omega} \hat{v}_{\omega} K_{C:\text{eML}}$	$Y(p \cdot \hat{v}_{\omega} K_{C:\text{eML}})$

Table 5.8: List of classification functions (\hat{c}) and confidence measures ($\hat{\kappa}$) tested in the evaluation. The \hat{c} and $\hat{\kappa}$ are those defined in sections 5.2 and 5.4. For clarity, we have used \hat{v}_{ω} as a shorthand for $v_{\omega}(\hat{q}_{\omega}(R))$, we have used \hat{v} as a shorthand for $v(\hat{q}(R))$, and the K terms should be taken as having an implicit (ω, R) argument—or an (\cdot, R) argument when the K itself is an argument to the Y functional.

type	name	$\hat{c}(R)$	$\hat{k}(R)$
Nested	eqMIXnnd	$\operatorname{argmax}_{\omega \in \Omega} p_{\omega} \hat{v}_{\omega} (K_{C:eML})^{1/ R }$	$Y(p \cdot \hat{v} \cdot (K_{C:eML})^{1/ R })$
Nested	eqKLd	$\operatorname{argmax}_{\omega \in \Omega} K_{C:eKL}$	$Y(K_{C:eKL})$
Nested	eqKLsd	$\operatorname{argmax}_{\omega \in \Omega} K_{C:eKLS}$	$Y(K_{C:eKLS})$
Nested	eqKLvd	$\operatorname{argmax}_{\omega \in \Omega} \hat{v}_{\omega} K_{C:eKL}$	$Y(\hat{v}_{\omega} K_{C:eKL})$
Nested	eqKLvod	$\operatorname{argmax}_{\omega \in \Omega} \tilde{v}_{\omega}(R) K_{C:eKL}$	$Y(\tilde{v}_{\omega}(R) K_{C:eKL})$
Nested	eqKLsvd	$\operatorname{argmax}_{\omega \in \Omega} \hat{v}_{\omega} K_{C:eKLS}$	$Y(\hat{v}_{\omega} K_{C:eKLS})$
Nested	eqKLsvod	$\operatorname{argmax}_{\omega \in \Omega} \tilde{v}_{\omega}(R) K_{C:eKLS}$	$Y(\tilde{v}_{\omega}(R) K_{C:eKLS})$
Nested	eqChi2d	$\operatorname{argmax}_{\omega \in \Omega} K_{C:e\chi^2}$	$Y(K_{C:e\chi^2})$
Nested	eqChi2nnd	$\operatorname{argmax}_{\omega \in \Omega} K_{C:e\chi^2n}$	$Y(K_{C:e\chi^2n})$
Nested	eqChi2vd	$\operatorname{argmax}_{\omega \in \Omega} K_{C:e\chi^2v}$	$Y(K_{C:e\chi^2v})$
Nested	eqChi2nnvd	$\operatorname{argmax}_{\omega \in \Omega} K_{C:e\chi^2nv}$	$Y(K_{C:e\chi^2nv})$
Nested	eqChi2vod	$\operatorname{argmax}_{\omega \in \Omega} K_{C:e\chi^2vo}$	$Y(K_{C:e\chi^2vo})$
Nested	eqChi2nnvod	$\operatorname{argmax}_{\omega \in \Omega} K_{C:e\chi^2nvo}$	$Y(K_{C:e\chi^2nvo})$

Table 5.8: List of classification functions (\hat{c}) and confidence measures (\hat{k}) tested in the evaluation. The \hat{c} and \hat{k} are those defined in sections 5.2 and 5.4. For clarity, we have used \hat{v}_{ω} as a shorthand for $v_{\omega}(\hat{q}_{\omega}(R))$, we have used \hat{v} as a shorthand for $v(\hat{q}(R))$, and the K terms should be taken as having an implicit (ω, R) argument—or an (\cdot, R) argument when the K itself is an argument to the Y functional.

Figure 5.22 shows a flowchart of steps involved in evaluating a classification algorithm: With a probability model, image data—including transformed colour channels and texture descriptors—and a partition of the image into regions, one classification algorithm produces a *class map*, an image where each pixel’s colour indicates the selected class for the corresponding image pixel, and a confidence image, where each pixel’s value indicates the confidence that the algorithm has in the given class. This is so even for region-based classification: in that case, all pixels of a region will have the same value in the class map and the confidence image. The class map is then compared to a reference class map, to obtain the classification accuracy. The confidence image is used to examine what would happen if a fraction of the pixels classified with least confidence were dropped—to be classified by a human photo-interpreter. Different classification accuracies are calculated for different quantities of dropped pixels.

For the images in this thesis showing classification results, the colour code of figure 5.23 has been used. The hue indicates the terrain type. To show not only the terrain type but also the confidence measure given to each region, we use the hue to indicate the terrain type and the intensity indicates the confidence measure. Therefore, regions classified with high confidence will have a very intense colour, whereas regions classified with low confidence will be darker, and even black for minimal confidence.

There are several ways of reporting classification performance. The most obvious one is *classification accuracy*. This is the ratio of the number of pixels that have been correctly classified to the number of pixels for which the true class is known. Pixels not in the ground truth—pixels for which the true class is not known—are not included in either term of the ratio. If some pixels are rejected, because their classification confidence is too low, they are not included in either term of the ratio, too. This measure is what will be used in most of this chapter.

Another measure, the *kappa parameter*, or its approximation KHAT [Wil05], is less intuitive but more objective, since it takes into account the chance occurrence of correct classifications—the fact that even in a random classification some answers will be correct.

Finally, for a given class, the *producer accuracy* is the ratio of the number of pixels correctly classified as belonging to that class, relative to the total number of pixels defined in the ground

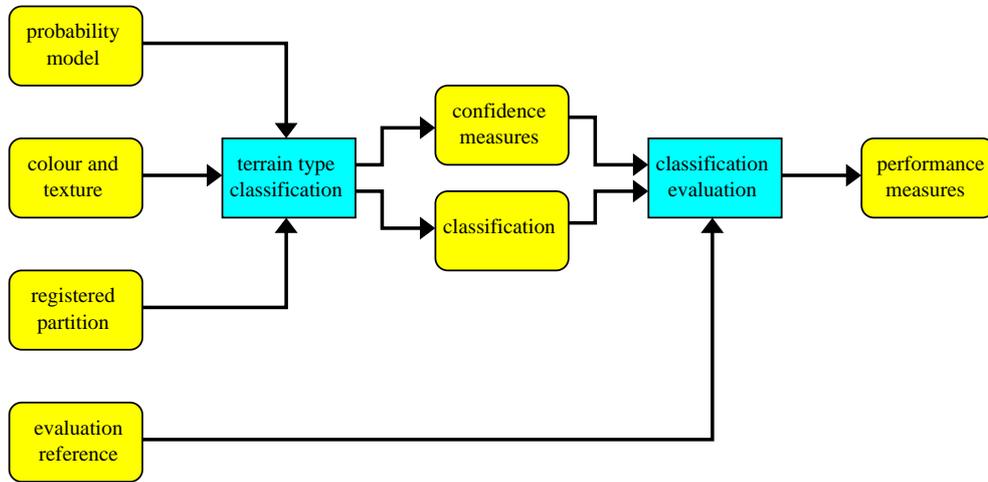


Figure 5.22: Flowchart depicting the processing steps (in blue rectangles) and intermediate results (in yellow rounded boxes) for the evaluation of a classification algorithm.

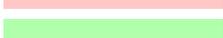
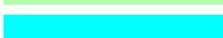
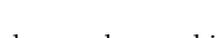
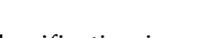
		Field
		Forest
		Lake
		Orchard
		Quarry
		River
		Sand
		Tillable
		Vineyard
		Cadastre built

Figure 5.23: Colour codes used in classification images. If only the class is shown, we use the fully-saturated colour of the first column. If both the class and the confidence are shown, we use the second column, where intense colours correspond to high confidence and darker colours correspond to low confidence.

truth as belonging to that class. The *user accuracy* is the ratio of the number of pixels correctly classified as belonging to that class, relative to the total number of pixels that the classification system has classified into that class.

5.9.2 Input channels

Since there are so many combinations of factors —test sites, probability models, classification algorithms—, we will start with some simple comparisons to investigate the effects of these factors separately.

Let's start with the choice of input channels to the classification and model estimation. From table 5.1 we have, for each test site, the choice of 2 or 3 input channels, and the choice of a standard (with values in $\{0, \dots, 255\}^d$) or a wider data domain \mathbf{D} . At first sight, it would seem that 3 channels would be better than 2, since more information is available. Also, a wider radiometry domain may give better results since we avoid some problems related to parametric models whose support significantly falls outside of the data domain. However, both using 3 channels instead of 2, and using a wider data domain instead of the standard one, may make classification —and estimation— much slower. Also, although this is unlikely to be the case here, it is well known in hyperspectral image analysis that using too many input channels, if not enough training samples are available, can yield worse results.

Figure 5.24 contains two scatterplots showing a comparison of classifications using two and three input channels. Each point represents two classifications which only differ in that one used two channels (*stl2* or *tor2*) and the other one used three channels (*stl3* or *tor3*), with all other factors equal (test site, estimation mode, classification algorithm, ...). A point's x coordinate is the classification accuracy of the corresponding test with two channels, and its y coordinate is that with three channels. Tests for the Saint-Léger and Toulouse sites are drawn in different colours, and because of the required computation time, we could not test all classification algorithms with the Toulouse site. The left scatterplot corresponds to a 100% coverage, that is, all pixels returned by the classifier are accepted. The right scatterplot corresponds to a 75% coverage, that is, the 25% of pixels that have the worst classification confidence are rejected.

At first sight, it seems that for the Toulouse site, 3-channel classification performs better, since these points are above the $x = y$ line, while for the Saint-Léger site, 2-channel classification is best, since these points are below the $x = y$ line. However, if we focus on the best-performing experiments, those near the top-right corner of the plots, we notice that also for Saint-Léger 3-channel classification is best, although by not as much as for Toulouse. Three-channel classification, however, is in general slower.

Figure 5.25 contains two scatterplots showing a comparison of classifications using standard data domains ($\mathbf{D} = \{0, \dots, 255\}^d$) and wide data domains ($\mathbf{D} = \{-64, \dots, 320\}^d$). The scatterplots were constructed as the previous ones (but in this case the number of input channels is also one of the factors held constant for both coordinates of a data point). The x coordinate is the accuracy using the standard data domain, and the y coordinate that using the wide data domain. Only data for the Toulouse site is shown: since results with standard data domains for Saint-Léger were already very satisfying, we did not run them with wider data domains. Again, the left scatterplot corresponds to a 100% coverage and the right one to 75% coverage.

It seems that, in general, tests with a wide data domain give better results than tests with the standard data domain. However, for the best-performing tests (close to the top-right corner), there does not seem to be any difference —certainly not enough to justify the extra computation time required, in general, by wide-domain classification.

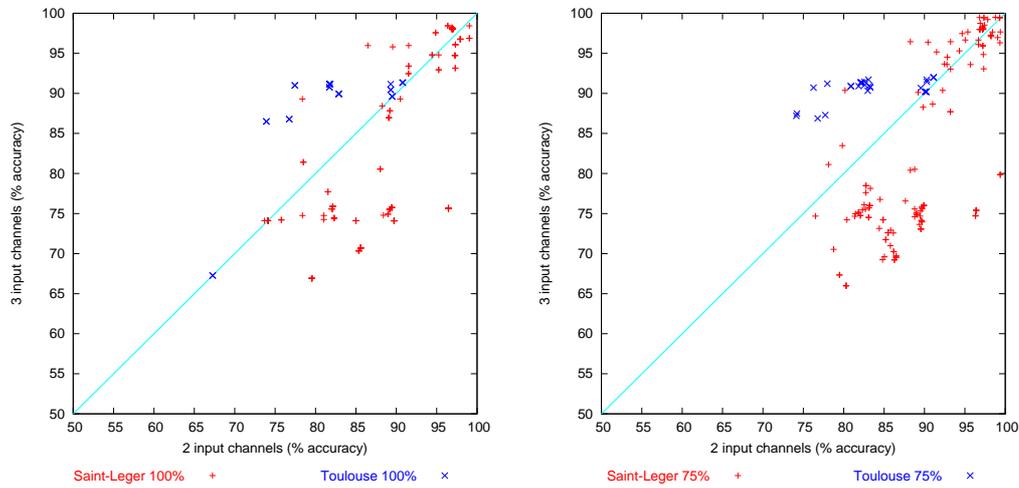


Figure 5.24: Scatterplots showing a comparison of classifications using two and three channels.

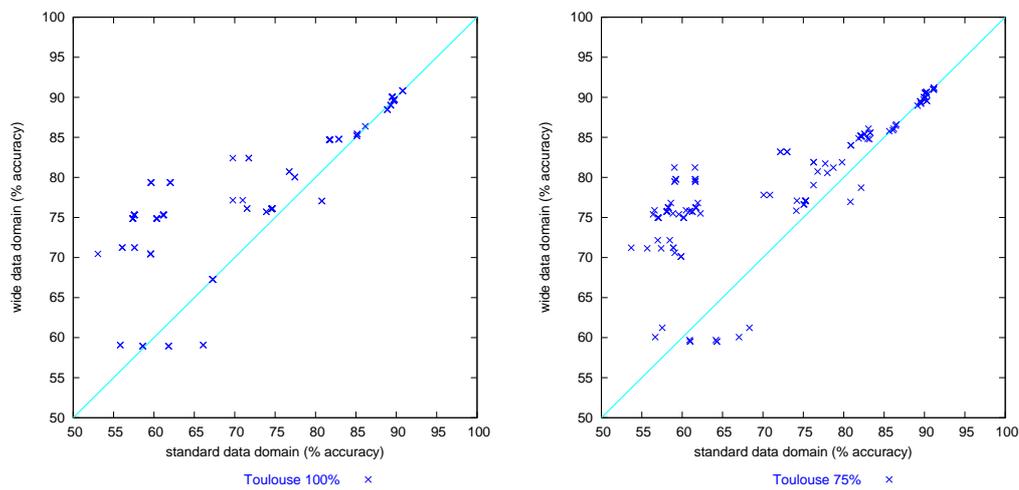


Figure 5.25: Scatterplots showing a comparison of classifications using standard and wide data domains.

5.9.3 Source of image partition

Another factor to take into account is whether the regions to be classified are those obtained by registering the cadastre, or simply those given by a fine image segmentation. If no cadastre is available, we will be forced to use the second option.

Regions to classify are larger in the first case, so in theory, this should give better results.

Figure 5.26 shows two scatterplots with a comparison of classifications using different kinds of image partitions. The scatterplots were constructed as the previous ones. The x coordinate is the accuracy using a fine segmentation of the image as regions to classify, and the y coordinate is that using regions from the registered cadastre. Since only for Saint-Léger was enough cadastre data available, tests shown in these plots are only for the Saint-Léger site. The left scatterplot shows data for a 100% coverage, and the right scatterplot shows data for several coverage levels, only for high performances.

It is not so clear which of the two region sources gives better results. However, given that when using the cadastre there are less regions to classify, and therefore the classification is faster, I would recommend using the registered cadastre when possible. Furthermore, comparing figures 5.29 and 5.32, we can see that cadastre classification is more regular, less noisy.

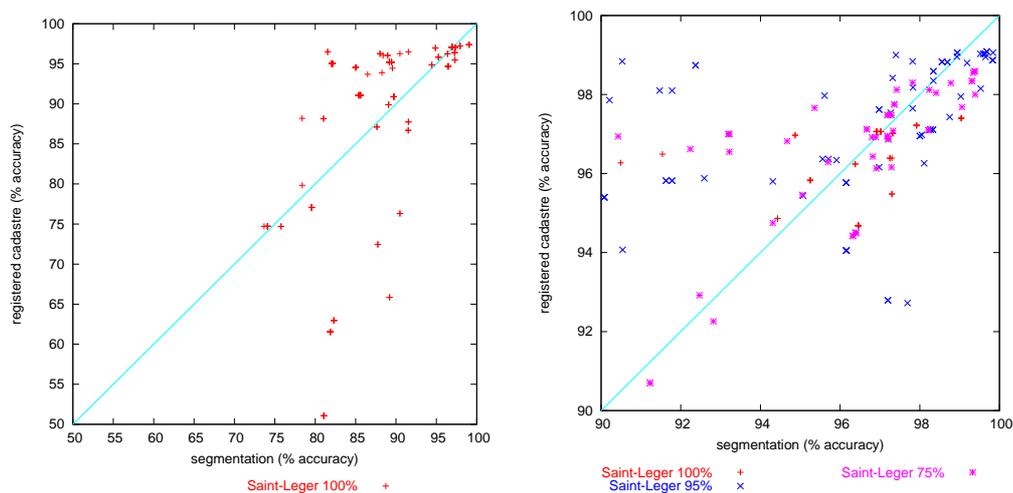


Figure 5.26: Scatterplots showing a comparison of classifications using a fine image segmentation or the registered cadastre as source of regions for the region-based classifiers.

5.9.4 Yuille's confidence measure

In section 5.4.3 we described Yuille's confidence measure as an alternative to the ones proposed in this chapter. We will now study whether it performs better or not.

Figure 5.27 shows several plots comparing tests with and without Yuille's confidence measure. The top row shows scatterplots, constructed as in previous sections, comparing the confidence measures presented in this chapter (as the x coordinate) and Yuille's confidence measure (as the y coordinate), for three coverage levels. The right plot is simply a close-up of the left one. Data points at 100% coverage lie in the diagonal, since at 100% coverage the confidence measure is not used to discard a part of the results. Results include tests for both the Saint-Léger and Toulouse sites. The middle row shows scatterplots for 95% coverage (on the left) and 75% coverage (on the right), which do not clarify the situation. In the bottom row we

show a different kind of graphs: for each pair of tests (with or without Yuille’s measure, giving classification accuracies of F_{with} and F_{without} respectively) we draw a point at coordinates

$$x = \frac{F_{\text{with}} + F_{\text{without}}}{2}, \quad y = \frac{F_{\text{with}}}{F_{\text{without}}}. \quad (5.215)$$

This gives a clearer picture: it seems that using Yuille’s measure has only a negligible effect on results, compared to using the measures proposed here. In most cases, it gives a very slight improvement, but in a few cases it causes a larger loss. For the best-performing tests the effect is tiny.

5.9.5 Choice of nested estimation method

In this subsection we study nested classifications only. We want to know whether, if using nested estimation, it is better or worse to allow the “translated and scaled” models of section 5.5.2, in addition to the “translated” models of section 5.5.2 and flat models. In the naming scheme of table 5.2, the question is whether the “39” or the “79” submodels are better.

We study this as before. Figure 5.28 contains two scatterplots showing a comparison of classifications using the “39” model (that is, allowing estimation of “flat” and “translated” models), and classifications using the “79” model (allowing estimation of “flat”, “translated”, and “translated and scaled” models), constructed as before. The x coordinate is the accuracy using the “39” model and the y coordinate is that using the “79” model. The right plot is simply a close-up of the left one. Data is shown for both the Toulouse and Saint-Léger test sites.

Although not for all cases, it seems that *not allowing* “translated and scaled” models gives better results. This is surprising, given that the estimation procedure evaluates all available models and selects the best one according to the Bayes Information Criterion. Therefore, when “translated and scaled” models are allowed, if they are not actually the best model, the “translated” or “flat” will be chosen. The fact that it is better to not even allow “translated and scaled” models would mean that a model selected as “best” by BIC is not actually the best one. Since the differences are not very large, especially for the best-performing algorithms, this may simply be due to the fact that the BIC, and the estimated models, are calculated using the training reference, while classification accuracy is calculated using the testing reference. On the other hand, it might mean that the BIC is not such a good criterion for this kind of choice.

5.9.6 Comparison of classification algorithms

In previous sections I presented a large quantity of classification algorithms. Some, notably those in sections 5.4.5 and 5.4.6, seemed interesting but did not have a sound mathematical justification. In this section we will compare the performances of each group of algorithms.

Tables 5.9, 5.10, and 5.11 compare these groups of algorithms, at 100% coverage, 95% coverage, and 75% coverage respectively, for different test sites, input channels, and other parameters. We can see that the “q” and the “eq” algorithms, of sections 5.4.5 and 5.4.6 respectively, perform better, in some cases, than the remaining nested classifiers. This justifies the inclusion of these less mathematically justified classification methods.

These tables also show a very important result: At full coverage, when no pixel is rejected, flat per-region classification works best for Saint-Léger, and standard majority vote works best for Toulouse, but in no case is nested per-region classification the best performer. Does this mean that these new methods are useless?

No. First, although they are not the top performers in any of the cases shown in table 5.9, they are not far behind. Perhaps some further research could yield improved figures. But,

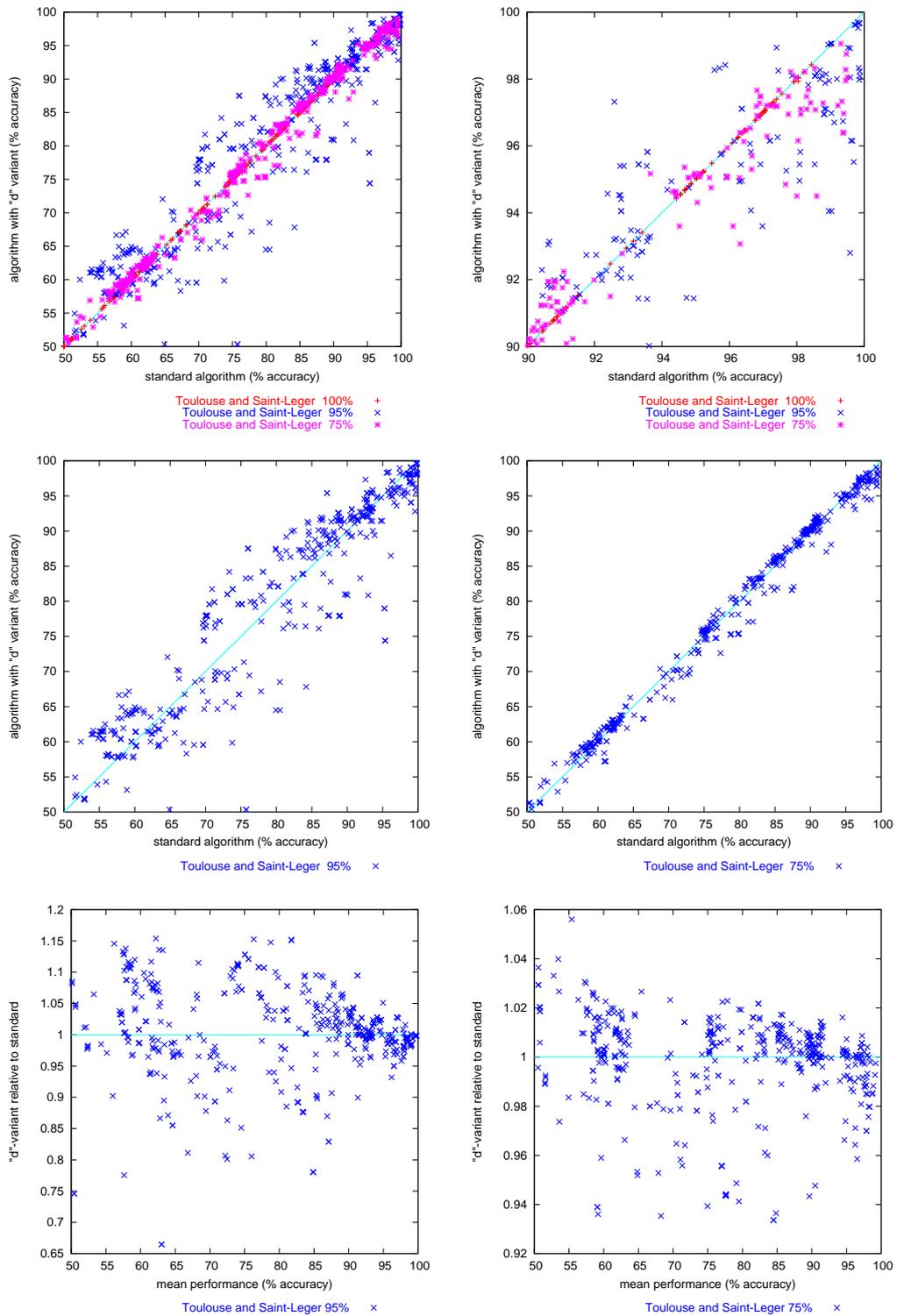


Figure 5.27: Plots showing a comparison of classifications using Yuille’s confidence measure or those presented in this chapter.

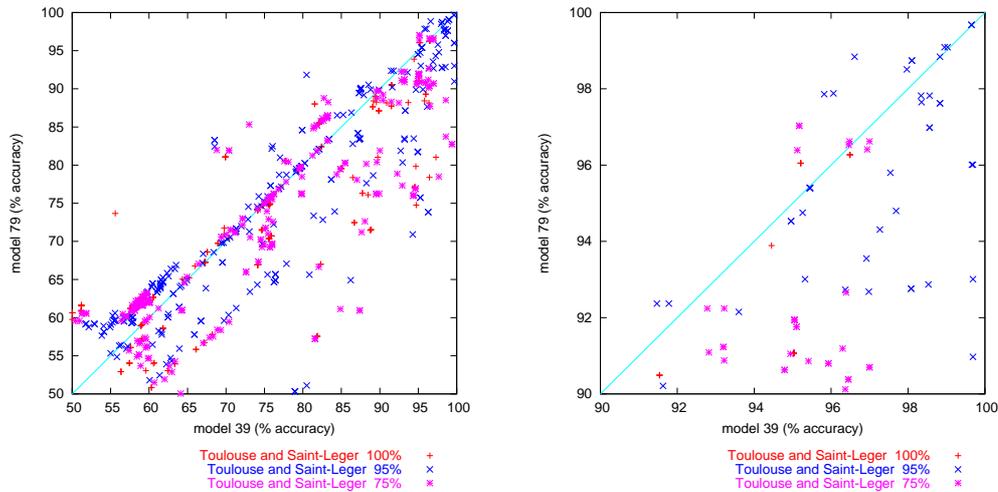


Figure 5.28: Scatterplots showing a comparison of classifications using different nested estimations.

chan.	regions	09 pix.	09 maj.	09 flat	39 nested	39 eq. . .	39 q. . .	79 nested	79 eq. . .	79 q. . .
stl2	cad.	95.83%	97.02%	97.40%	96.39%	96.49%	95.21%	88.16%	96.27%	96.05%
stl2	segm.	95.25%	97.32%	99.04%	97.30%	91.54%	89.45%	81.03%	90.49%	88.97%
stl3	segm.	94.80%	96.09%	98.43%	94.73%	95.98%	75.75%	74.76%	89.30%	74.93%
tor2	segm.	89.48%	90.79%	89.31%	88.90%	69.77%	50.02%	71.53%	71.74%	60.64%
tor2x	segm.	90.02%	90.81%	89.02%	88.47%	82.41%	34.26%	76.11%	82.41%	34.26%
tor3	segm.	89.61%	91.34%	91.16%	n/c	n/c	n/c	n/c	n/c	n/c

Table 5.9: Classification accuracies, for different test sites, source of image regions, and input channels, of the best-performing classification algorithm for different algorithm groups, at 100% coverage. *Chan* is the test site and channel selection of table 5.1. *Regions* is *segm* when a fine segmentation was used as source of regions, and *cad* when the registered cadastre was used. Column *pix* groups all per-pixel classifiers. Column *maj* corresponds to the majority vote after a MAP per-pixel classification. Column *flat* corresponds to all flat per-region classifiers. Column *eq* corresponds to the best performing of all algorithms presented in section 5.4.6. Column *q* corresponds to the best performing of all algorithms presented in section 5.4.5. Column *nested* corresponds to the best performing of all algorithms presented elsewhere in section 5.4. *N/c* indicates that tests were not run for that combination. *09*, *39*, and *79* are the submodels of table 5.2.

more importantly, if we look at the figures for partial coverage of table 5.11, we see that nested classifiers are the best performers for Toulouse, and are the best or very close to the best for Saint-Léger. This means that the confidence measures associated with nested classifiers perform better than the others, since they are more able to reliably detect possible mistakes. Since one of the main goals of this classification system was to obtain these kind of confidence measures, which would allow semi-automatic classification, we must consider this a success.

Note, also, that the actual classification accuracy figures obtained, close to 95% in the difficult test site Toulouse, are exceptionally good for this kind of classification application. The figures for the simpler Saint-Léger test site, over 99%, are excellent.

Tables 5.12 to 5.17 show, for each test site, and for different coverages, the 20 best-performing

chan.	regions	09 pix.	09 maj.	09 flat	39 nested	39 eq...	39 q...	79 nested	79 eq...	79 q...
stl2	cad.	97.12%	96.87%	98.34%	98.59%	97.00%	95.17%	88.51%	96.62%	97.03%
stl2	segm.	96.67%	97.21%	99.31%	99.39%	93.22%	89.92%	82.75%	92.24%	89.73%
stl3	segm.	96.66%	95.94%	99.50%	97.67%	96.45%	76.02%	78.48%	90.38%	75.57%
tor2	segm.	90.21%	91.12%	90.29%	89.94%	72.16%	50.95%	76.27%	72.99%	59.57%
tor2x	segm.	90.61%	91.00%	89.56%	90.01%	83.19%	33.67%	81.90%	83.19%	33.22%
tor3	segm.	90.22%	91.99%	91.71%	n/c	n/c	n/c	n/c	n/c	n/c

Table 5.10: Classification accuracies, for different test sites, source of image regions, and input channels, of the best-performing classification algorithm for different algorithm groups, at 95% coverage. Columns are as in table 5.9.

chan.	regions	09 pix.	09 maj.	09 flat	39 nested	39 eq...	39 q...	79 nested	79 eq...	79 q...
stl2	cad.	98.59%	95.77%	99.06%	99.03%	98.83%	98.10%	99.09%	98.84%	98.74%
stl2	segm.	98.34%	96.15%	99.84%	99.65%	98.56%	91.78%	99.68%	97.82%	92.37%
stl3	segm.	99.37%	95.74%	99.87%	99.69%	99.69%	77.87%	93.01%	96.01%	78.72%
tor2	segm.	93.15%	91.45%	93.41%	94.41%	75.80%	54.93%	83.48%	77.30%	58.76%
tor2x	segm.	93.59%	91.56%	93.48%	94.69%	83.42%	35.17%	89.89%	83.42%	30.33%
tor3	segm.	92.22%	92.14%	94.57%	n/c	n/c	n/c	n/c	n/c	n/c

Table 5.11: Classification accuracies, for different test sites, source of image regions, and input channels, of the best-performing classification algorithm for different algorithm groups, at 75% coverage. Columns are as in table 5.9.

classification algorithms and channel selections. For Saint-Léger, only an image segmentation was used as a source of regions to classify. For each algorithm and channel selection, we give the overall classification accuracy, and the kappa parameter. Note that in these evaluations, the classes *tillable*, *field*, and *vineyard* are merged into one, and the classes *forest* and *orchard* are merged too. The reason for merging *tillable* and *field*, as explained in section 5.7, is that it was difficult to distinguish them when preparing reference data. The reason for the other merges is that these classes are easily confused by radiometry and standard texture descriptors alone (see figures 5.14 to 5.17), and it is better to use the algorithm presented in chapter 6 to separate them than to try to do it in this step. In the *images* showing classification maps, however, we have kept them separated.

Finally, we show some graphical results in figures 5.29 to 5.39. Figure 5.29 is the class map for the algorithm that performs best in Saint-Léger at 75% coverage using a segmentation as partition, that is, *Flat MAPnn* with *stl3* channels. In this image, hues indicate the selected class for each pixel, and intensities indicate the confidence measure, as explained in figure 5.23. That is, classifications made with confidence are shown in brighter colours, while those that the algorithm is more unsure of are shown in darker colours.

Figure 5.30 is the class map for the best-performing *per-pixel* algorithm for Saint-Léger, at 75% coverage. It's the *Pixel MAP* algorithm with *stl3* channels, giving 99.37% accuracy and $100\kappa = 98.11$. Note in the close-up of figure 5.31 the salt-and-pepper noise, despite the high classification accuracy.

Figure 5.32 is the class map for the best-performing algorithm for Saint-Léger, at 100% coverage, using the registered cadastre. It's the *Flat MAPnn* algorithm with *stl2* channels, giving 97.40% accuracy and $100\kappa = 93.24$. Note that, since regions are larger, salt-and-pepper noise is reduced compared to figures 5.29 and 5.30.



Figure 5.29: Classification map for Saint-Léger, with the *Flat MAPnn* algorithm with *stl3* and segmentation as region source. North is to the left of the page.



Figure 5.30: Classification map for Saint-Léger, with the *Pixel MAP* algorithm with *stl3* and segmentation as region source. North is to the left of the page.



Figure 5.31: Close-up of figure 5.30, a per-pixel classification map for Saint-Léger. Note the salt-and-pepper noise.



Figure 5.32: Classification map for Saint-Léger, with the *Flat MAPnn* algorithm with *stl2* and registered cadastre. North is to the left of the page.

channels	model		algorithm	accu.	$100 \cdot \kappa$
stl2	09	Flat	MAPnn[d]	99.04%	97.52
stl3	09	Flat	KLs[d]	98.43%	95.84
stl3	09	Flat	KL[d]	98.21%	95.25
stl3	09	Flat	MLn	98.04%	94.81
stl3	09	Flat	MLnnd	98.04%	94.81
stl3	09	Flat	ML[d]	98.04%	94.81
stl3	09	Flat	MAPn	98.04%	94.81
stl3	09	Flat	MAP[d]	98.04%	94.81
stl2	09	Flat	Chi2n	97.92%	94.70
stl2	09	Flat	Chi2[d]	97.92%	94.70
stl3	09	Flat	MAPo[d]	97.56%	93.57
stl2	09	Majority	MAP[d]	97.32%	93.13
stl2	39	Nested	MAPnn[d]	97.30%	92.77
stl2	39	Nested	MLnnd	97.25%	92.87
stl2	09	Flat	MAPn	97.02%	92.49
stl2	09	Flat	MAP[d]	97.02%	92.49
stl2	09	Flat	MLn	96.92%	92.24
stl2	09	Flat	MLnnd	96.92%	92.24
stl2	09	Flat	ML[d]	96.92%	92.24
stl2	09	Flat	KL[d]	96.91%	92.23

Table 5.12: Twenty best-performing classification algorithms for Saint-Léger at 100% coverage, using an image segmentation as source of regions. *Channels* is the test site and channel selection of table 5.1. *Model* is the submodel of table 5.2. *Algorithm* is the algorithm type and name as defined in table 5.8. At 100% coverage, the confidence measure is not taken into account, so algorithms with Yuille’s variant are equivalent to those without; they are marked with “[d]”. *Accu.* is the global classification accuracy, and $100 \cdot \kappa$ is the kappa parameter, multiplied by 100.

Figure 5.33 and 5.34 show the class map for the best-performing algorithm for Toulouse at 75% coverage, *Nested MAPnnd* with *tor2x 39*, split in two halves.

Figure 5.35 and 5.36 show the class map for the best-performing *per-pixel* algorithm for Toulouse at 75% coverage, *Pixel MAP* with *tor2x*, also split in two halves. Notice in the close-up of figure 5.37 the increased salt-and-pepper noise compared to figures 5.33 and 5.34.

For comparison, finally, we show in figures 5.38 and 5.39 the class map for one badly-performing algorithms for Toulouse, *Nested qMLnv* with *tor2 39*, giving 39.15% accuracy and $100\kappa = 13.05$, again split in two.

5.9.7 Study of the Nested MAPnnd algorithm

We will now study in more detail the algorithm that gives the best results for Toulouse, namely, the *Nested MAPnnd* algorithm with *tor2x 39*, using a segmentation as image partition.

Figures 5.40 and 5.41 show how the global accuracy for this algorithm behaves related to the confidence measure (merging some classes as described before for figure 5.40, and not doing so for figure 5.41). We can see that classification accuracy and the kappa parameter increase as we reject a larger fraction of the pixels classified with bad confidence. This has two implications: first, that the confidence measure does indeed reflect the difficulty of classifying a certain region, and that lower confidences indicate a higher probability of misclassification.

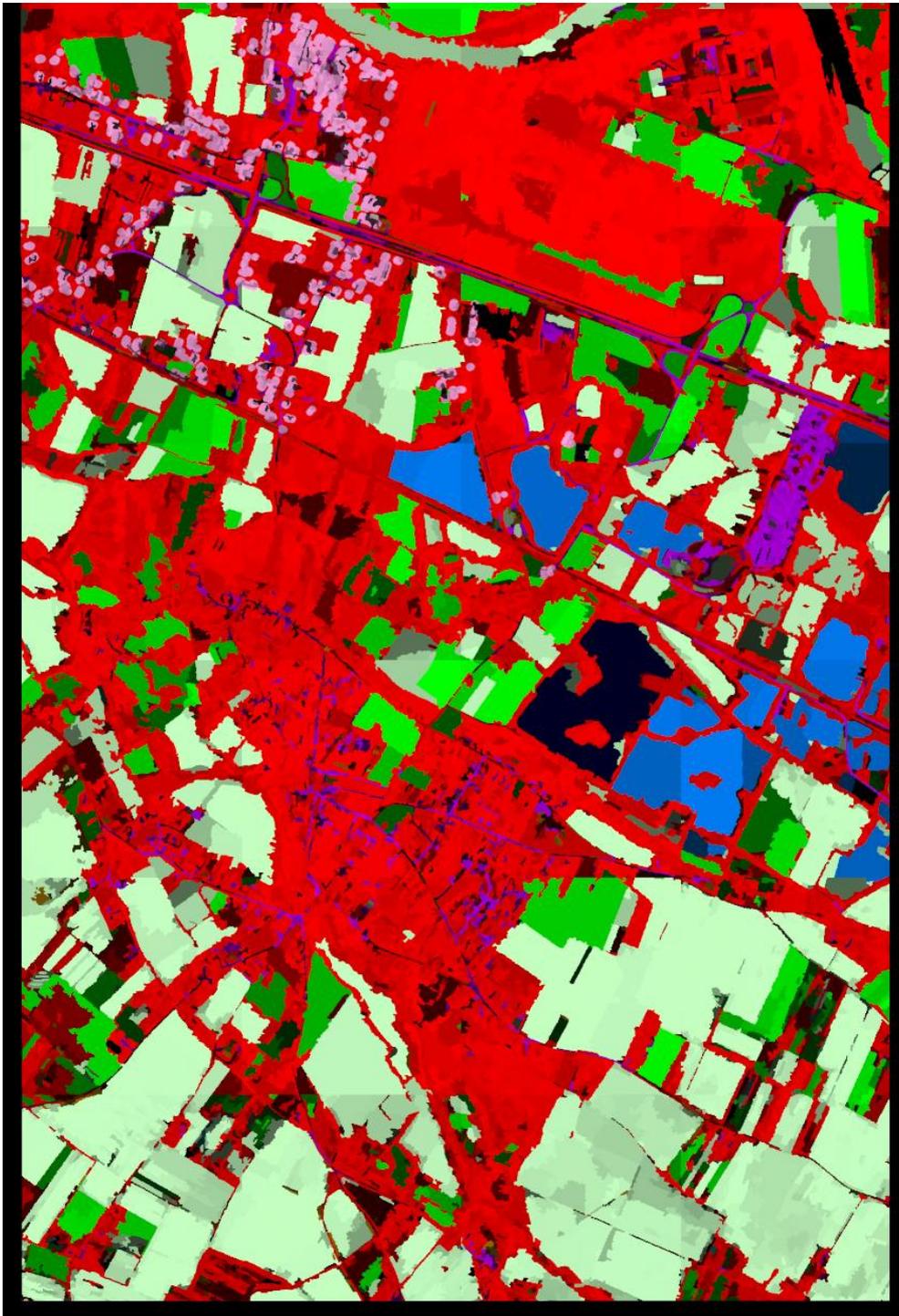


Figure 5.33: Western half of the classification map for Toulouse, with the *Nested MAP_{nnd}* algorithm with *tor2x 39* and segmentation as region source. North is to the left of the page.

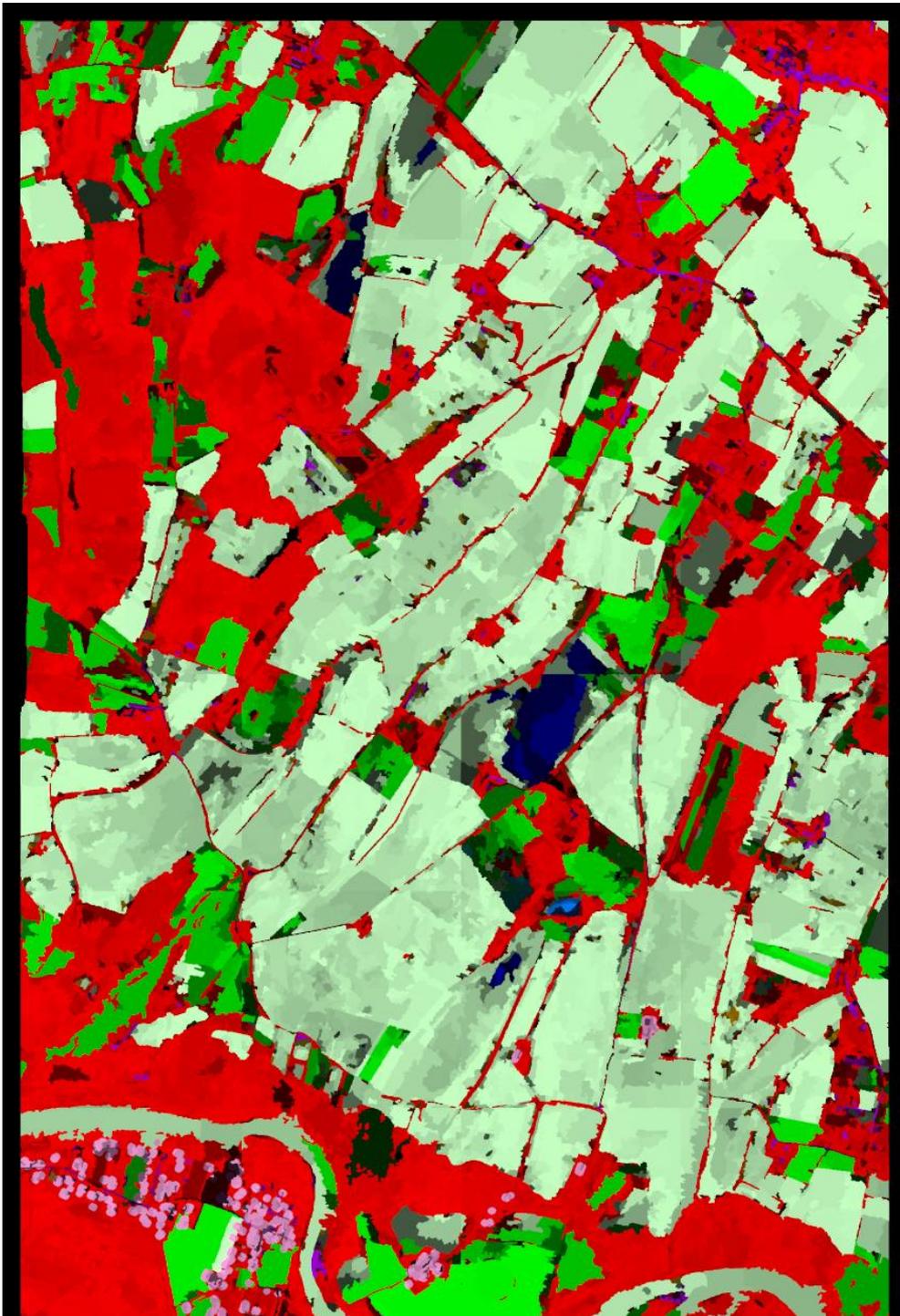


Figure 5.34: Eastern half of the classification map for Toulouse, with the *Nested MAP_{nd}* algorithm with *tor2x 39* and segmentation as region source. North is to the left of the page.

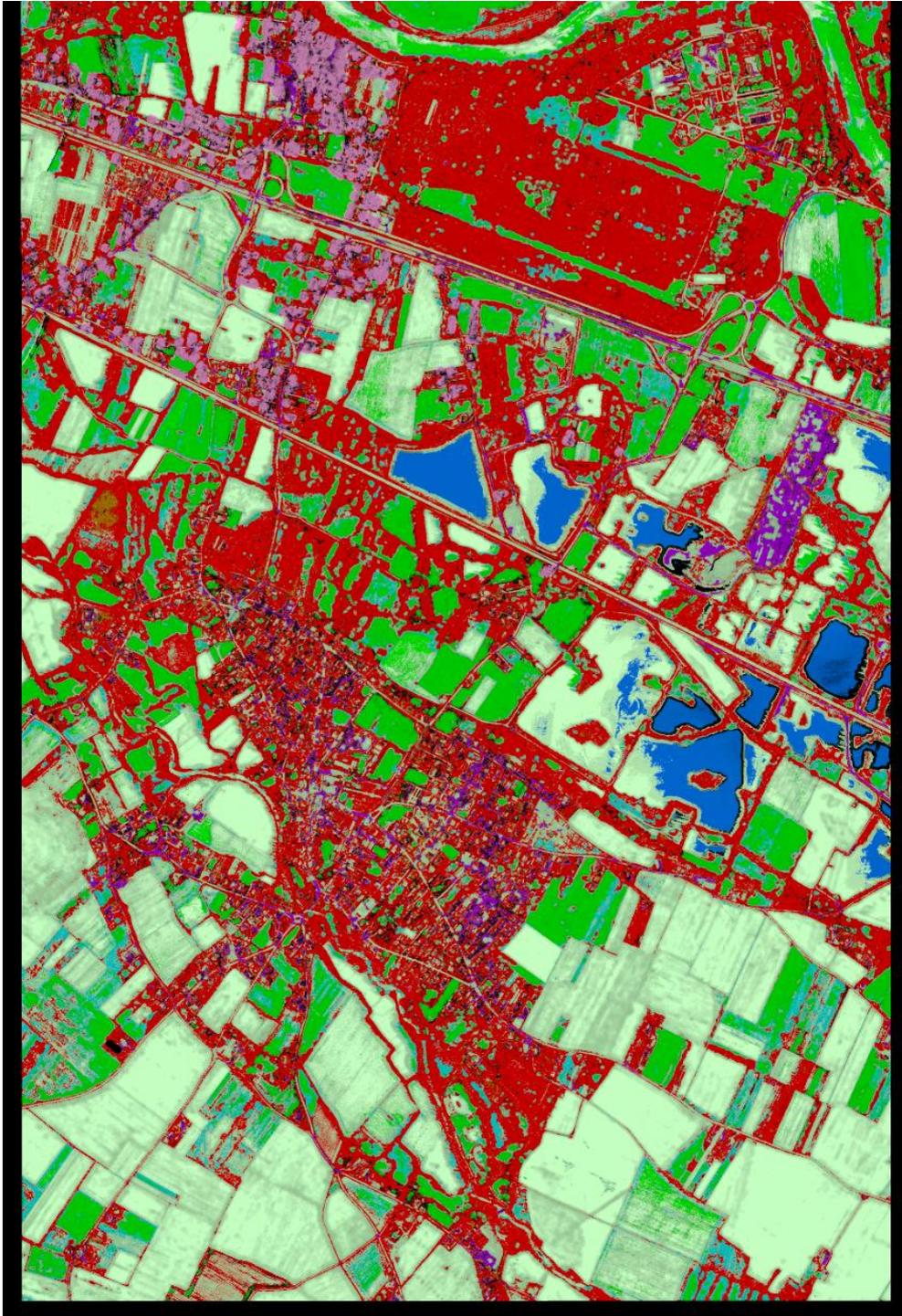


Figure 5.35: Western half of the classification map for Toulouse, with the *Pixel MAP* algorithm with *tor2x* and segmentation as region source. North is to the left of the page.



Figure 5.36: Eastern half of the classification map for Toulouse, with the *Pixel MAP* algorithm with *tor2x* and segmentation as region source. North is to the left of the page.

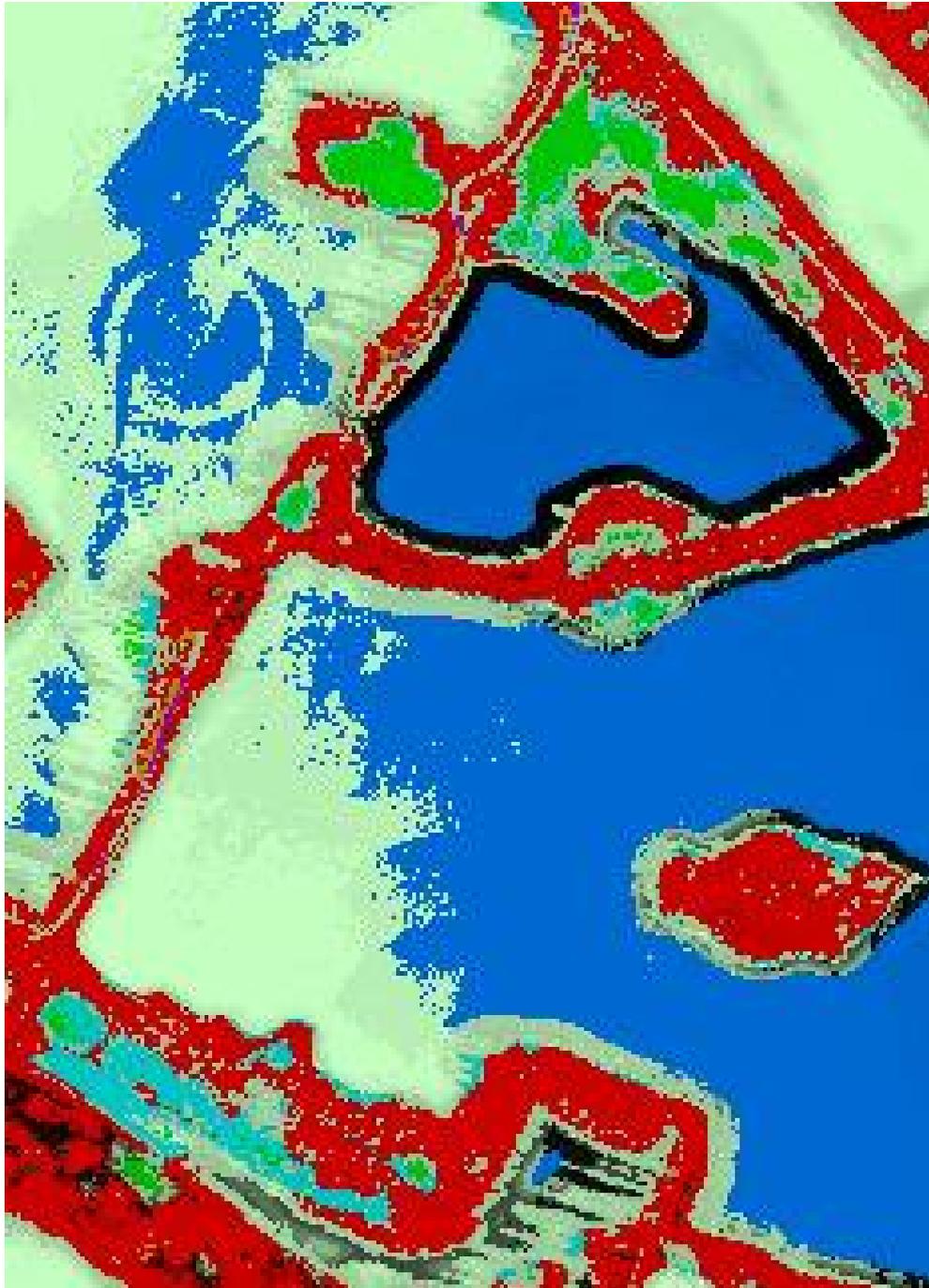


Figure 5.37: Close-up of figure 5.35, a per-pixel classification map for Toulouse. Note the salt-and-pepper noise.

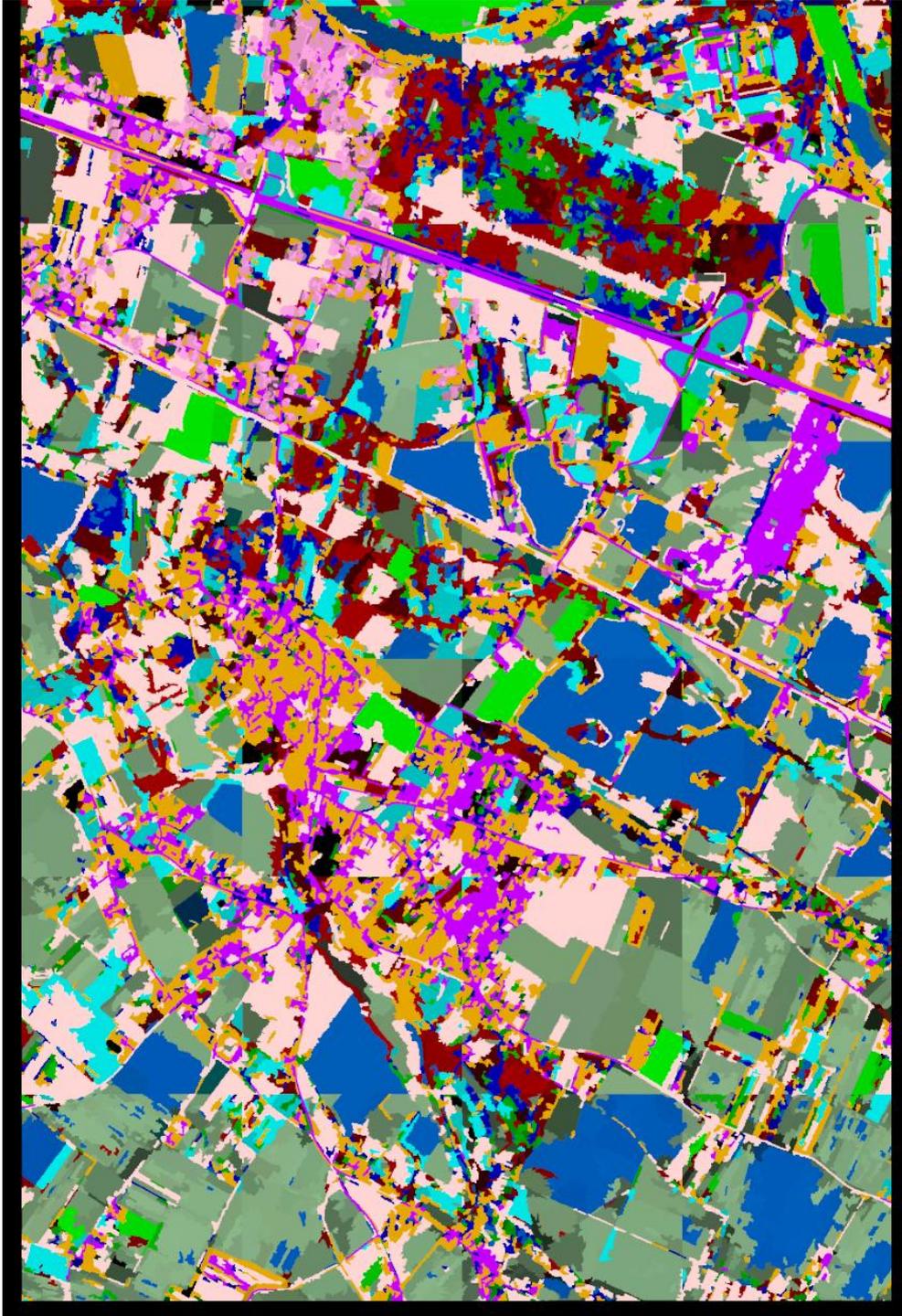


Figure 5.38: Western half of the classification map for Toulouse, with one badly-performing algorithm, the *Nested qMLnv* algorithm with *tor2 39* and segmentation as region source. North is to the left of the page.

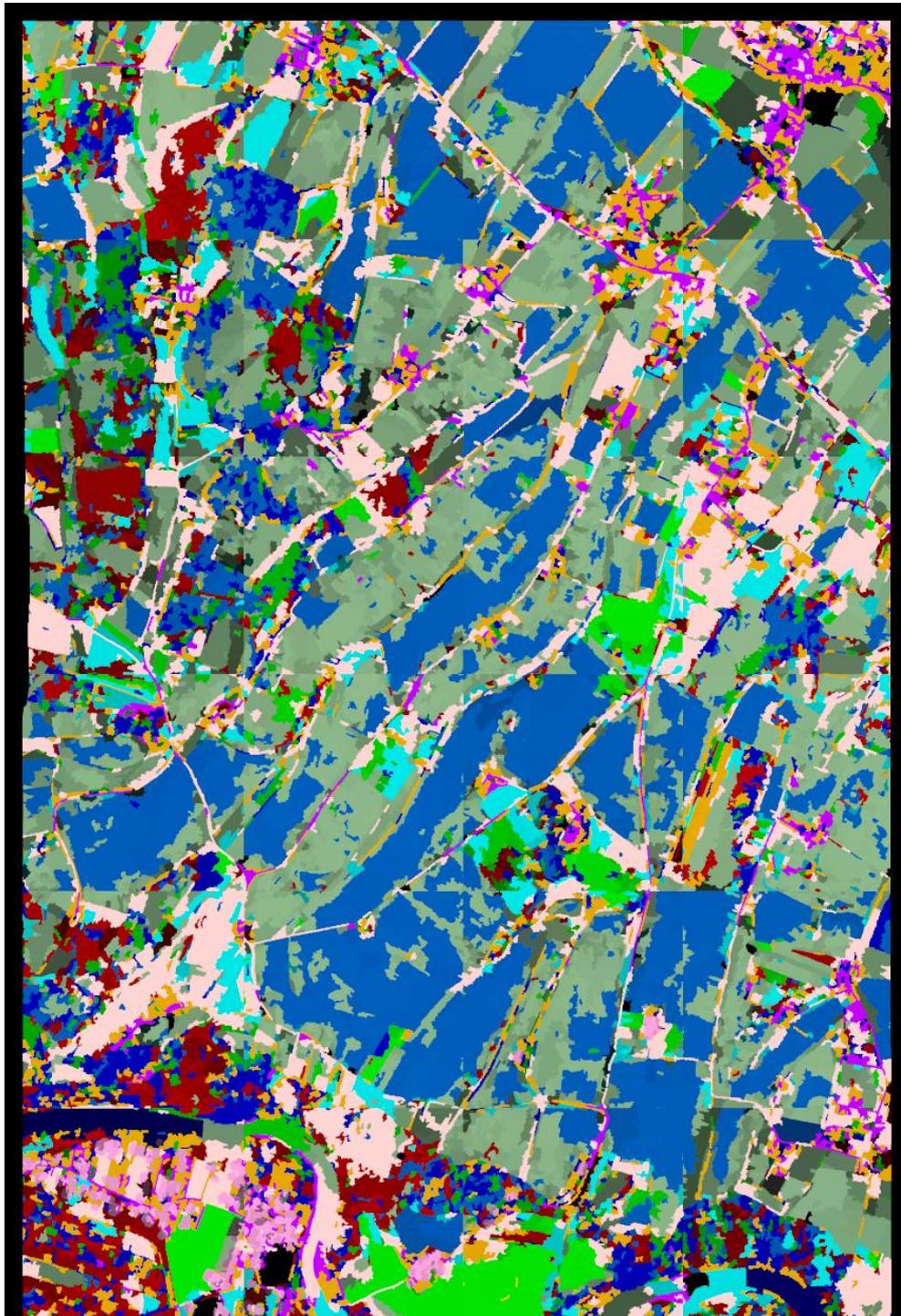


Figure 5.39: Eastern half of the classification map for Toulouse, with one badly-performing algorithm, the *Nested qMLnv* algorithm with *tor2 39* and segmentation as region source. North is to the left of the page.

channels	model	algorithm		accu.	$100 \cdot \kappa$
stl3	09	Flat	MAPn	99.50%	98.59
stl3	09	Flat	KLs	99.47%	98.56
stl3	09	Flat	MLn	99.45%	98.49
stl3	09	Flat	MAPnn	99.39%	98.28
stl2	39	Nested	MAPn	99.39%	98.30
stl2	39	Nested	MLn	99.37%	98.27
stl2	39	Nested	MAPnn	99.34%	98.18
stl2	09	Flat	MAPnn	99.31%	98.16
stl3	09	Flat	Chi2	99.22%	97.74
stl2	09	Flat	MAPnnd	99.06%	97.63
stl2	09	Flat	MAPn	98.78%	96.87
stl3	09	Flat	KLsd	98.75%	96.75
stl3	09	Flat	KL	98.53%	96.06
stl2	39	Nested	MLnn	98.41%	95.81
stl3	09	Flat	KLd	98.33%	95.68
stl3	09	Flat	MLnnd	98.28%	95.56
stl2	09	Flat	Chi2n	98.24%	95.45
stl2	09	Flat	Chi2d	98.22%	95.51
stl3	09	Flat	MLd	98.05%	94.75
stl3	09	Flat	MAPd	98.05%	94.75

Table 5.13: Twenty best-performing classification algorithms for Saint-Léger at 95% coverage, using an image segmentation as source of regions. Columns are as in table 5.12.

channels	model	algorithm		accu.	$100 \cdot \kappa$
stl3	09	Flat	MAPnn	99.87%	99.59
stl3	09	Flat	MAPn	99.87%	99.58
stl3	09	Flat	KL	99.86%	99.62
stl3	09	Flat	MLn	99.85%	99.57
stl2	09	Flat	MAPn	99.84%	99.58
stl2	09	Flat	MAPnn	99.82%	99.55
stl3	09	Flat	KLs	99.80%	99.47
stl3	09	Flat	Chi2n	99.75%	99.05
stl3	09	Flat	Chi2	99.71%	98.96
stl3	09	Flat	KLsd	99.70%	99.29
stl3	39	Nested	MLnn	99.69%	99.11
stl3	39	Nested	eqKLsvo	99.69%	99.19
stl3	09	Flat	MLnnd	99.68%	99.23
stl2	79	Nested	MLn	99.68%	97.08
stl3	09	Flat	KLd	99.67%	99.22
stl3	39	Nested	eqKLvo	99.66%	99.06
stl2	39	Nested	MLn	99.65%	99.17
stl3	09	Flat	Chi2d	99.62%	98.96
stl2	39	Nested	MAPn	99.61%	98.99
stl2	39	Nested	MAPnn	99.59%	98.88

Table 5.14: Twenty best-performing classification algorithms for Saint-Léger at 75% coverage, using an image segmentation as source of regions. Columns are as in table 5.12.

channels	model	algorithm		accu.	$100 \cdot \kappa$
tor3	09	Majority	MAP[d]	91.34%	80.22
tor3	09	Flat	MAPn	91.16%	81.44
tor3	09	Flat	MAP[d]	91.16%	81.44
tor3	09	Flat	MLn	91.15%	81.42
tor3	09	Flat	MLnnd	91.15%	81.42
tor3	09	Flat	ML[d]	91.15%	81.42
tor3	09	Flat	MAPo[d]	91.00%	81.07
tor2x	09	Majority	MAP[d]	90.81%	79.47
tor2	09	Majority	MAP[d]	90.79%	79.15
tor3	09	Flat	KL[d]	90.76%	80.71
tor3	09	Flat	MAPnn[d]	90.45%	79.24
tor2x	09	Pixel	MAP[d]	90.02%	77.39
tor3	09	Flat	Chi2n	89.94%	79.06
tor3	09	Flat	Chi2[d]	89.94%	79.06
tor3	09	Pixel	MAP[d]	89.61%	76.06
tor2	09	Pixel	MAP[d]	89.48%	75.86
tor2	09	Flat	MAPnn[d]	89.31%	77.10
tor2x	09	Flat	MAPnn[d]	89.02%	76.55
tor2	39	Nested	MAPnn[d]	88.90%	76.19
tor2x	39	Nested	MAPnn[d]	88.47%	76.25

Table 5.15: Twenty best-performing classification algorithms for Toulouse at 100% coverage. At 100% coverage, the confidence measure is not taken into account, so algorithms with Yuille’s variant are equivalent to those without; they are marked with “[d]”. Columns are as in table 5.12.

Second, that depending on the accuracy requirements of the application, a correspondingly appropriate coverage ratio can be selected.

Note that these graphs are similar to “receiver operating characteristic” (ROC) graphs, but do not show the same information. ROC graphs are used in two-class problems, and show the true positive ratio as a function of the false positive ratio, that is, the ratio of positive samples detected as positive to the total number of positive samples, as a function of the ratio of negative samples detected as positive to the total number of positive samples. The given *accuracy* and *coverage* ratios are also similar, but different, to the *recall* (number of relevant retrieved objects in a search relative to the total number of relevant objects in the test area) or the *precision* (number of relevant retrieved objects relative to the total number of retrieved objects), which are used in database search applications.

Figure 5.42 shows the user and producer accuracy as a function of coverage, separately for several classes. We can see that, when each class is studied separately, confidence measures behave as expected in some cases (accuracy increases as coverage decreases), but do not do so in some other cases. Probably, investigating why it does not always behave correctly, and solving this problem, would increase performances significantly.

Finally, in table 5.18 we show the confusion matrix for this test. Each row and column corresponds to a terrain type (or shows a per-row or per-column total). The value in a certain row i and column j is the number of pixels that were classified as belonging to class j , when in fact, according to the ground truth, they were of class i . Pixels in the diagonal of the matrix, therefore, are correctly classified, and pixels elsewhere in the matrix are classification errors.

We can see from the user and producer accuracies that, although the overall classification accuracy is high, some classes are not so well classified.

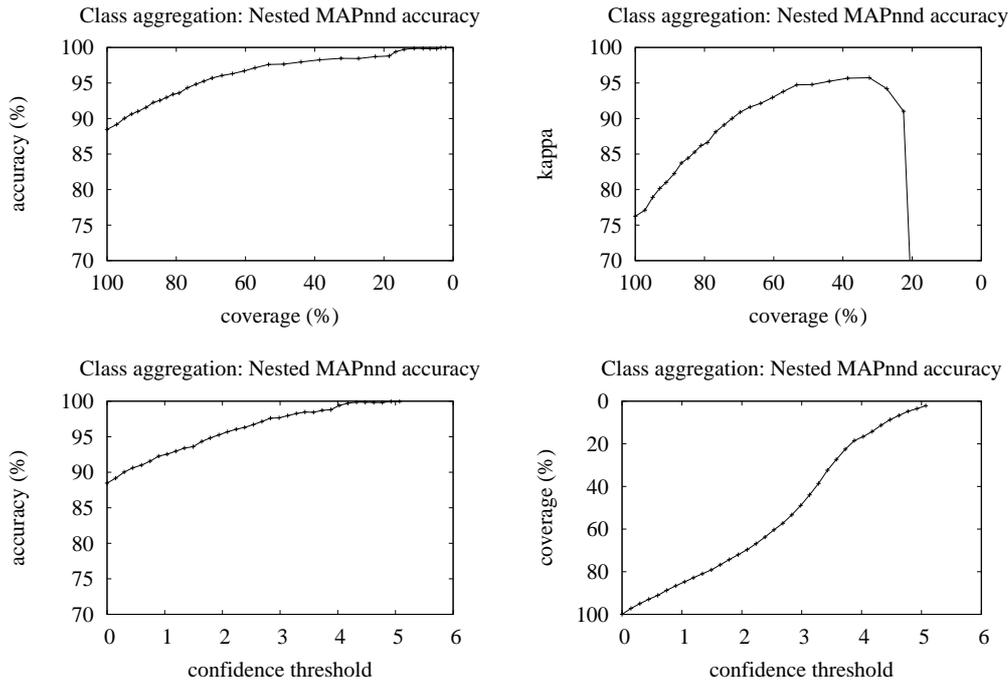


Figure 5.40: Plots showing the performance of the *Nested MAPnnd* algorithm with *tor2x 39* using a segmentation as image partition. Shown are global performances, merging the *tillable*, *field*, and *vineyard* classes, and the *forest* and *orchard* classes. Top left is the classification accuracy as a function of coverage —as a larger fraction of the result is discarded, the accuracy of the remaining classification improves, thus showing that the confidence measure does indicate difficult areas. Top right is the kappa parameter as a function of coverage; performance also increases with decreasing coverage, but it breaks down after a certain point. This happens at coverages too small for our application, anyway. Bottom left is the classification accuracy as a function of the selected threshold for the confidence measure—all pixels with lower confidence are rejected; the higher the threshold, the more pixels are rejected, and the accuracy increases, which again shows that the confidence measure is correct. Finally, bottom right is the coverage as a function of the threshold in the confidence measure, which has the expected behaviour.

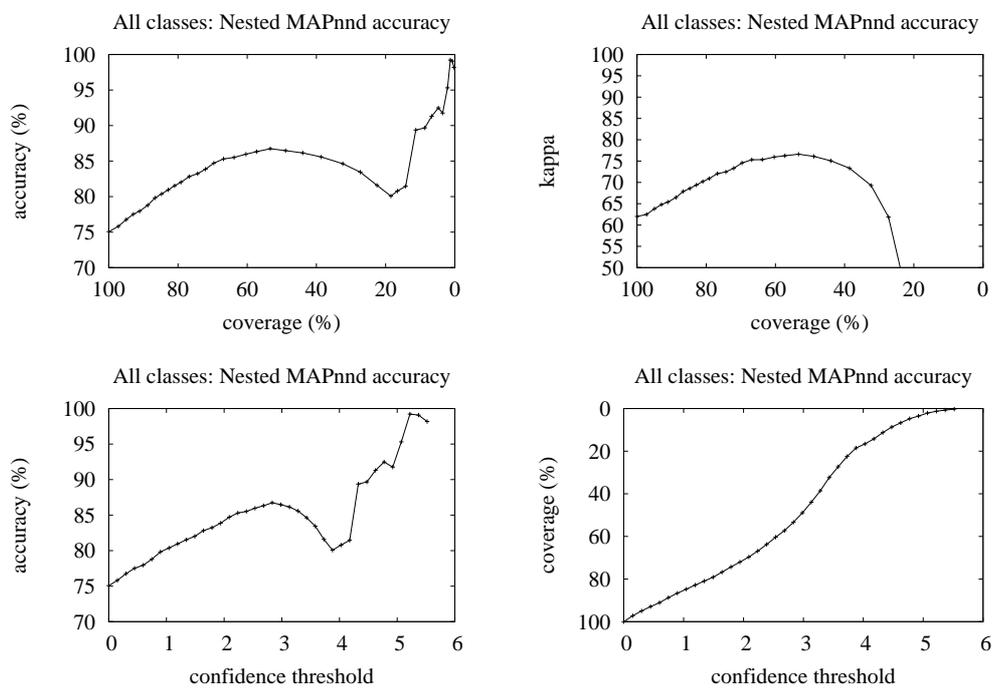


Figure 5.41: Plots showing the performance of the *Nested MAPnnd* algorithm with *tor2x 39* using a segmentation as image partition. Shown are global performances, not merging any class. Graphs are as in figure 5.40. Note that the confidence measures give correct values for coverages above 60% approximately, but behave erratically below that.

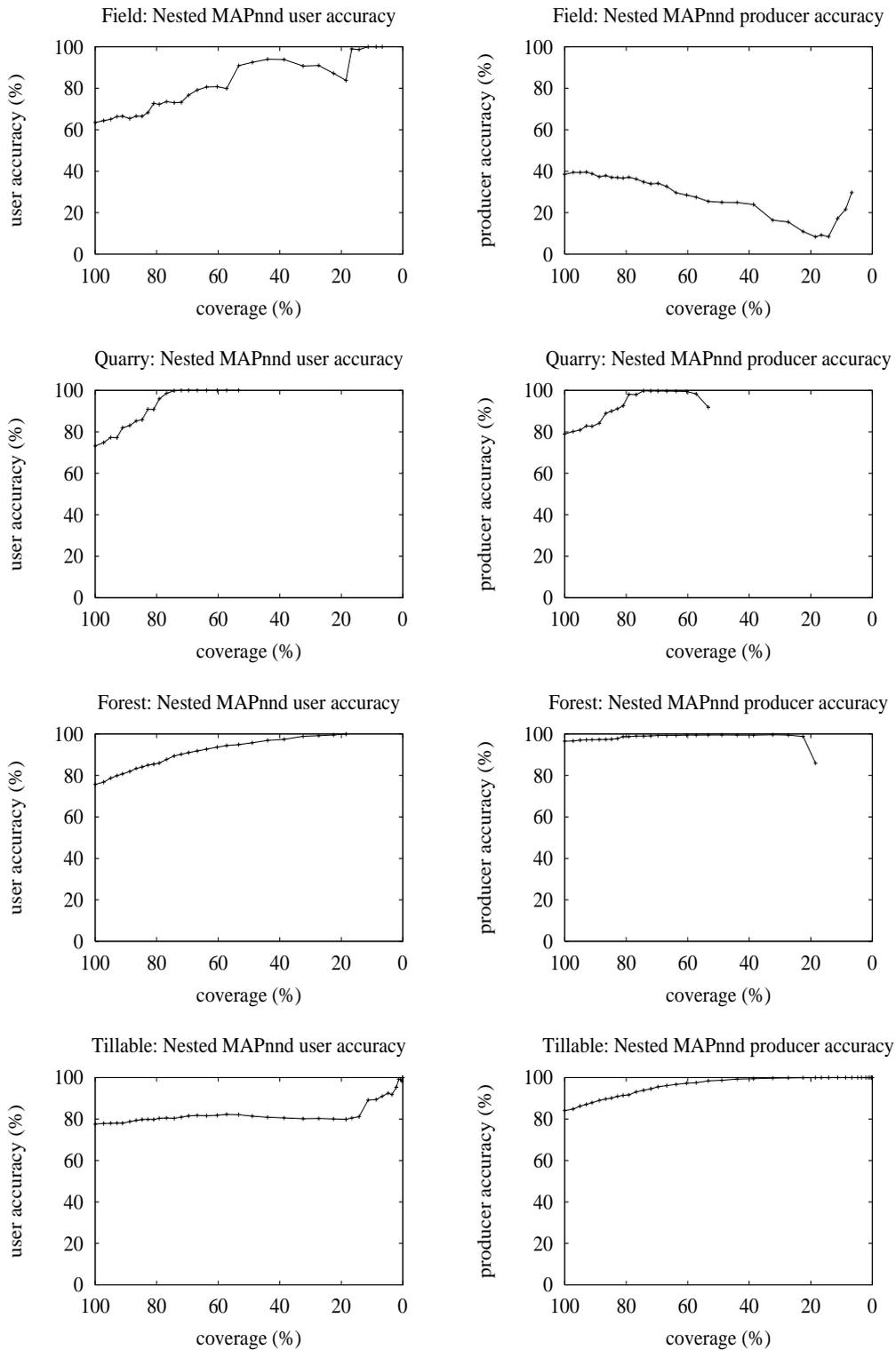


Figure 5.42: Plots showing the performance of the *Nested MAPnnd* algorithm with *tor2x 39* using a segmentation as image partition. Shown are user and producer accuracies (left and right respectively) as a function of coverage, separately for the *field*, *quarry*, *forest*, and *tillable* classes.

channels	model	algorithm		accu.	$100 \cdot \kappa$
tor3	09	Majority	MAP	91.99%	81.86
tor3	09	Majority	MAPd	91.99%	81.86
tor3	09	Flat	MAPn	91.71%	82.39
tor3	09	Flat	MAPnnd	91.45%	81.51
tor3	09	Flat	MLnnd	91.44%	82.33
tor3	09	Flat	MLn	91.35%	81.84
tor3	09	Flat	MAPd	91.34%	81.81
tor3	09	Flat	MLd	91.33%	81.80
tor3	09	Flat	MAPod	91.20%	81.47
tor2	09	Majority	MAP	91.12%	79.99
tor2	09	Majority	MAPd	91.12%	79.99
tor3	09	Flat	KLd	91.06%	81.64
tor2x	09	Majority	MAP	91.00%	80.06
tor2x	09	Majority	MAPd	91.00%	80.06
tor3	09	Flat	Chi2	90.94%	80.62
tor3	09	Flat	MAP	90.90%	80.89
tor3	09	Flat	KL	90.90%	80.99
tor3	09	Flat	ML	90.89%	80.87
tor3	09	Flat	Chi2d	90.75%	80.96
tor3	09	Flat	MAPo	90.72%	80.49

Table 5.16: Twenty best-performing classification algorithms for Toulouse at 95% coverage. Columns are as in table 5.12.

channels	model	algorithm		accu.	$100 \cdot \kappa$
tor2x	39	Nested	MAPnnd	94.69%	88.83
tor3	09	Flat	MAPnnd	94.57%	88.69
tor2	39	Nested	MAPnnd	94.41%	87.73
tor2x	39	Nested	MAPnn	94.16%	87.93
tor2x	09	Pixel	MAP	93.59%	84.43
tor2x	09	Flat	MAPnnd	93.48%	86.82
tor2	09	Flat	MAPnn	93.41%	85.92
tor2x	09	Pixel	MAPd	93.26%	84.73
tor2	09	Pixel	MAP	93.15%	83.15
tor2x	09	Flat	MAPnn	93.08%	85.23
tor3	09	Flat	MAPn	93.07%	85.51
tor2	09	Pixel	MAPd	93.00%	83.64
tor2	09	Flat	MAPnnd	92.84%	85.42
tor3	09	Flat	MLn	92.79%	85.19
tor2	39	Nested	MAPnn	92.79%	84.38
tor3	09	Flat	Chi2d	92.74%	86.11
tor3	09	Flat	MAPnn	92.39%	83.65
tor3	09	Flat	MLnnd	92.32%	85.35
tor3	09	Pixel	MAP	92.22%	82.03
tor3	09	Flat	KL	92.21%	83.76

Table 5.17: Twenty best-performing classification algorithms for Toulouse at 75% coverage. Columns are as in table 5.12.

	field	forest	orchard	lake	river	quarry	tillable	vine	prod.	total
unclassified	711393	3064650	73303	132697	28436	171920	2061170	9234	0%	6252800
field	247400	90547	1705	0	10353	0	291632	1156	38.49%	642793
forest	19755	813517	70	7	147	80	9142	258	96.51%	842976
orchard	12290	22843	1948	0	0	13	1838	0	5.00%	38932
lake	202	2338	33	85409	0	22	19067	0	79.77%	107071
river	4289	466	481	0	0	85	27983	0	0%	33304
quarry	0	780	336	0	0	7080	770	0	78.97%	8966
tillable	81430	132090	12422	52	1835	1501	1212640	470	84.07%	1442440
vine	24345	12550	0	0	0	0	56	117	0.32%	37068
user	63.48%	75.66%	11.35%	99.93%	0%	73.22%	77.58%	5.85%		
total	1101100	4139860	90453	218165	40771	181590	3624300	11235	9407480	

Table 5.18: Confusion matrix for the *Nested MAP_{nnd}* algorithm with *tor2x 39* using a segmentation as image partition. Classes in rows are those in the ground truth, classes in columns are the output of the algorithm. *Prod* is the producer accuracy, and *user* is the user accuracy, in percentage. All other figures are in number of pixels.

5.9.8 Execution time

As for model estimation, computation time required for classification varies significantly, from 7 s/km² to more than 60 min/km², depending on the specific classification algorithm used, the number of input channels, and other factors.

Figure 5.43 shows the histogram of required classification time, in seconds per square kilometre, using only data for tests on Toulouse with a fine segmentation as partition. Figure 5.44 shows the same data, separately for nested and flat classification. As expected, nested classification is slower.

We can see that, although for some classification algorithms computation time is too high, for most algorithms it is reasonable. For example, table 5.19 gives the classification time, in seconds per square kilometre, for the top five algorithms of table 5.17. It may make sense, however, to use the *second* algorithm of this table, instead of the first, since the loss in classification accuracy is small, but the algorithm is more than ten times faster.

channels	model	algorithm	accu.	100 · κ	time
tor2x	39	Nested MAP _{nnd}	94.69%	88.83	845 s/km ²
tor3	09	Flat MAP _{nnd}	94.57%	88.69	70 s/km ²
tor2	39	Nested MAP _{nnd}	94.41%	87.73	229 s/km ²
tor2x	39	Nested MAP _{nn}	94.16%	87.93	881 s/km ²
tor2x	09	Pixel MAP	93.59%	84.43	65 s/km ²

Table 5.19: Classification time, in seconds per square kilometre, of the five best-performing classification algorithms for Toulouse at 75% coverage. Columns are as in table 5.12.

5.10 Conclusion

In this chapter, I have briefly presented traditional Bayesian per-pixel classification, and described its drawbacks, mainly salt-and-pepper noise and lack of use of contextual information.

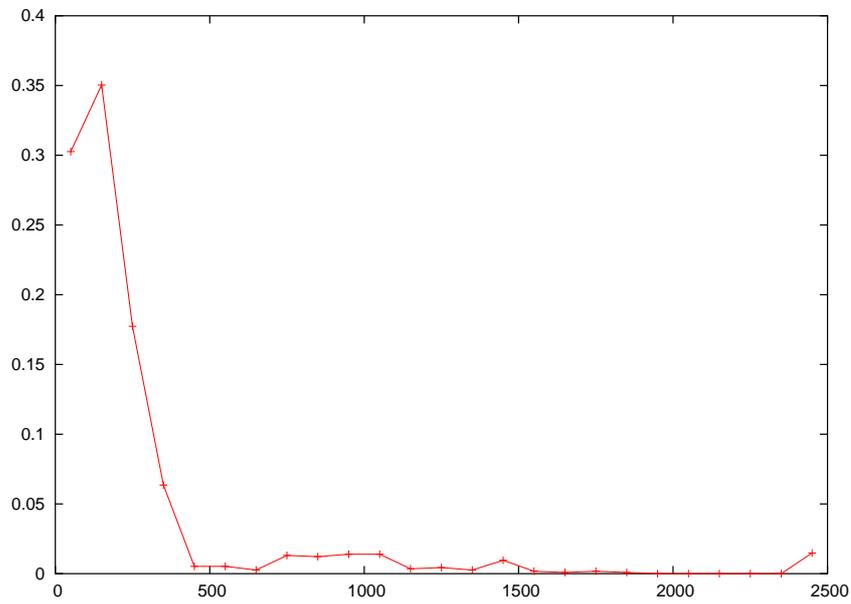


Figure 5.43: Histogram of computation time (in seconds per square kilometre) for terrain classification. The last bin also includes times larger than 2500 s/km².

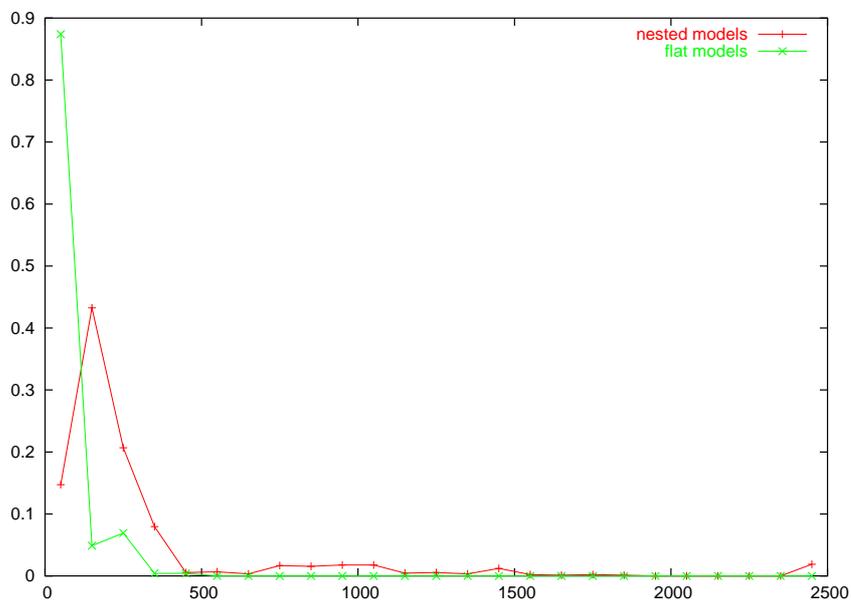


Figure 5.44: Histogram of computation time (in seconds per square kilometre) for terrain classification, separately for nested and flat classification algorithms. The last bin also includes times larger than 2500 s/km².

To solve these problems, I have presented, first, classification algorithms that use standard *flat* probability models but operate region-by-region, thus reducing noise, and, second, a novel *nested* probability model, and new estimation and classification procedures using this new model. These per-region flat and nested classification algorithms also calculate a *classification confidence measure* that can be used to discard difficult-to-classify areas.

I have shown, using the Bayes Information Criterion, that nested models reproduce more faithfully than flat models the kind of data used in remote sensing applications such as this one. Further, I have shown, through extensive tests, that, if we reject the fraction of results having the worst confidence measure, per-region classification (flat, in some cases, and nested, in others) outperform traditional methods. In particular, if the worst 25% of pixels—according to calculated confidence measures—are rejected, we obtain 99.9% classification accuracy for the simpler Saint-Léger test site, and 94.7% classification accuracy for the more complex Toulouse test site. Furthermore, I have shown that, in most cases, the calculated confidence measures are related to the probability of misclassification.

6

Texture orientation and period estimation

6.1 Introduction

As we have seen in chapter 5, it is possible to obtain a land cover and land use classification from high spatial resolution and low spectral resolution images. The low spectral resolution can be compensated by the use of texture features, which become meaningful at high spatial resolution.

However, if only radiometry and standard textural features are used, together with a supervised classifier such as those presented, there is a strong overlap between some terrain classes, for example between *forest* and *orchard*, between *orchard* and *vineyard*, as well as a less significant overlap between *tilled field*, *orchard*, and *vineyard*, and between *tilled field* and *untilled field*. Due to their similar radiometric and textural characteristics, these classes can be distinguished only by precise orientation characteristics: forests and untilled fields are not oriented, tilled fields have one orientation, and orchards and vineyards have two. It is possible to distinguish between orchards and vineyards by the precise separation between rows of trees or vine stocks, which is determined by agricultural practices and local legislation.

It is therefore necessary to obtain accurate estimates of the periods of periodic textures, and to be able to distinguish between different kinds of textures, namely non-oriented, oriented along one direction, and oriented along two directions; in the one-directional case, several sub-cases can be distinguished depending on whether the texture is periodic or not along the main direction or the direction orthogonal to that.

However, existing orientation estimation methods are not well suited for this discrimination task. Some are based on Fourier analysis [Gar02, CRH02], which presents problems for arbitrarily-shaped regions; in Fourier-based methods, furthermore, texton shape translates into harmonics in the Fourier domain, which are treated as noise by these algorithms—in our case, each tree, vine stock, or plough mark is a texton. Others [DCGB02, ZXZ03, MGBdC04] use image gradients and are suited for statistic textures but not for texton-based ones. Chanussot, Bas, and Bombrun [CBB05] use a combination of Fourier analysis and Radon transform.

Warner and Steinmaus's method [WS05] is the one that most closely matches our requirements. They use the autocorrelation along the vertical, horizontal, and two diagonal directions in the neighbourhood of each pixel to distinguish between orchards, vineyards and other fields. If at least five peaks with a consistent spacing are found in at least one of the autocorrelograms for a pixel, the pixel is deemed to be an orchard or vineyard. Row spacing is computed as the

period of that autocorrelogram. A 95% classification accuracy is reached, but orientation and realistic periodicity information are not obtained.

In this chapter, I propose a variogram-based method, first presented in summarized form in [TS05b], to determine the main orientation of arbitrarily-shaped multi-channel images, including the period and other related features, inter-row spacing, and secondary orientation if present. These features can be used to discriminate between the problematic overlapping classes listed above. Experimental results show that 95.5% of test regions are correctly classified, that 82% of estimated directions are within 3° of their correct value, and that 81% of estimated periods are within 1 pixel of their correct value.

Figure 6.1 shows as a flowchart the part of the complete process described in this chapter.

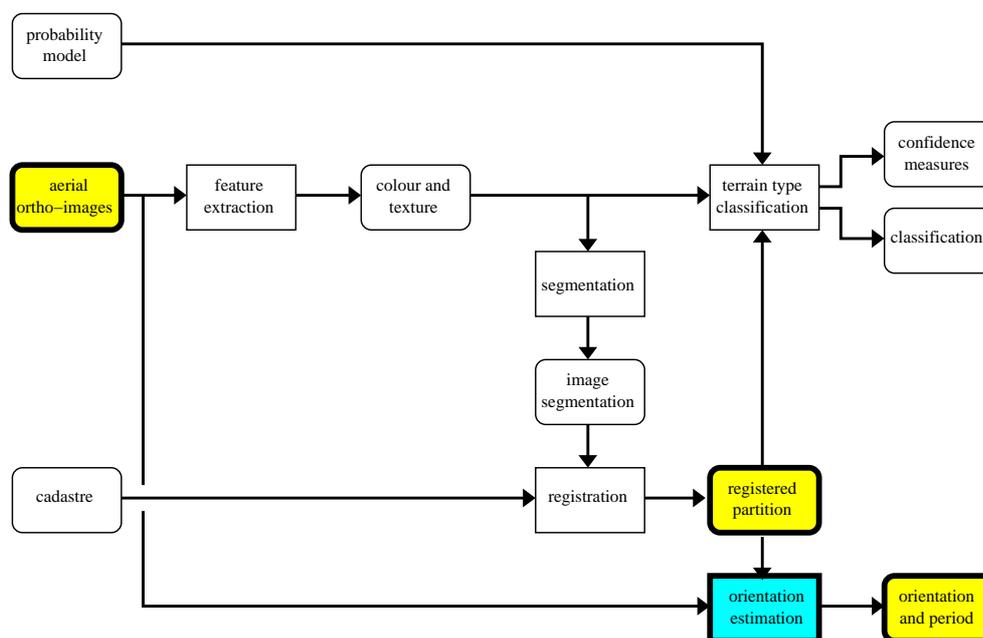


Figure 6.1: Flowchart depicting the processing steps (in blue rectangles) and intermediate results (in yellow rounded boxes) involved in the estimation of orientations and periods, in the context of the complete system.

6.2 Algorithm

This algorithm attempts to determine if the texture in a region of arbitrary shape—not necessarily rectangular—is not oriented, or is oriented along one or two main directions. If texture is oriented, texture periods (that is, characteristic distances between textons) are calculated. Several vegetation types can be distinguished with these features: forests (not oriented), untilled fields (not oriented), tilled fields (one direction), orchards (two directions, large spacing), and vineyards (two directions, small spacing). The actual orientations are also calculated and can be useful for some applications. Figure 6.2 shows an example of each vegetation type in an arbitrarily shaped region.

Figure 6.3 shows some sample images (first row), and the parameters that we wish to obtain in each case (second row): primary and secondary orientation (θ_1, θ_2), period along primary and secondary orientation (T_1, T_2) and inter-row spacing ($T_\perp = T_2 \sin |\theta_2 - \theta_1|$).



Figure 6.2: Sample images containing different vegetation types. From top to bottom, and left to right: forest, orchard; tilled field, untilled field; vineyard.

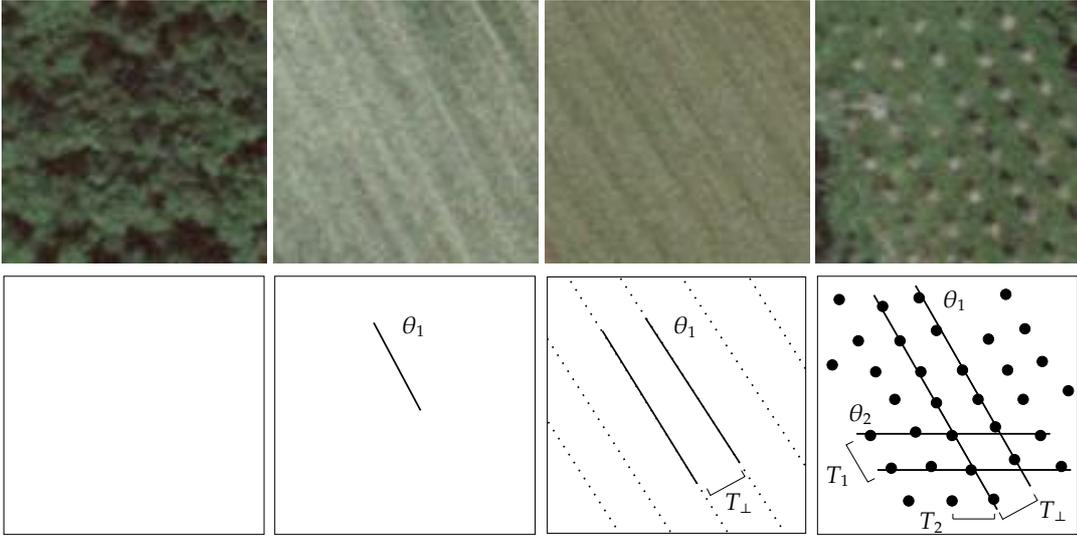


Figure 6.3: Sample regions (from left to right, a forest, two fields, an orchard), and the parameters that should be obtained for each region.

Processing follows these steps: First, the variogram of the image to be processed is calculated, and normalized (Sec. 6.2.1). The statistical variances along straight cuts of the variogram at various angles and distances to the centre are calculated and accumulated, and the image's main orientations are derived from them (Sec. 6.2.2). Then, a cut of the variogram is extracted along each detected orientation, and the directions orthogonal to them. If the cut shows a repeating pattern, its period is extracted (Sec. 6.2.3) using a novel watershed-based extrema detection method (Sec. 6.3). Finally, the orientations and the periods along these and the orthogonal directions are combined (Sec. 6.2.4).

6.2.1 Variogram

Let $I[z]$ be the n -channel image of the region to be processed. Let $S \subset \mathbb{Z}^2$ be the set of pixels for which I is defined, which need not be rectangular: $I : S \rightarrow \mathbb{R}^n$.

The variogram of I is defined as

$$D[\mathbf{u}] = \sqrt{\frac{\sum_{z \in S_{\mathbf{u}}} \|I[z] - I[z + \mathbf{u}]\|^2}{\text{card } S_{\mathbf{u}}}}, \quad (6.1)$$

(where $S_{\vec{u}} = \{\vec{z} \in S : \vec{z} + \vec{u} \in S\}$). For \vec{u} such that $S_{\vec{u}} = \emptyset$, $D[\vec{u}]$ in eq. 6.1 is not defined, and we use $D[\vec{u}] = \max_{\vec{w} : S_{\vec{w}} \neq \emptyset} D[\vec{w}]$ instead. This is similar to the autocorrelation, except for the use of the difference of pixel values instead of their product. In practice, we calculate $D[\mathbf{u}]$ only for $\mathbf{u} \in \{-32, \dots, 32\} \times \{-32, \dots, 32\}$.

The use of autocorrelation for periodicity detection is well entrenched in signal analysis circles because of its strong—and useful—links with the spectral power density of random signals. Some remote sensing algorithms attempting to detect crop orientation and tilling spacing use it [WS05] presumably for that reason. I believe that the variogram is a better choice: Pure Fourier analysis of such small texton-based textured images cannot handle arbitrarily-shaped regions, and is perturbed by the shape of the texton. Also, the product of digital numbers

—numerical values of pixels— used in autocorrelation analysis is not a good indicator of pixel similarity in this context, and is restricted to single-channel images; the variogram can use any reasonable —from a perceptual point of view— distance measure between multi-dimensional pixel values (I used the L^2 distance in eq. 6.1), thus allowing the processing of colour images.

A normalized variogram V is obtained by reversing and normalizing D to the range $[0, 1]$ while ignoring values near the origin, as

$$\begin{aligned} d_{\max} &= \max_{\|\mathbf{u}\| \geq 3} D[\mathbf{u}], \\ d_{\min} &= \min_{\|\mathbf{u}\| \geq 3} D[\mathbf{u}], \\ V[\mathbf{u}] &= \text{bound}_{[0,1]} \frac{d_{\max} - D[\mathbf{u}]}{d_{\max} - d_{\min}}, \end{aligned} \quad (6.2)$$

where $\text{bound}_{[a,b]} x$ is a if $x < a$, b if $x > b$, and x otherwise. Some sample variograms are shown in Fig. 6.4.

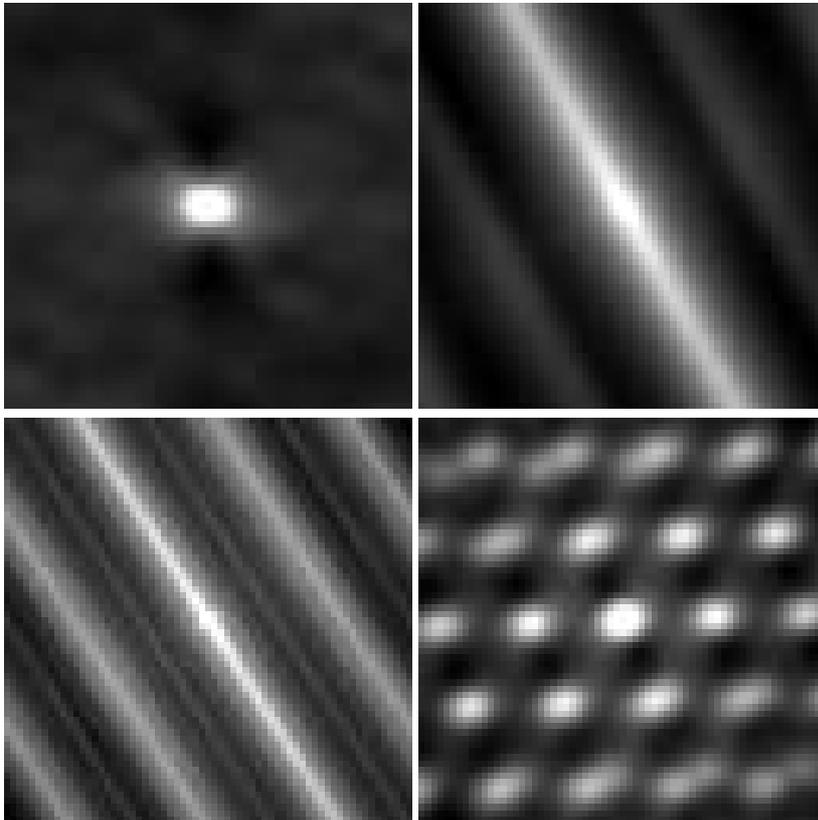


Figure 6.4: Normalized variograms V . Top-left, top-right, bottom-left, and bottom-right correspond to the first, second, third, and fourth columns of figure 6.3 respectively.

At this point we compute

$$\begin{aligned} e_{\text{in}} &= \max_{\|\mathbf{u}\| < 27} V[\mathbf{u}], \\ e_{\text{out}} &= \max_{\|\mathbf{u}\| \geq 27} V[\mathbf{u}]. \end{aligned} \quad (6.3)$$

If $e_{\text{in}} \cdot e_{\text{out}} < 0.45$, the region is classified as not having any directionality. This corresponds to variograms with higher values near the origin than farther away, like the first case of Figs. 6.3 and 6.4, as opposed to the other three cases where values far from the origin are comparable to those near it. If $e_{\text{in}} \cdot e_{\text{out}} \geq 0.45$, we continue with sections 6.2.2 and 6.2.3.

6.2.2 Direction estimation

Notice that the variogram reproduces the direction and inter-row spacing of the original image. In particular, when scanning the variogram along a straight line, if the statistical variance of the values found along that line is minimal then the line is oriented in a significant direction of the image (but in some cases, such as the bottom-right case in Fig. 6.4, the converse is not true). We use this property to estimate the texture orientations (the θ_1 and θ_2 angles in Fig. 6.3). Let

$$P(\alpha, r) = \text{var}_i V(\mathbf{z}_0 + t\mathbf{v}), \quad (6.4)$$

where $\mathbf{z}_0 = (r \sin \alpha, -r \cos \alpha)$, $\mathbf{v} = (\cos \alpha, \sin \alpha)$, and where $V(\mathbf{z})$ is an interpolation for $\mathbf{z} \in \mathbb{R}^2$ of the discrete-domain $V[\mathbf{u}]$. That is, $P(\alpha, r)$ is the statistical variance of the values found when V is scanned along a line of angle α and distance to the origin r . We then accumulate the values of P for all r at a given angle, and invert and smooth the result by convolution ($*$) by a Gaussian kernel of $\sigma = 4/3$,

$$A(\alpha) = -\mathcal{N}_{4/3}(\alpha) * \int_r P(\alpha, r) dr. \quad (6.5)$$

I use notation corresponding to continuous domains in order to simplify the exposition, but that all calculations are actually performed on a sufficiently fine discrete domain.

Figure 6.5 shows $P(\alpha, r)$ and $A(\alpha)$ for some sample images. Note that main texture directions have a corresponding maximum in $A(\alpha)$. Note also that undirected textures, such as the first case, also have a maximum in $A(\alpha)$, because of normalization issues, and also because of a residual orientation due to the direction of incident sunlight. However, with the test of equation (6.3) we are able to detect undirected textures before this direction estimation step.

Then, the local extrema of $A(\alpha)$ for $\alpha \in [0, \pi)$ are located. Let a_i be the value of α for the i -th extremum in $A(\alpha)$. The *strength* of the i -th extremum is defined as

$$s_{A,i} = \frac{|A(a_i) - A(a_{i-1})| + |A(a_i) - A(a_{i+1})|}{2}, \quad (6.6)$$

with suitable wrap-around for the first and last extrema. Only maxima with strength $s_{A,i} > 0.001$ are considered, in order to remove non-significant maxima. The maximum with highest strength is taken as the primary orientation θ_1 , and that with the second-highest as the secondary orientation θ_2 . It may be that only one orientation is found, or even none if no maxima has a high enough strength.

6.2.3 Periodicity estimation

For each of the directions θ_1 , $\theta_{1\perp} := \theta_1 + \frac{\pi}{2}$, θ_2 , and $\theta_{2\perp} := \theta_2 + \frac{\pi}{2}$, the period, if there is one, of the texture along that direction is estimated. This section describes how to estimate the period along an arbitrary direction θ .

First, a cut of the variogram V is obtained in the direction θ . The cut is then smoothed by convolution by a Gaussian kernel of $\sigma = 1$, giving C :

$$C(r) = V(r \cos \theta, r \sin \theta) * \mathcal{N}_1(r). \quad (6.7)$$

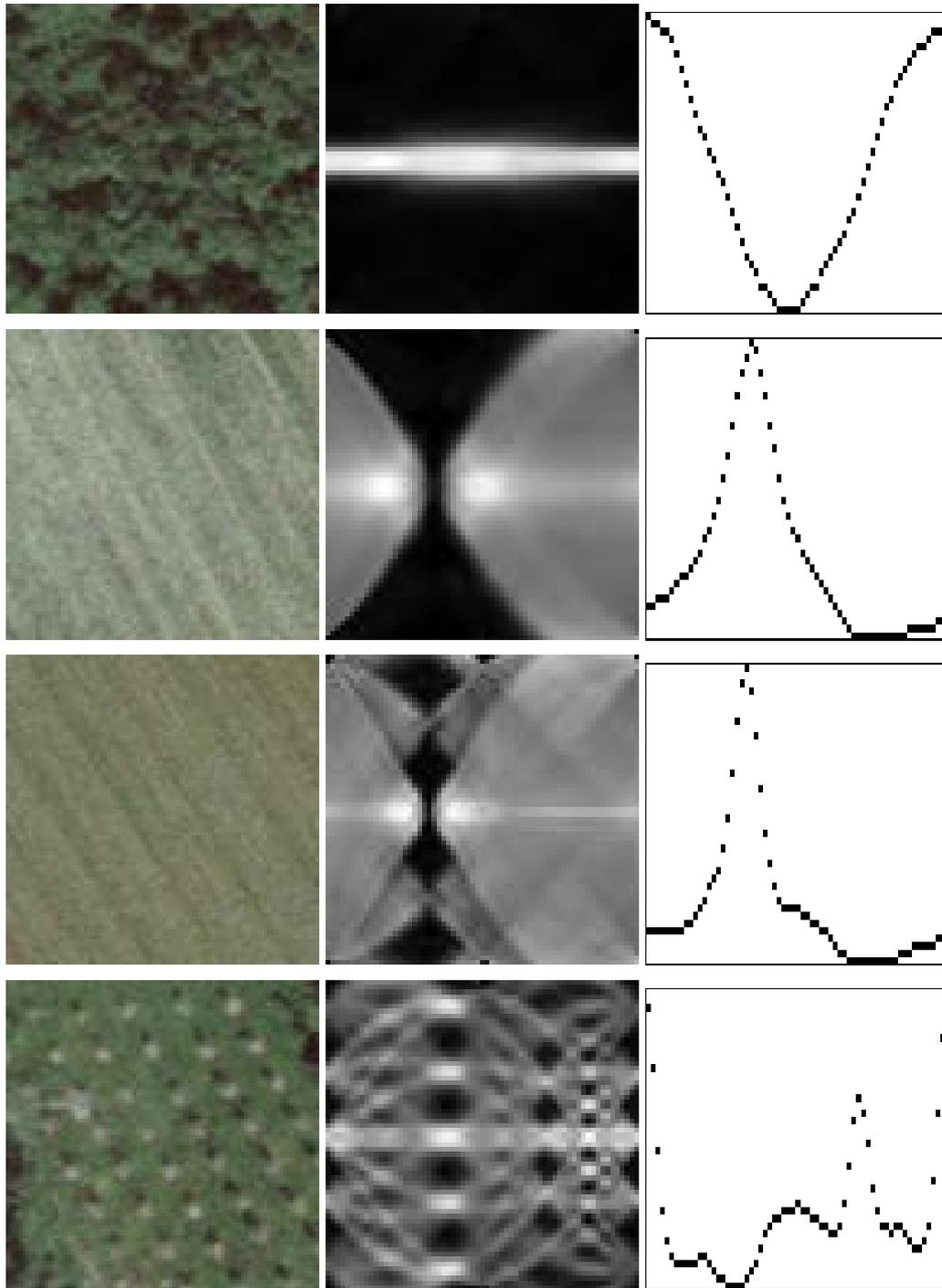


Figure 6.5: $P(\alpha, r)$ and $A(\alpha)$ for some sample images. Left: original images (the same as in Fig. 6.3); center: $P(\alpha, r)$, angles $\alpha \in [0, \pi)$ are in the horizontal axis and radii r in the vertical axis; right: $A(\alpha)$, vertical axes not at the same scale.

Then, the local extrema of the smoothed cut $C(r)$ are located. Because $C(r)$ is actually discrete, and also noisy with many spurious extrema that we do not wish to detect, local extrema are not detected by examining whether the value of C at one point is higher or lower than that at neighbouring positions, but by using a novel watershed-based method, which I describe in detail in section 6.3. Briefly, basins are filled using the classically-located minima as seeds; the same is done for maxima; extrema which are close —both in the values of r and of $C(r)$ — are merged; those which thus become non-extremal points are removed.

Let c_i be the value of r for the i -th extremum in $C(r)$. The *strength* of the i -th extremum is defined as

$$s_{C,i} = \frac{|C(c_i) - C(c_{i-1})| + |C(c_i) - C(c_{i+1})|}{2}. \quad (6.8)$$

We then compute

$$t_c = \frac{\max_{\|r\| \geq 3} C(r) - \min_{\|r\| \geq 3} C(r)}{4} \quad (6.9)$$

and discard extrema with strength $s_{C,i} < t_c$. Let N be the number of remaining extrema, \hat{c}_j the value of r for the j -th remaining extremum in $C(r)$, and $\hat{s}_{C,j}$ its strength. Ideally, the distance between two adjacent extrema is half a period. We calculate the period T along θ as a weighted average of these distances, and also calculate its strength K , as

$$a = \sum_{j=1}^{N-1} (\hat{c}_{j+1} - \hat{c}_j) \cdot \sqrt{\hat{s}_{C,j+1} \cdot \hat{s}_{C,j}}, \quad (6.10)$$

$$K = \sum_{j=1}^{N-1} \sqrt{\hat{s}_{C,j+1} \cdot \hat{s}_{C,j}}, \quad (6.11)$$

$$T = 2a/K, \quad (6.12)$$

If $K < 0.65$ (the peaks in the cut do not seem very significant, so there is probably not a real periodic pattern) or $T > 20$ (the period is too large to be computed reliably), it is decided that the image has no periodicity along θ .

Because we calculate the period by averaging between many inter-extrema distances (in eq. 6.10), and weighting by the extremum strength, we obtain a higher resolution than what would be possible using a Fourier transform on the cut C or on the variogram V , especially for large periods. This is because, for a Fourier-based analysis, in practice, we would be using a discrete Fourier transform (or, more precisely, the FFT); the FFT of a 64×64 image (the variogram is computed for a 65×65 square) has a constant frequency resolution of $\pi/32$ rad, but the period resolution is not constant and is very coarse for small frequencies. For example, the largest four periods that can be obtained from that FFT are 12.8, 16, 21.3 and 32 pixels; intermediate values cannot be obtained using the FFT.

By this procedure we obtain, for the directions θ_1 , $\theta_{1\perp}$, θ_2 , and $\theta_{2\perp}$ the periods T_1 , $T_{1\perp}$, T_2 , and $T_{2\perp}$, respectively. It may be that in section 6.2.2 less than two directions were found, or that, for some directions, no periodicity can be found.

6.2.4 Region classification

Depending on how many directions are found for a region, and on which of these directions (and their perpendicular directions) exhibit a periodicity, the region is classified in different ways.

If no direction is found, the region is classified as *undirected*. Otherwise,

1. If periodicities are found neither along the main direction θ_1 nor its perpendicular $\theta_{1\perp}$, the region is classified as *linear*, oriented along θ_1 with no periodicities.
2. If periodicities are found along $\theta_{1\perp}$ but not θ_1 , the region is classified as *linear*, oriented along θ_1 with inter-row spacing $T_{1\perp}$.
3. If periodicities are found along θ_1 but not $\theta_{1\perp}$, and either only one direction is found, or the second direction has no periodicity along θ_2 , the region is classified as *linear*, oriented along θ_1 with along-row periodicity T_1 .
4. If only one direction is found, with periodicities along θ_1 and $\theta_{1\perp}$, the region is classified as *linear* with inter-row spacing $T_{1\perp}$ and along-row periodicity T_1 .
5. If periodicities are found along θ_1 and $\theta_{1\perp}$, and a second direction is detected but no periodicity is found along θ_2 , the region is classified as *bidirectional* with main orientation θ_1 , second orientation $\theta_1 + \frac{\pi}{2}$, inter-row spacing $T_{1\perp}$ and along-row periodicity T_1 .
6. Finally, if at least two directions are found, with periodicities along θ_1 and θ_2 , the region is classified as *bidirectional* with main orientation θ_1 , second orientation θ_2 , period T_1 along θ_1 , period T_2 along θ_2 , and inter-row spacing $T_{\perp} = T_2 \sin|\theta_2 - \theta_1|$.

6.3 Watershed-based extrema detection

In section 6.2.3 I describe a procedure in which it is necessary to obtain the local extrema of a certain function $C(r)$. However, because this function is discrete (both r and $C(r)$ take discrete values) and very noisy, and we want to ignore extrema caused by noise, and obtain instead the extrema present in an ideal, continuous, noise-free function from which $C(r)$ would be derived, we cannot simply examine whether the value of $C(r)$ at one point is higher or lower than at neighbouring positions. Noise may produce many spurious local extrema. It can also, for example, transform a local maximum into two local maxima at different positions with a local minimum in between, by lowering the value of $C(r)$ at the original maximum. Finally, the discrete nature of $C(r)$ makes some extrema be spread among several positions (values of r). Figure 6.6 shows an example of such a function.

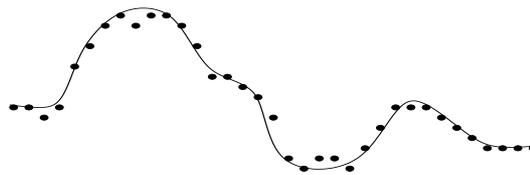


Figure 6.6: Example of noisy function $C(r)$ (shown as thick dots). The ideal continuous, noise-free function from which $C(r)$ is derived, is shown as a continuous curve.

This section describes an algorithm that I developed to obtain the local extrema in a noisy, discrete function $v[x]$, with $x \in \{1, \dots, N\}$.

Let $A = \{a_1, a_2, \dots, a_v\}$ be the sequence of the values taken by $v[x]$ sorted in increasing order: $a_i < a_{i+1}$, and for all x , $v[x] \in A$. Step by step, we will define two sets of *labellings* for v , $L_{\min}[x]$ and $L_{\max}[x]$, and two *sets of labels*, R_{\min} and R_{\max} , initially empty, as follows:

1. Set $\ell := 1$.

2. For i from 1 to V ,
 - For x from 1 to N ,
 - If $v[x] = a_i$ then
 1. If $x > 1$ and $L_{\min}[x - 1]$ has already been defined, define $L_{\min}[x] := L_{\min}[x - 1]$,
 2. else, if $x < N$ and $L_{\min}[x + 1]$ has already been defined, define $L_{\min}[x] := L_{\min}[x + 1]$,
 3. else, define $L_{\min}[x] := \ell$, add x to the set R_{\min} , and then increase ℓ by one.
3. For i from V down to 1,
 - For x from 1 to N ,
 - If $v[x] = a_i$ then
 1. If $x > 1$ and $L_{\max}[x - 1]$ has already been defined, define $L_{\max}[x] := L_{\max}[x - 1]$,
 2. else, if $x < N$ and $L_{\max}[x + 1]$ has already been defined, define $L_{\max}[x] := L_{\max}[x + 1]$,
 3. else, define $L_{\max}[x] := \ell$, add x to the set R_{\max} , and then increase ℓ by one.

Figure 6.7 shows, underneath the function $C(r)$, the values of $L_{\min}[x]$. The positions r belonging to the set R_{\min} are marked with an asterisk. Above the function $C(r)$ the values of $L_{\max}[x]$ are shown, using letters starting from "A" instead of numerical values of $\ell \geq 10$. Positions belonging to R_{\max} are also marked with an asterisk.

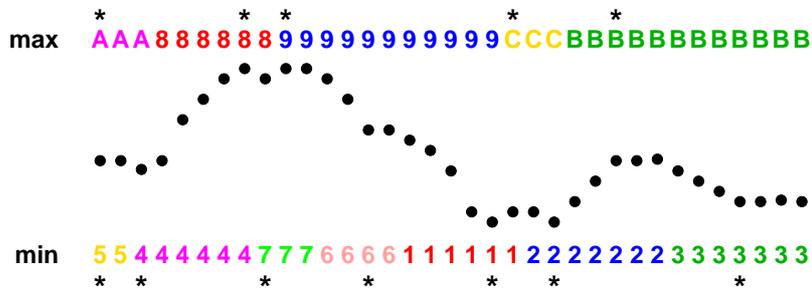


Figure 6.7: Function $C(r)$ with values of L_{\min} (under the function), the set R_{\min} (under the function, marked with asterisks), values of L_{\max} (over the function), and the set R_{\max} (over the function, marked with asterisks).

At this point, R_{\min} and R_{\max} are tentative sets of the positions (values of x) of local minima and maxima in $v[x]$. These two sets of positions are joined, removing positions that correspond to both a maximum and a minimum, to obtain the set R ,

$$R = (R_{\min} \cup R_{\max}) \setminus (R_{\min} \cap R_{\max}), \tag{6.13}$$

and the first and last extrema are removed from R :

$$R' = R \setminus \{\min R, \max R\}. \tag{6.14}$$

Next, the extrema of $v[x]$, retained after equation (6.14), that are close to each other (both in the value of v at the extremum, and in the position x) are joined together, disregarding whether each joined extremum is a maximum or a minimum —mixed-type extrema may be merged— by the following procedure, which requires a threshold for the value of v , T_v , and one for the value of x , T_x . Let's write the set of retained extrema as a set E of *pairs*, the first component being the position x and the second the value of $v[x]$ at the extremum:

$$E = \{(x, v[x]) : x \in R'\}. \quad (6.15)$$

We define two extrema $a, b \in E$ to be *close*, $a \sim b$, if

$$(x_a, v_a) \sim (x_b, v_b) \iff |x_a - x_b| \leq T_x \text{ and } |v_a - v_b| \leq T_v, \quad (6.16)$$

and partition the set E into subsets E_1, \dots, E_p (mutually disjoint, and with $E = \cup_{i=1}^p E_i$) which are maximal by " \sim ", that is, for each element e in a subset E_i ,

$$\exists m \in E_i \setminus \{e\} : e \sim m \quad (6.17)$$

$$\forall u \in E \setminus E_i : \neg(e \sim u). \quad (6.18)$$

Note that \sim is not transitive, so it is not true that $\forall m \in E_i \setminus \{e\} : e \sim m$.

Then, each subset of extrema E_i is replaced by a single "extremum" w_i at the mean position of its elements,

$$\begin{aligned} E_i &= \{e_{i1}, e_{i2}, \dots, e_{in_i}\}, \\ e_{ik} &= (x_{ik}, v_{ik}), \\ w_i &= \left(\frac{1}{n_i} \sum_{k=1}^{n_i} x_{ik}, \frac{1}{n_i} \sum_{k=1}^{n_i} v_{ik} \right). \end{aligned} \quad (6.19)$$

Finally, we need to classify the extrema w_1, \dots, w_p into *local maxima*, *local minima*, and *non-extremal points*. Let's assume $p \geq 2$ (otherwise we have a single extremum, from which we cannot calculate a period with equation (6.10)). If the extrema $w_i = (x_i, v_i)$ are sorted such that $x_i < x_{i+1}$, then

1. if $v_1 < v_2$, then w_1 is a local minimum;
2. if $v_1 > v_2$, then w_1 is a local maximum;
3. if $v_p < v_{p-1}$, then w_p is a local minimum;
4. if $v_p > v_{p-1}$, then w_p is a local maximum;
5. if $1 < i < p$, and $v_{i-1} > v_i$ and $v_i < v_{i+1}$, then w_i is a local minimum;
6. if $1 < i < p$, and $v_{i-1} < v_i$ and $v_i > v_{i+1}$, then w_i is a local maximum;
7. otherwise, w_p is classified as a non-extremum.

6.4 Evaluation

I have run this algorithm on 67 test regions of varied shapes, containing different kinds of vegetation, taken from aerial imagery at 50 cm resolution with red, green, and blue channels. Each test region was manually labelled as *forest*, *planted forest* (non-fruit trees laid out in a grid),

orchard, *untilled*, *tilled*, *weakly tilled* (plow marks barely visible), and *wide tilled* (visible parallel marks too widely spaced to be plow marks). In addition, I determined reference values for each region's primary and secondary orientations and periods, where applicable.

There were no vineyards in the test image. Vineyards should be detected as *bidirectional* regions, and they can be distinguished from orchards by their inter-row spacing (or period) T_{\perp} , which should be smaller than that of orchards.

These "application" classes do not correspond to the "algorithm" classes given by the algorithm: *undirected*, *linear* with or without along-row and inter-row spacing, and *bidirectional*. A classification into these algorithm classes is, however, sufficient to disambiguate among the easily confused application classes given in the introduction: *forest* and *orchard* (undirected vs. bidirectional); *orchard* and *vineyard* (different inter-row periods T_{\perp}); and less importantly *tilled field*, *orchard*, and *vineyard* (T_1 not detected for tilled fields, orchards and vineyards have different T_1); and *tilled* and *untilled* (T_{\perp} detected for tilled fields only).

Table 6.4 shows the number of regions for each combination of ground truth class (row) and detected class (column). Note that the classes given by this algorithm (undirected, bidirectional, and four kinds of linear texture) are not the same as the terrain classes for our problem (*forest*, *orchard*, *tilled field*, ...). However, the goal of the presented algorithm was not to classify images into these latter classes, but to disambiguate between some often confused classes—as classified previously by classical methods. In this sense, combinations of output and ground truth that are classification errors—that is, that would not disambiguate these easily confused classes—are shown with a grey background. T_1 and T_{\perp} indicate whether a periodicity has been detected for the θ_1 and $\theta_{1\perp}$ orientations, respectively. Of 67 regions, 64, or 95.5%, are correctly classified.

	undir.	linear	linear (T_1)	linear (T_{\perp})	linear (T_1, T_{\perp})	bidir.
forest	7					
planted f.					1	1
orchard			1			9
untilled	1		1			
tilled		4		6		
weak till	2	19	1	7		1
wide till		6				

Table 6.1: Number of regions for each combination of ground truth class (row) and detected class (column).

The differences between the detected values and the reference for the primary and secondary orientations, the period along the primary orientation, and the inter-row period, were computed for each region. Figures 6.8 and 6.9 show the histograms of these differences. Only regions for which data was available both in the reference and the detection were considered; for example, a region's secondary orientation is only considered if the region is detected as *bidirectional* and its ground truth class is *orchard* or *planted forest*.

In 85% of regions, the detected primary orientation is within 3° of its reference value. In 81% of regions, the detected primary and inter-row periods are within 1 pixel of their reference values. Secondary orientation is detected with a much poorer precision (60% of regions within 3°) but this does not affect the precision of the inter-row period.

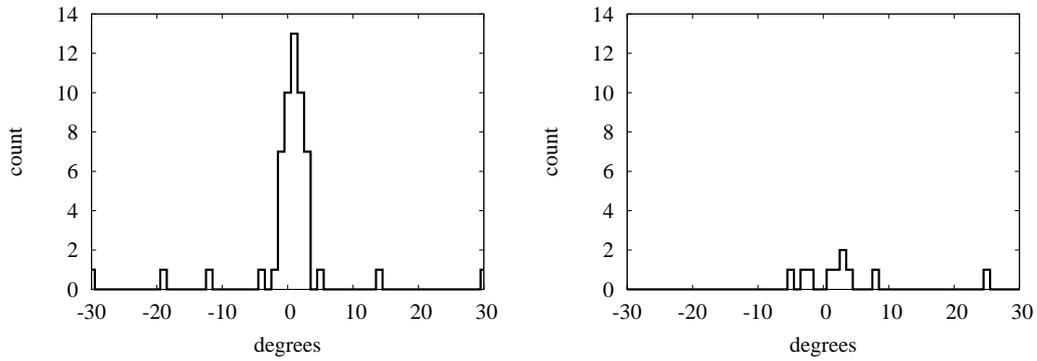


Figure 6.8: Histograms of differences between detected and reference values. Left: primary orientation θ_1 (in degrees); right: secondary orientation θ_2 (in degrees).

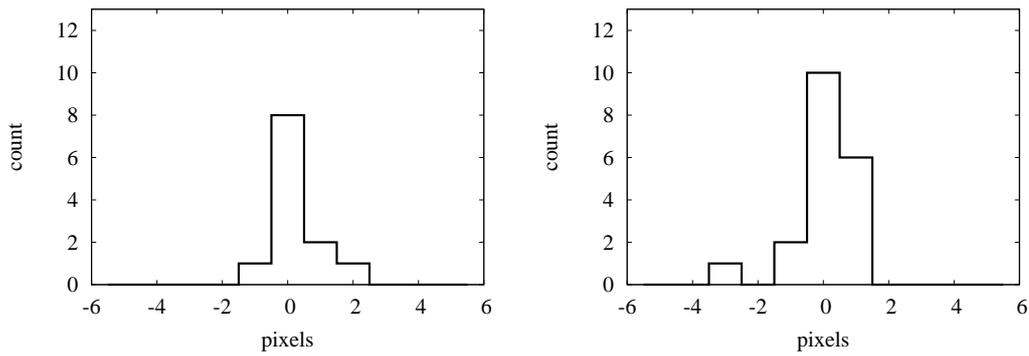


Figure 6.9: Histograms of differences between detected and reference values. Left: primary period T_1 (in pixels); right: inter-row spacing T_\perp (in pixels).

6.5 Conclusion

In order to characterize fields and to discriminate between forests, orchards, tilled fields, untilled fields, and vineyards, which can in some cases be confused if only radiometry and standard textural features are used for classification, I have presented a method for determining the primary orientation—and, eventually, the secondary orientation—of an arbitrarily-shaped vegetation region, and for computing the inter-row and along-row spacing if there is one.

The algorithm, based on a novel way of extracting direction and period information from a variogram, correctly classifies 95.5% of a set of test regions, gives orientations which are within 3° of their reference values in 82% of cases, and gives inter-row and along-row spacings which are within 1 pixel of their reference values in 81% of cases.

I have not had the time to integrate this post-classification module into the main processing chain. Therefore, evaluation results from the chapter on classification, chapter 5, do not take into account this further disambiguation. Also, the accuracy figures provided in this chapter do not take into account errors that may have been made by the algorithms in chapter 5. I propose the following integration strategy: A classification, as described in chapter 5, should be run for the relevant image regions. For those regions belonging to one of the easily confused classes given in the introduction to this chapter, this orientation estimation algorithm should then be run. There are several possible outcomes for a region: Both algorithms may agree, such as when a region is classified as *orchard* by the classifier and as *bidirectional* by the orientation estimator. The algorithms may give conflicting answers which are to be expected taking into account that some classes are easily confused; for example, a region may be classified as *orchard* by the classifier and as *undirected* by the orientation estimator, which suggests that the classifier made the typical mistake of confusing a *forest* with an *orchard*, and that the region was actually a *forest*. Finally, the algorithms may give incompatible answers, such as a region being classified as *vineyard* by the classifier and as *undirected* by the orientation estimator. In the latter case, we may choose to keep the *vineyard* classification but significantly lower the region's classification confidence value.

An important question remains: if the orientation estimator can be used to distinguish between these easily confused classes, why not run it for every region, before the classifiers of chapter 5, and use its results as an additional input to these classifiers? Although this is an interesting idea which might be tested in further research, there are two main problems. First, the orientation estimation algorithm is too slow to run it on every region, in particular when a registered cadastre is not available and it is the small regions given by a fine segmentation that are classified. Since a cadastre was not available for most of our most interesting test site (*Toulouse*) I could not have tested such a setup. The second reason is that adding input channels to the classifier increases the dimensionality of the models and the problem, thus making the classification phase itself much slower.

7 | Conclusion

This document presents the design of a complete system able to locate vegetation in high-resolution aerial images of rural areas, and to classify this vegetation into terrain classes relevant to cartography. In addition to aerial images of high spatial resolution and low spectral resolution, this system uses cadastre data to obtain hints of possible field locations; if the system is used to bring up to date an old classification, when new images become available, the old classification can be used instead of the cadastre, but this operating mode has not been tested in the thesis.

The goal of this thesis is to obtain methods to solve a practical, real-world problem, but this required developing new algorithms and mathematical frameworks. Therefore, theory and application are tightly interwoven in this document, giving it a particular structure. Certainly, a purely theoretical thesis, or one in which pre-existing algorithms were combined in the right way to obtain the desired results, would have looked very different.

In this section I list the contributions I have presented for each of these steps, as well as their quantitative performances. Later, I will give some perspectives on future research and closing remarks.

7.1 Land cover analysis

With the advent of affordable high-quality, high-resolution aerial and satellite imagery, and ever-increasing computational power, almost-fully-automatic map generation from images appears increasingly feasible in a half-decade time frame.

Current land cover analysis systems involve little semantic knowledge, and take classification decisions based mostly on local information only. This thesis pushes the limits of such systems by using specific techniques —registration, per-region classification, nested class models— made possible by the particular characteristics of the input data, which make better use of context to give improved performance. However, classification decisions are taken on the basis of short-range context only. Furthermore, only image contents are reported; for example, with such a system —adapted to road detection—, a portion of road obscured by trees will be classified —as it should— as *trees*, not *road*.

My opinion is that a complete image interpretation system for map generation will require, after an image-based classification such as the one described in this thesis, a further step involving higher-level terrain models, with concepts such as *forests*, *towns*, *cities*, *suburbs*, *roads* or *hills*.

Such high-level objects will be initialised from image classification results, but further refined using such “expert” knowledge as the fact that nearby towns tend to be linked by roads, that rivers follow valleys, and that roads do not have discontinuities —bridges or tunnels, conveniently, may be inferred from elevation data, also available. By inferring information not visible in the image, this semantically-rich analysis would fill the gap between images and maps.

7.2 Contributions

The system is an image processing chain with five steps. First, transformed colour channels and texture features are extracted from the input images. These colour transformation and texture descriptors are taken from the literature, and the only contributions I made in this area are some minor adaptations to improve their usability in this application.

Second, using these derived channels in addition to the original radiometric channels, input images are segmented using a multi-scale segmentation algorithm, in order to obtain a partition of the image into small regions, each containing only one terrain type, separated by edges whose strength, or contrast, is calculated.

For this step, I developed a novel method for evaluating the quality of a multi-scale segmentation—in the literature I could only find methods for evaluating single-scale segmentations. I also presented a modification to Guigues’ energies to take into account the similarity between segmentation edges and cadastre edges, using anisotropic notions of length and angle, but these give poor results. Finally, I presented an adaptation of the Stepwise Forward Selection optimization method. Using this modified SFS method and the multi-scale segmentation quality measures, I obtain an optimal parameter set for segmentation with quality $M = 0.3490, F = 0.1848$, compared to $M = 0.3722, F = 0.2569$ when using the original radiometry channels (these quality measures, defined in section 3.3.1, measure the amount of missed detections and false detections for a multi-scale segmentation, and lower values correspond to better segmentations). This evaluation is interesting not only in the context of this thesis, but also in the wider context of high-resolution remote sensing in rural areas.

Third, the cadastre—or an old classification, if the system is used in update mode—is registered onto the segmentation edges obtained in the previous step. In this way, larger regions are obtained, whose limits—unlike those of the original cadastre or old classification—follow salient image boundaries.

In this step, I presented two graph-to-image registration algorithms, one of which is region-based and the other edge-based, with different quantitative and qualitative performances. Using these algorithms to register a cadastre, the distance from cadastre to ground truth is reduced by 32.2%. For especially bad initial conditions, the improvement goes up to 43.7%.

Next, each region obtained in the previous step is classified using one of the several per-region classification algorithms. These algorithms give, for each region, a single most-probable terrain type, and the confidence which can be given to this decision. Because these classification algorithms are supervised, a training phase has to be performed beforehand, using training data.

In this step, I presented several “flat” classification algorithms which operate region-by-region, using standard probability models. In addition, I presented a novel “nested” probability model that, in some cases, can describe input data better than standard models. I also present new classification and estimation algorithms that use these novel nested models. Both flat and nested algorithms give, in addition to a classification, a confidence map indicating the reliability

of their classification for each region, which can be used for semi-automatic classification. For a semi-automatic classification with 75% coverage—ignoring the 25% of pixels with the worst confidence values—a classification accuracy of 99.9% ($\kappa = 0.996$) is obtained for the simpler Saint-Léger test site, and 94.7% ($\kappa = 0.888$) for the more complex Toulouse test site.

Finally, some regions can be post-processed using an orientation and period estimator, in order to remove some common classification ambiguities that appear in the previous step.

This estimator is also a contribution made in this thesis, to solve the drawbacks of more traditional Fourier-based orientation and period estimators. Using this estimator to resolve these classification ambiguities, 95.5% of regions in a test set are correctly classified, 82% of estimated directions are within 3° of their correct value, and 81% of estimated periods are within 1 pixel of their correct value.

These are, in summarized form, the major contributions that I have presented in this thesis:

1. Evaluation method for multi-scale segmentations.
2. Results of extensive evaluation of image segmentations.
3. Region-based graph-to-image registration algorithm.
4. Edge-based graph-to-image registration algorithm.
5. Per-region flat classification algorithms.
6. Nested probability models for classification.
7. Per-region nested classification algorithms.
8. Per-region nested model estimation algorithms.
9. Orientation and period estimator.

7.3 Perspectives for future research

A thesis, like all research, is never a finished work. Because of time and resource constraints, perfect results cannot be obtained, and we often have to settle for good-enough ones. Furthermore, many interesting ideas, found during the development of the thesis but not directly related to it, cannot be properly explored.

Some developments could simply not be included in the limited time available for this thesis. They have little research value and the thesis results can stand on their own without them. However, if extra time is available, they could be studied. The first such development is the integration of the orientation and period estimator with the rest of the processing chain; this is purely a simple matter of programming with no research interest, but would be interesting for the practical use of this system.

In addition, the system should be tested in update mode, that is, using an old classification instead of a cadastre graph; this could not be done during the thesis for lack of data. Although I designed the system with this operation mode in mind, and I expect it to function correctly in this situation, it should be actually tested.

Furthermore, the ideas of section 4.6, or perhaps others, should be used to implement a method to detect class-heterogeneous regions, and to partition them into smaller class-homogeneous parts. This should allow a larger fraction of regions to be correctly classified, and therefore to obtain better classification accuracies for a fixed classification coverage ratio.

Finally, a global evaluation of this system should be done, using for each step the best parameter sets. In this thesis, each step has been presented separately, and the input to each step was often not the best possible output of the previous step, but one obtained with a reasonable, but probably sub-optimal, parameter set. Global classification accuracies for the whole system running in optimal conditions should be obtained.

Other, more interesting, research ideas were found during the development of this thesis, but were not explored because their scope was too large.

The first one is about the use of shape for segmentation. The fit-to-cadastre anisotropic variant to Guigues' energies, proposed in this thesis, does not seem to work—and the reason why, as yet unclear, should be found. Furthermore, shape seems to be an obvious segmentation criterion for the kind of objects we are dealing with—mostly rectangular fields—so more research on how to incorporate shape in segmentation is in order. Shape could also be used for classification, since straight edges and right angles are much more typical of fields than of forests, lakes, or rivers.

We have seen that some classification algorithms perform better for some terrain classes than for others. This leads naturally to the suggestion of using data fusion techniques for merging the results of several classifiers in order to obtain better results.

Another way of improving classification accuracy is the use of additional external information. At IGN, for example, digital elevation or terrain models, which give the height of each pixel in an image, are easily available. There is already ongoing research by Arnaud Le Bris on the use of height maps, and slope and orientation data derived from them, to obtain prior probabilities for different classes; using flat per-region classification algorithms developed in this thesis he obtains very good results for classification of high-mountain terrain into rocks, glaciers, snow, forest, and others.

Finally, what I believe to be the most interesting research perspective after this thesis is the use of higher-level contextual information for classification. The system should construct semantic models of the terrain area under study. These models should be of a higher level than that given by the terrain class for each region, and include concepts such as *towns*, *roads* linking towns, *rivers* (not just regions of *river* terrain type), *forests* (connex regions of forest type), and so on, with appropriate uncertainty values, and use them to correct local classification errors. For this we could use Markovian or other probability models [Rab89, LBMN00], or explicit spatial grammars [WSK03, AKT⁺05], constructed either by hand, or learning from annotated data. Ideas from the language processing community [Bon95, GLL99] may be useful. Iterative methods, where the results of higher-level analysis are themselves reused to refine the low-level classification, should be investigated.

7.4 Concluding remarks

Traditional land cover classification algorithms are not accurate enough for fully-automatic production of cartographic databases. The data available to us, with very low spectral resolution and very high spatial resolution, is expected to give even worse results than those typically obtained for traditional land cover classification algorithms, which often use higher spectral resolution images.

At high spatial resolution, however, we can compensate for the lack of spectral resolution by using texture indicators. Also, per-region classification becomes meaningful. We have shown that, by combining both, and by using a semi-automatic classification approach instead of a fully-automatic one, we obtain very good land cover classification accuracies in the context of vegetation detection in rural areas.

pedalling through
the dark currents
I find an accurate copy
a blueprint
of the pleasure in me
[...]
five fingers
they form a pattern
yet to be matched

—Björk, "Pagan Poetry"

A | Texture descriptors and transformed colour spaces

This appendix describes the transformed colour spaces and texture descriptors used for the segmentation of chapter 3 and the classification of chapter 5.

A.1 Colour spaces

A.1.1 Red-green-blue (RGB) space

Together with the near-infrared channel, this is the original colour space as given by the acquisition digital camera. Being untransformed, it has the advantage of having a higher precision; however, it has been long known that the RGB colour distance does not correspond to subjective colour distance. Let's describe a colour in this space as a vector $c = (r, g, b) \in [0, 1]^3$.

In section 3.4, the red, green, and blue channels are designated as "red", "green", and "blue" respectively. The near-infrared channel is designated as "ired".

Figure A.1 shows these channels for a portion of a test site.

A.1.2 Simple haze removal

A common simple haze removal technique is to subtract from each radiometry channel its minimum value across an image or test site. If $I_i(x)$ is the image value for channel i at position x , the new channels i' are given by

$$I_{i'}(x) = I_i(x) - \alpha_i \tag{A.1}$$

$$\alpha_i = \min_x I_i(x). \tag{A.2}$$

This is applied to the raw red, green, blue, and near-infrared channels.

The haze-removed channels corresponding to the red, green, blue, and near-infrared channels are designated in section 3.4 as "red-z", "green-z", "blue-z", and "ired-z" respectively.

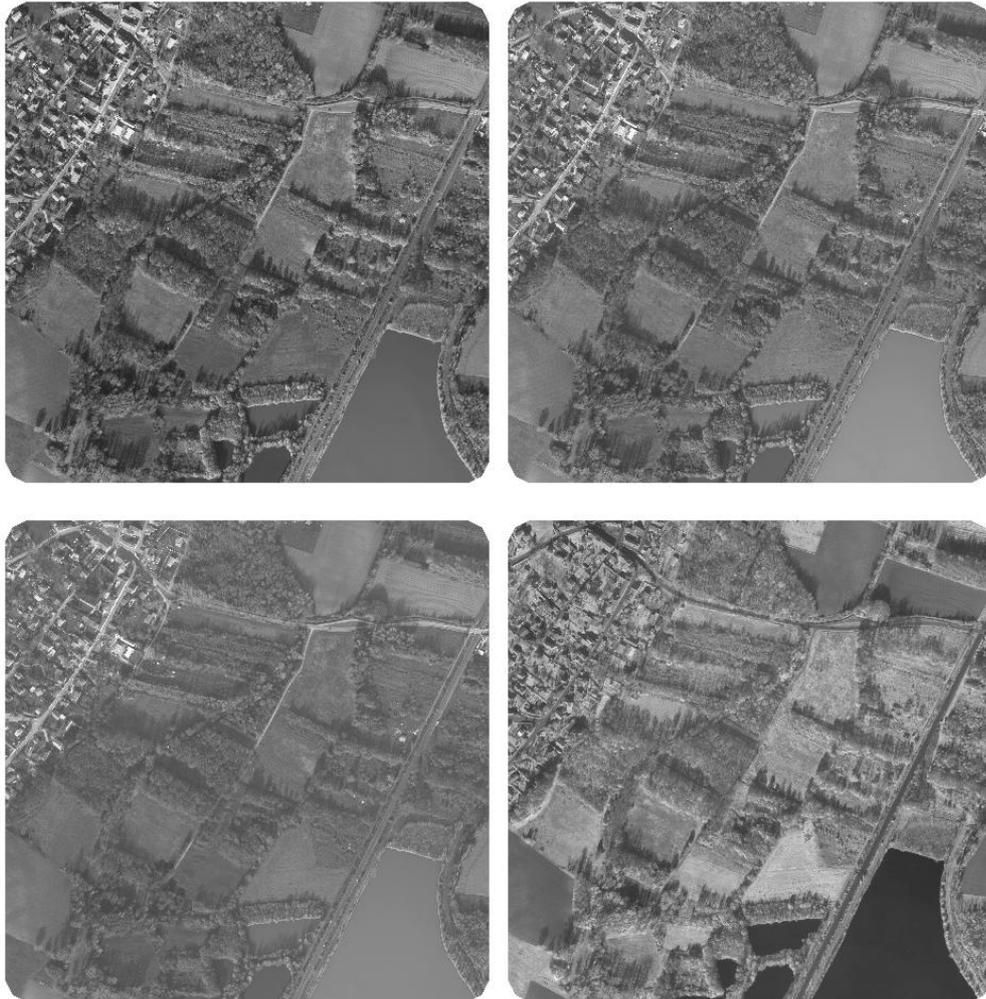


Figure A.1: Red (top left), green (top right), blue (bottom left), and near-infrared (bottom right) channels for the G003002 section of the Toulouse data set.

A.1.3 Hue-saturation-intensity (HSI) space

This was one of the first colour transformations to describe colours according to human perception. We can transform the colour (r, g, b) in RGB space to (h, s, i) in HSI space as

$$h = \frac{1}{2\pi} \cdot \arccos \frac{2r - g - b}{2\sqrt{(r-g)^2 + (r-b)(g-b)}}, \quad (\text{A.3})$$

$$i = \frac{1}{3}(r + g + b), \quad (\text{A.4})$$

$$s = 1 - \frac{1}{i} \min\{r, g, b\}, \quad (\text{A.5})$$

in the general case, and as $h = 0, s = 0, i = \frac{1}{3}(r + g + b)$ if $r = g = b$.

This transformation, as many others that separate the chromaticity from the luminance, has the advantage that, by discarding the luminance component, we get a description which is invariant under intensity variations.

In section 3.4, the h, s , and i channels are designated as “hue”, “sat”, and “value” respectively.

Figure A.2 shows these channels for a portion of a test site.

Angulo [ASH03] suggests combining these three channels as follows:

$$c_1 = s h, \quad (\text{A.6})$$

$$c_2 = (1 - s) i. \quad (\text{A.7})$$

In section 3.4, the c_1 and c_2 channels are designated as “sathue” and “satint” respectively.

A.1.4 One-dimensional colour constancy

In discussing in the literature review the advantages of chromaticity/luminance separation for achieving invariance, we assumed that object illumination would vary only in intensity. While this is true for a single image, it may be less so for larger work. The hue of the Sun varies during the day (it is redder at dawn and sunset than at noon), and there may be other phenomena causing hue variations.

Finlayson [Fin00] proposes a 1-dimensional colour space which is also invariant to illuminant colour. Unfortunately, its calculation requires knowledge of two camera-specific parameters, which would be difficult for us to obtain. We will therefore not use this method.

A.1.5 Log-opponent chromaticity

Log-opponent chromaticity coding has been proposed [Fau79, BF00] as a colour transformation because of multiple advantages: First, being a chromaticity coding it separates the chromaticity and luminance parts of colour. Second, it uses opponent chromaticities, like our own visual system, so it may have a strong perceptual relevance. Third, using logarithms means that changes in the illuminant colour translate into translations in the log-opponent chromaticity space. We can then achieve illuminant invariance by mean subtraction.

The two coordinates (l_1, l_2) of log-opponent chromaticity space can be derived from the (r, g, b) coordinates as follows:

$$l_1 = \log r - \log g \quad (\text{A.8})$$

$$l_2 = \log r + \log g - 2 \log b \quad (\text{A.9})$$

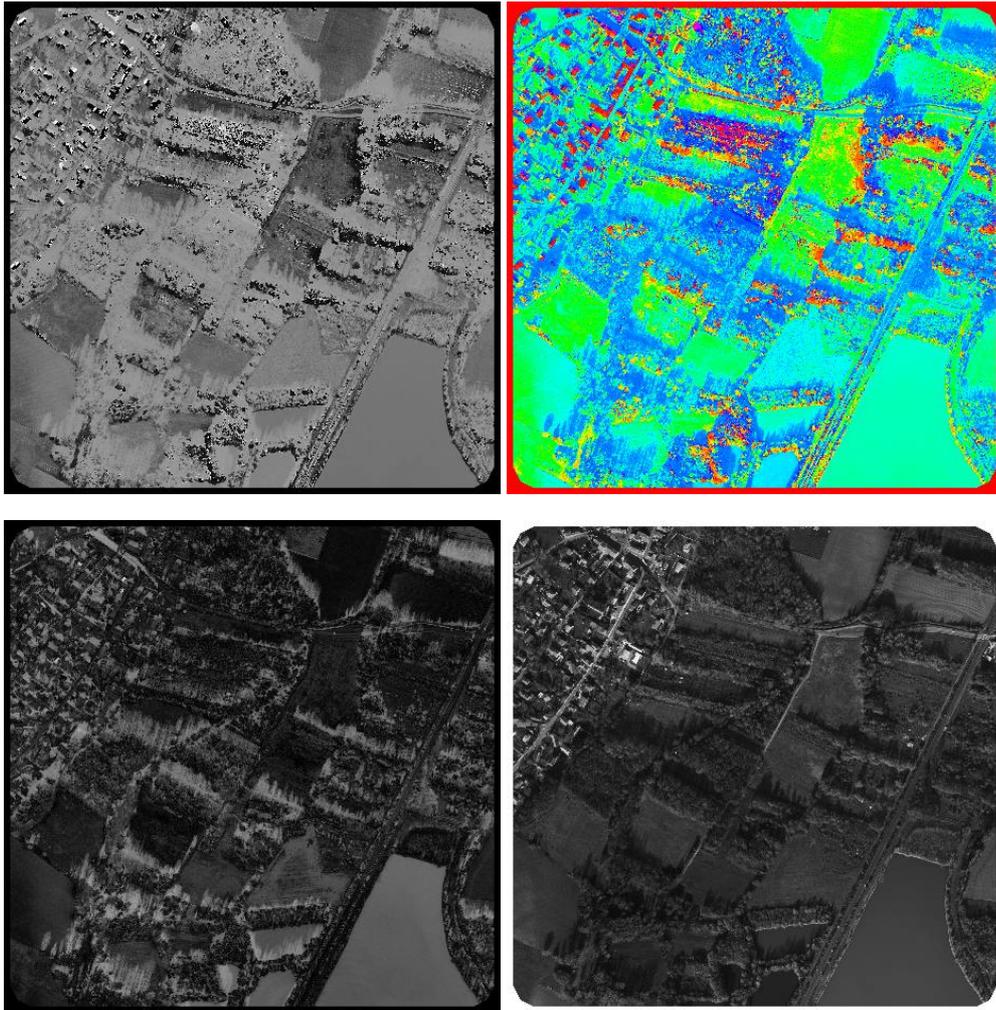


Figure A.2: Hue h (using a linear black-to-white palette; top left), hue (using a circular red-green-blue-red palette; top left), saturation s (bottom left), and value i (bottom right) channels of section A.1.3 for the G003002 section of the Toulouse data set.

Let's assume a simple illumination model whereby an object's perceived colour (r_p, g_p, b_p) depends on the illuminant colour (r_i, g_i, b_i) , and the object's reflectance (r_ρ, g_ρ, b_ρ) as

$$\begin{pmatrix} r_p \\ g_p \\ b_p \end{pmatrix} = k \cdot \begin{pmatrix} r_i & 0 & 0 \\ 0 & g_i & 0 \\ 0 & 0 & b_i \end{pmatrix} \begin{pmatrix} r_\rho \\ g_\rho \\ b_\rho \end{pmatrix}. \quad (\text{A.10})$$

We see that an object's perceived colour in the log-opponent chromaticity space, (l_1, l_2) for a given illuminant colour (r_i, g_i, b_i) and its colour (l'_1, l'_2) under another illuminant colour (r'_i, g'_i, b'_i) are related by the object-independent translation

$$l'_1 = l_1 + \log \frac{r'_i}{r_i} - \log \frac{g'_i}{g_i}, \quad (\text{A.11})$$

$$l'_2 = l_2 + \log \frac{r'_i}{r_i} + \log \frac{g'_i}{g_i} - 2 \log \frac{b'_i}{b_i}. \quad (\text{A.12})$$

In section 3.4, the l_1 and l_2 channels are designated as "log-rg" and "log-rgbb" respectively.

Figure A.3 shows these channels for a portion of a test site.



Figure A.3: Log-opponent chromaticity channels l_1 (left) and l_2 (right) of section A.1.5 for the G003002 section of the Toulouse data set.

A.1.6 Karhunen-Loève colour space

Although in principle a Karhunen-Loève transform [VdWSLVD99] of a RGB colour space is image-dependent, since it depends on the eigenvectors of the correlation matrix of a specific image, in practice the eigenvectors remain approximately the same for a large set of natural colour images. This "fixed" Karhunen-Loève transform reproduces the average "real" Karhunen-Loève transform found when analysing natural images.

Its three components (k_1, k_2, k_3) are derived from the (r, g, b) coordinates as

$$\begin{pmatrix} k_1 \\ k_2 \\ k_3 \end{pmatrix} = \begin{pmatrix} 1/3 & 1/3 & 1/3 \\ 1/2 & 0 & -1/2 \\ -1/2 & 1 & -1/2 \end{pmatrix} \begin{pmatrix} r \\ g \\ b \end{pmatrix}. \quad (\text{A.13})$$

Note that, after all, this is a chromaticity/luminance separation, with k_1 being the intensity, k_2 the red/cyan chromaticity, and k_3 being the green/magenta chromaticity.

In section 3.4, the k_1 , k_2 , and k_3 channels are designated as “kl1”, “kl2”, and “kl3” respectively.

Figure A.4 shows these channels for a portion of a test site.

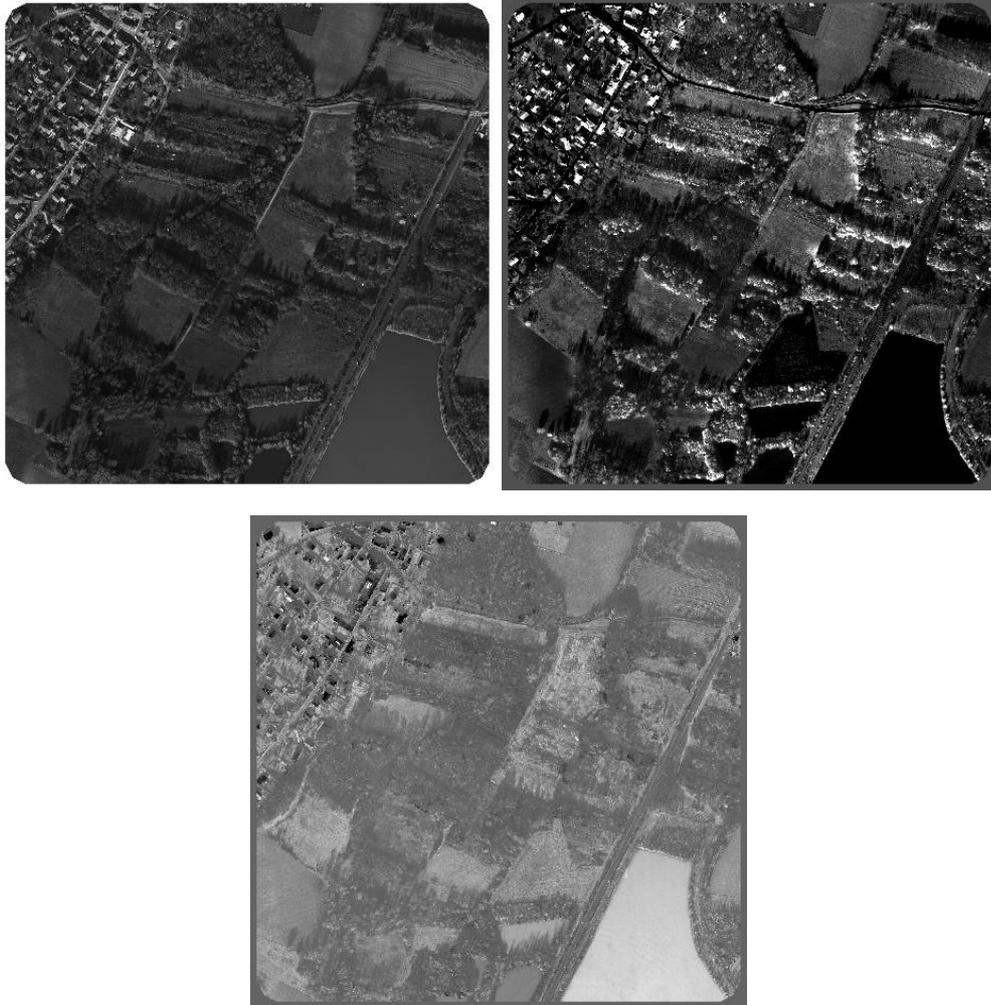


Figure A.4: Karhunen-Loève channels k_1 (top left), k_2 (top right), and k_3 (bottom) of section A.1.6 for the G003002 section of the Toulouse data set.

A.1.7 CIE XYZ, Lab, and Luv

There are several colour spaces defined by the CIE (Commission Internationale de l'Éclairage). The X, Y, Z coordinates are used to represent any colour as a sum of three standard primary colours. Some real colours have negative coordinates in RGB space (and therefore cannot be shown in a computer screen, for example), but they are all representable in XYZ space. The L, a^*, b^* and the L, u^*, v^* colour spaces are transformations of the XYZ space which represent perceived colour difference *linearly* through Euclidean distance [VB00, FP03]; CIE Lab is more apt at measuring large differences, whereas CIE Luv focuses on small differences. Takamura and Kobayashi [TK02] improve the uniformity of the CIE Luv space.

In section 3.4, the X , Y , and Z channels are designated as “ciex”, “ciey”, and “ciez” respectively. The L , a^* , b^* , u^* , and v^* channels are designated as “ciel”, “ciea”, “cieb”, “cieu”, and “ciev” respectively.

Figures A.5 and A.6 show these channels for a portion of a test site.

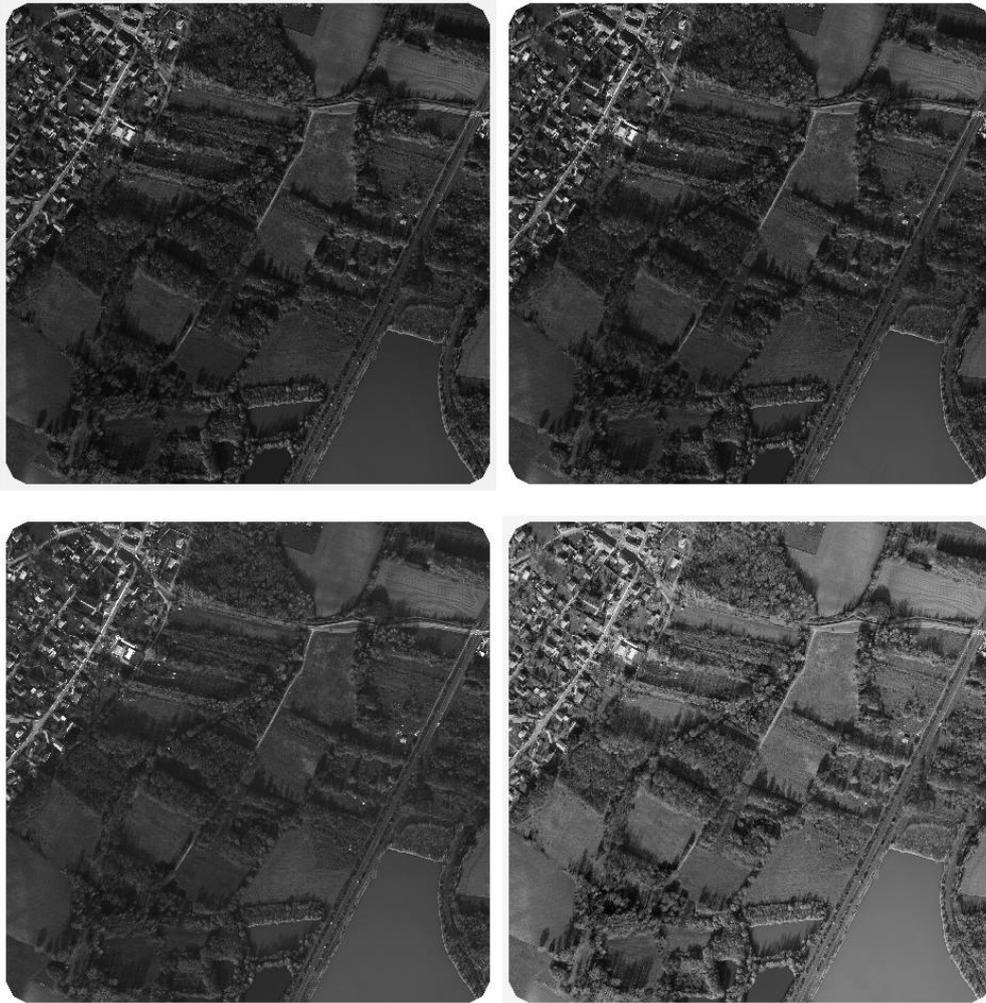


Figure A.5: CIE channels X (top left), Y (top right), Z (bottom left), and L (bottom right) of section A.1.7 for the G003002 section of the Toulouse data set.

A.1.8 Other colour transformations

We also experimented with other colour transformations. From (r, g, b) coordinates,

$$e_{1a} = \log \frac{r}{r + g + b}, \quad e_{1b} = \log \frac{g}{r + g + b}, \quad (\text{A.14})$$

$$e_{2a} = \frac{r}{g}, \quad e_{2b} = \frac{b}{g}, \quad (\text{A.15})$$

$$e_3 = \log \frac{b}{g}. \quad (\text{A.16})$$

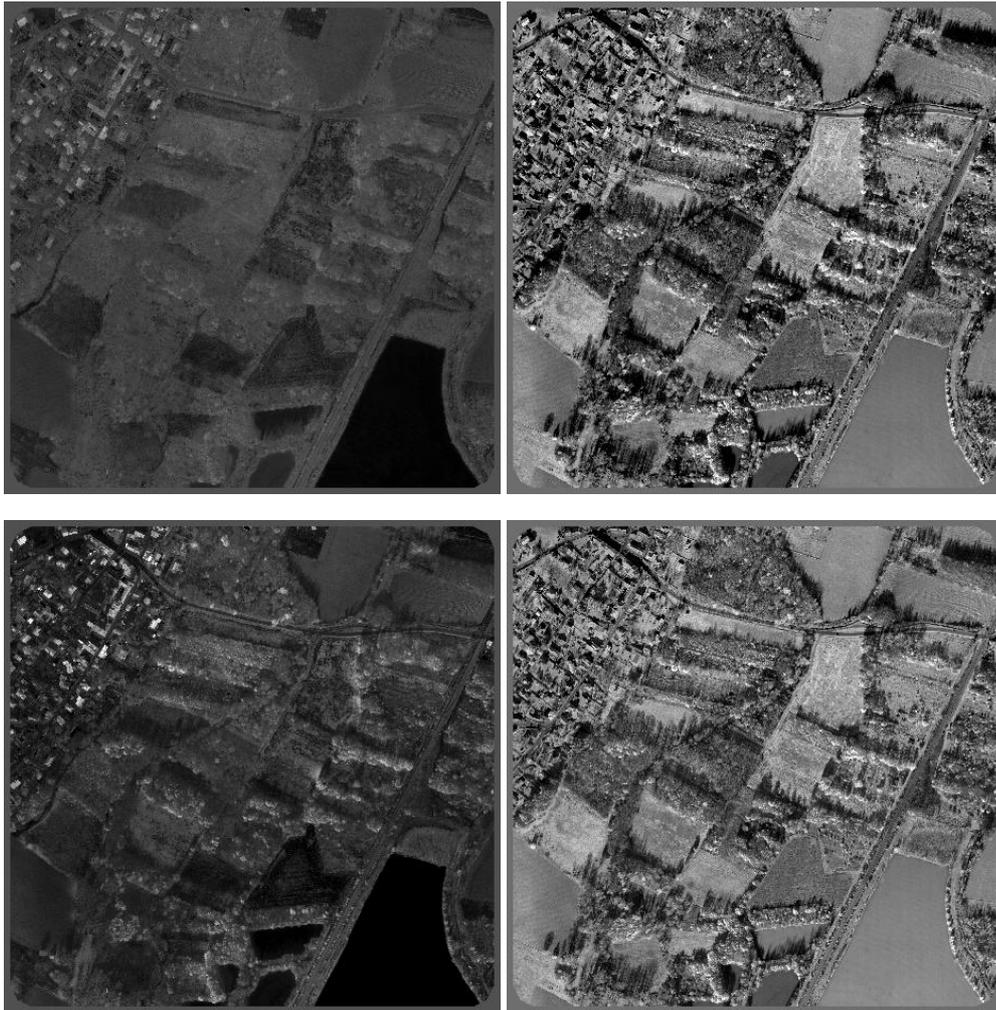


Figure A.6: CIE channels a^* (top left), b^* (top right), u^* (bottom left), and v^* (bottom right) of section A.1.7 for the G003002 section of the Toulouse data set.

In section 3.4, the e_{1a} , e_{1b} , e_{2a} , e_{2b} , and e_3 channels are designated as “log-rs”, “log-gs”, “chr-rg”, “chr-bg”, and “log-bg” respectively.

Figure A.7 shows these channels for a portion of a test site.

A.1.9 Vegetation indices

Instead of using the raw infrared data, it is a common practice to use the *Normalized Difference Vegetation Index*, or NDVI, which is calculated from the red r and near infrared i_n channels as

$$NDVI = \frac{i_n - r}{i_n + r}. \quad (\text{A.17})$$

This index has high values for vegetation areas and low values elsewhere.

Ninomiya [Nin03] proposes another index, the Stabilized Vegetation Index (StVI) which is more stable than NDVI when applied to data without atmospheric correction. It can be computed from the green g , red r , and near infrared i_n bands as

$$StVI = \frac{i_n g}{r^2}. \quad (\text{A.18})$$

For arid and semi-arid regions, Ninomiya suggests using the ratio of StVI to its average across a scene, the standardized StVI, or sStVi.

Tang *et al.* [TZS⁺03] propose an alternative to NDVI, the Three-band Gradient Difference Vegetation Index (TGDVI) with supposedly better performance. It can be computed from the green g , red r , and near infrared i_n bands (whose central wavelengths are, respectively, λ_g , λ_r , and λ_{i_n}) as

$$TGDVI = \min \left\{ 0, \frac{i_n - r}{\lambda_{i_n} - \lambda_r} - \frac{r - g}{\lambda_r - \lambda_g} \right\}. \quad (\text{A.19})$$

In the camera used for the “Toulouse” datasets in this thesis, we have $\lambda_g = 550$ nm, $\lambda_r = 640$ nm, and $\lambda_{i_n} = 900$ nm.

The Water Index

$$WI = \frac{g - r}{g + r}, \quad (\text{A.20})$$

where g and r are the green and red channels respectively, is supposed to be useful for discriminating water.

In section 3.4, the NDVI, TGDVI, and WI channels are designated as “ndvi”, “tgdvi”, and “wi” respectively.

Figure A.8 shows these channels for a portion of a test site.

A.2 Texture parameters

A.2.1 Gabor filters

Gabor filters [Mac91, BS98] are filters localized in space and frequency, which can be used to retrieve frequential properties of a texture. In their most common use, a bank containing a number of Gabor filters is run on an image. This generates a large number of values —since the filters are localized, they have to be applied at different image locations. While this can be used for whole-image retrieval from databases, it has too high a dimensionality for use as segmentation input. I have implemented some aggregate measures more suitable for image segmentation.

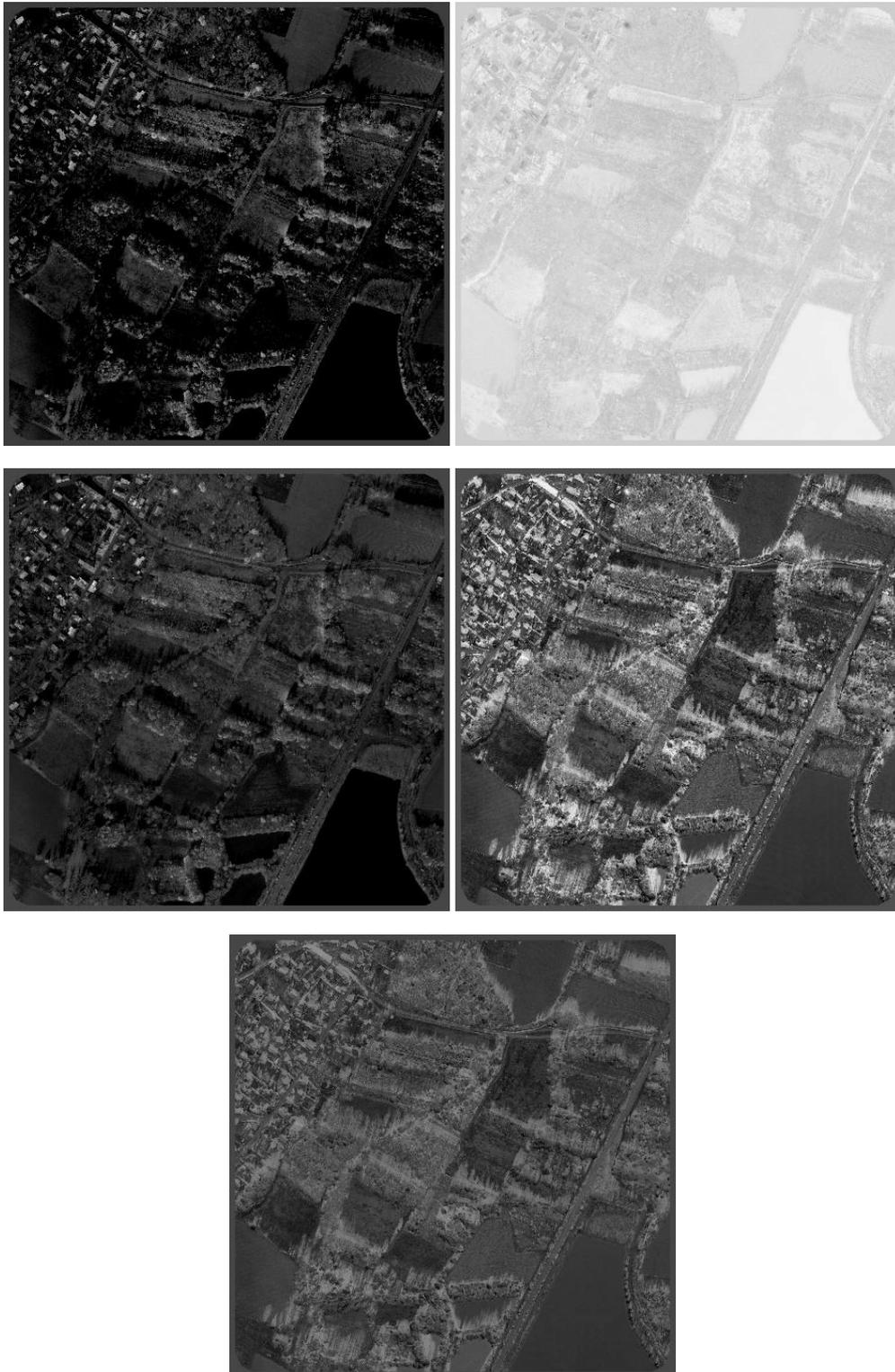


Figure A.7: Channels $e_{1a} = \log r - \log(r + g + b)$ (top left), $e_{1b} = \log g - \log(r + g + b)$ (top right), $e_{2a=r/g}$ (middle left), $e_{2b} = b/g$ (middle right), and $e_3 = \log b/g$ (bottom) of section A.1.8 for the G003002 section of the Toulouse data set.

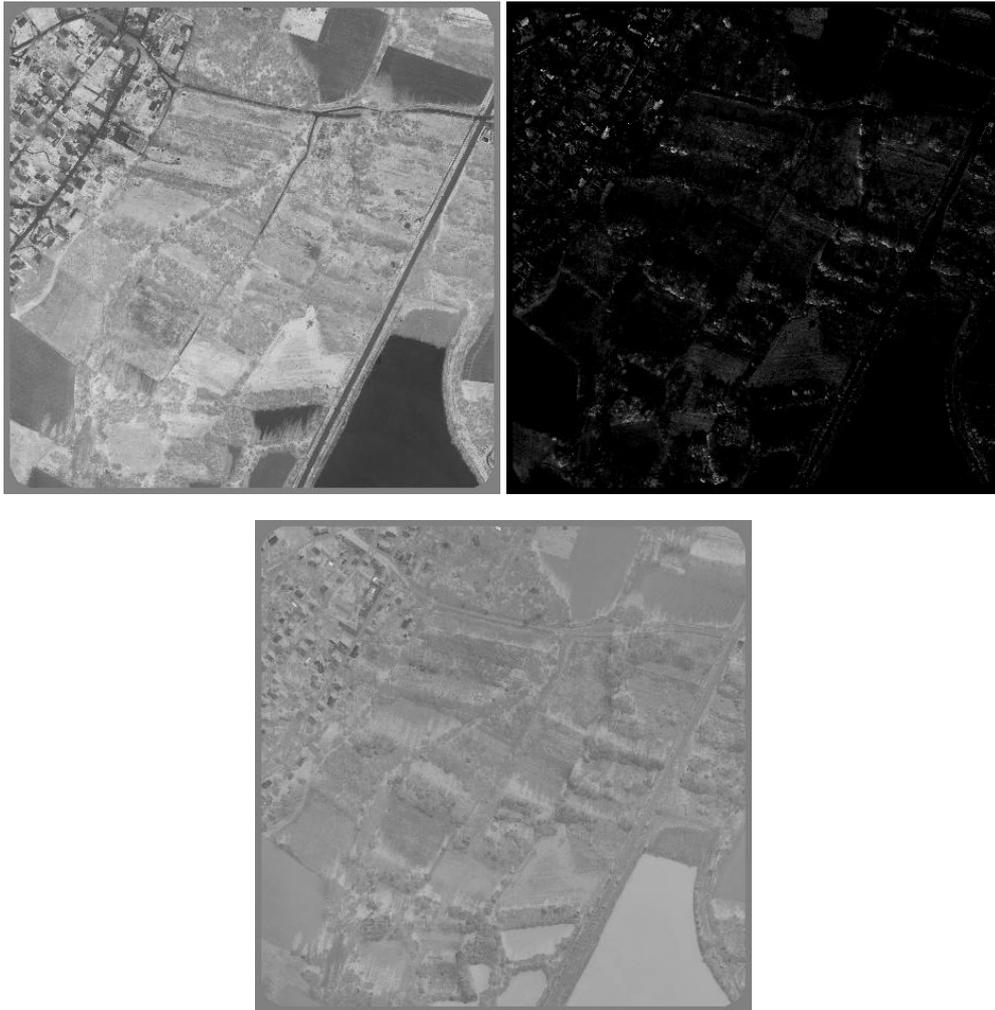


Figure A.8: Channels *NDVI* (top left), *TGDVI* (top right), and *WI* (bottom) of section A.1.9 for the G003002 section of the Toulouse data set.

A Gabor filter is a sine wave, with a given wavelength and direction, localized by multiplication by a Gaussian kernel, centred on a certain pixel and with possibly different variances on the sine wave's direction and its perpendicular direction. By selecting appropriate wavelengths, directions, and variances, the set of Gabor filters centred on a given position can be made to partition the frequency domain into perceptually-relevant domains.

Let α be the angle of the direction of the sine wave (that along which it oscillates fastest), $\lambda = (2\pi\omega)^{-1}$ its wavelength, σ_1^2 the variance of the Gaussian envelope along the sine wave direction, and σ_2^2 that along the perpendicular direction. To obtain a good coverage of the frequency domain, we take $\sigma_1^2 = 4\lambda$ and $\sigma_2^2 = \lambda$. A non-normalized Gabor filter centred on the origin has then the following complex-valued impulse response $G_{\alpha,\omega}(\mathbf{z})$ (with $\mathbf{z} = (x, y)^T$):

$$G_{\alpha,\omega}(\mathbf{z}) = \frac{1}{2\pi\sqrt{\sigma_1^2\sigma_2^2}} e^{ix_r\omega} e^{-\frac{(x_r^2/\sigma_1^2)+(y_r^2/\sigma_2^2)}{2}}, \quad (\text{A.21})$$

with $\mathbf{z}_r = (x_r, y_r)^T = \begin{pmatrix} \cos\alpha & \sin\alpha \\ -\sin\alpha & \cos\alpha \end{pmatrix} \mathbf{z}$.

For values of m in $\{0, 1, \dots, m_{\max} - 1\}$ and values of k in $\{0, 1, \dots, k_{\max} - 1\}$, we calculate the convolution of the image by origin-centred Gabor filters with

$$\alpha = m \frac{\pi}{m_{\max}}, \quad (\text{A.22})$$

$$\omega = 2\pi\nu_0 2^{-k}. \quad (\text{A.23})$$

In this thesis, I took $m_{\max} = 12$, $k_{\max} = 6$, and $\nu_0 = \sqrt{2}$.

Let $R_{mk}(i, j) \in \mathbb{R}$ be the module of the convolution of the image by the Gabor filter given by m and k , $R_{mk} = \|I * G_{\alpha(m),\omega(k)}\|$. For each point \mathbf{u} , the values of m and k giving maximum R_{mk} , and the maximum value of R_{mk} , are used as texture features:

$$T_v(\mathbf{u}) := \max_{m,k} R_{mk}(\mathbf{u}) \quad (\text{A.24})$$

$$(T_m(\mathbf{u}), T_k(\mathbf{u})) := \operatorname{argmax}_{(m,k)} R_{mk}(\mathbf{u}). \quad (\text{A.25})$$

Thus, T_m and T_k give the direction and frequency of the strongest response, and T_v the strength of that response. In this respect, this may be seen as an orientation estimator.

In section 3.4, the T_v , T_m , and T_k channels are designated as “pix-gabor- m ”, “pix-gabor- a ”, and “pix-gabor- f ” respectively.

Figure A.9 shows these channels for a portion of a test site.

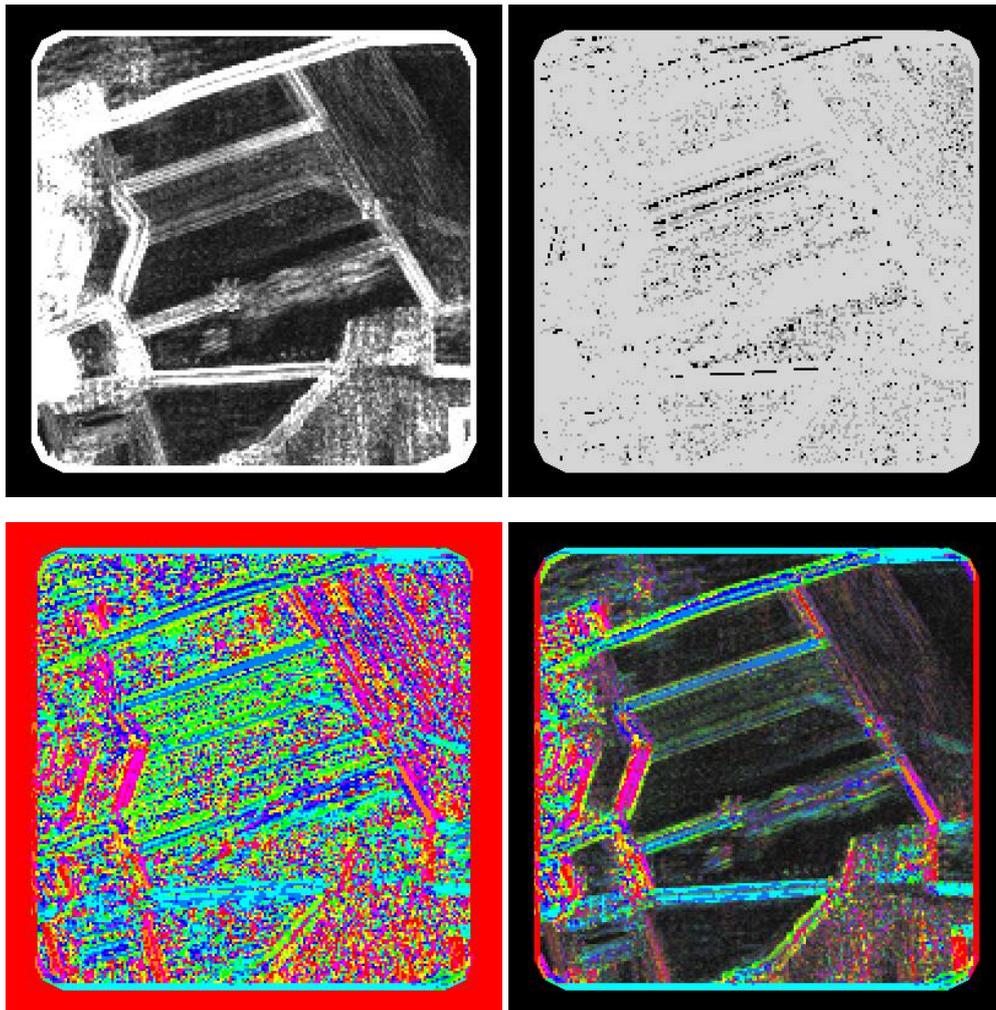
A.2.2 Fractal dimension

The fractal dimension is often used as a measure of complexity [PNHA84, TLT00, TA95]. We have used as texture features the results of Tao, Lam, and Tang's implementation [TLT00] of the *blanket* technique.

This implementation of the blanket technique proceeds as follows. Let a gray-level image (the intensity channel, in our case) be given by the function $I(\mathbf{z})$. Coordinates take integer values.

Given a distance $\delta \in \mathbb{N}$, we define the *upper blanket* at distance δ , $a_\delta(\mathbf{z})$ as

$$a_\delta(\mathbf{z}) = \max\left\{a_{\delta-1}(\mathbf{z}) + 1, \max_{\|\mathbf{u}-\mathbf{z}\|\leq 1} a_{\delta-1}(\mathbf{u})\right\} \quad (\text{A.26})$$



and the *lower blanket* at distance δ , $b_\delta(\mathbf{z})$ as

$$b_\delta(\mathbf{z}) = \min\{b_{\delta-1}(\mathbf{z}) - 1, \min_{\|(\mathbf{u})-(\mathbf{z})\| \leq 1} b_{\delta-1}(\mathbf{u})\}, \quad (\text{A.27})$$

with $\mathbf{z}, \mathbf{u} \in \mathbb{Z}^2$, and $a_0(\mathbf{z}) = b_0(\mathbf{z}) = I(\mathbf{z})$ the image.

The volume between blankets at distance δ is

$$V_\delta = \sum_{\mathbf{z}} (a_\delta(\mathbf{z}) - b_\delta(\mathbf{z})). \quad (\text{A.28})$$

We define

$$A_\delta = \frac{V_\delta}{2\delta} \quad (\text{A.29})$$

and pick two values of δ , δ_1 and δ_2 (in our implementation, $\delta_1 = 1$, $\delta_2 = 4$).

Then, the fractal dimension D can be approximated by

$$D \simeq 2 - \frac{\log_2 A_{\delta_1} - \log_2 A_{\delta_2}}{\log_2 \delta_1 - \log_2 \delta_2}. \quad (\text{A.30})$$

The fractal dimensions of square neighbourhoods of side 16 centred around each pixel are designated, in section 3.4, as “pix-fractal” (for pixel-based analysis). Those of regions of arbitrary shape are designated as “reg-fractal” (for region-based-analysis).

Figure A.10 shows these channels for a portion of a test site.

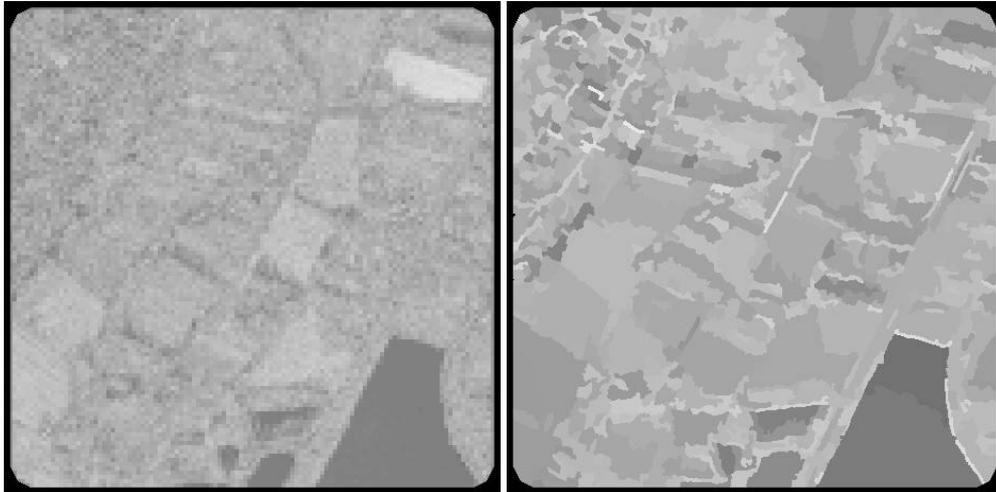


Figure A.10: Fractal dimension for pixel-based analysis (left) and region-based analysis (right) for the G003002 section of the Toulouse data set.

A.2.3 Local Binary Patterns

Combining rotational-invariance (or orientation detection, if needed) with multi-resolution analysis, Ojala, Pietikäinen and Mäenpää [OPM02] propose a texture classification method using *local binary patterns*: These are a reduced number of local patterns whose statistical frequencies are shown to be sufficient for texture classification. This is a complex algorithm that will be described here only briefly. The reader is referred to [OPM02] for a full description.

To calculate the Local Binary Pattern textural features corresponding to a pixel in position z , we start by obtaining the texture vector T_z of gray level values

$$T(z) = (I(z), g_{z,0}, g_{z,1}, \dots, g_{z,P-1}) \quad (\text{A.31})$$

where $g_{z,i}$ for $i = 0, \dots, P-1$ are the intensity values of P equally spaced pixels on a circle of radius R and center z :

$$g_{z,i} = I\left(z + R(\cos 2\pi i/P, \sin 2\pi i/P)^T\right). \quad (\text{A.32})$$

Bicubic interpolation is used to obtain pixel values at non-integer positions. We have used $R = 2$ and $P = 16$.

Then, a new texture vector is obtained that contains zeros and ones depending on whether each $g_{z,i}$ component is above or below $I(z)$:

$$T'(z) = (s_{z,0}, s_{z,1}, \dots, s_{z,P-1}), \quad (\text{A.33})$$

$$s_{z,i} = \begin{cases} 1 & \text{if } g_{z,i} \geq I(z) \\ 0 & \text{if } g_{z,i} < I(z). \end{cases} \quad (\text{A.34})$$

In [OPM02] it is argued that the greatest part of texture discrimination is provided by patterns whose T' vector only has two transitions between ones and zeros (T' should be interpreted circularly: the first and last values are adjacent). For example, the vector (0111110000000000) has two transitions, (1111111111111111) has none, (1111110000000000) also has two, and (0011110000011100) has four.

The number of transitions is calculated as

$$U_z = |s_{z,P-1} - s_{z,0}| + \sum_{p=0}^{P-2} |s_{z,p} - s_{z,p+1}|, \quad (\text{A.35})$$

and a characteristic number is calculated from T' :

$$LBP_{P,R}^{riu2}(z) = \begin{cases} \sum_{p=0}^{P-1} s_{z,p} & \text{if } U_z \leq 2 \\ P + 1 & \text{otherwise.} \end{cases} \quad (\text{A.36})$$

These numbers $LBP_{P,R}^{riu2}$ are invariant to any monotonic transformation of the gray scale, that is, to transformations that preserve the ordering of gray values, and also mostly invariant to rotation—although the image quantization may make it non-invariant in some cases.

Ojala, Pietikäinen and Mäenpää also propose a variance measure in case contrast is significant—since the LBP features discard contrast. From the T vector,

$$VAR_{P,R}(z) = \frac{1}{P} \sum_{p=0}^{P-1} (g_{z,p} - \mu_z)^2, \quad \text{where } \mu_z = \frac{1}{P} \sum_{p=0}^{P-1} g_{z,p}. \quad (\text{A.37})$$

In section 3.4, the $VAR_{P,R}$ feature is designated as “*pix-lbp-var*”. It is only provided for pixel-based analysis.

These texture features describe only individual points. In an extension to [OPM02], we propose a simple method to obtain aggregate descriptors to characterize neighbourhoods.

Let S be an image region. In region-based analysis, S is the region being characterized. In pixel-based analysis, S is a square neighbourhood centred on the current pixel (we have used square neighbourhoods of 8 pixels by 8 pixels).

The $LBP_{P,R}^{riu2}$ number is calculated for all pixels in S . For each different value of $LBP_{P,R}^{riu2}$, we find how many pixels in S give that value. The most frequent value, the *LBP mode*, will be used

as a texture feature. The number of pixels that give that most frequent value, divided by the number of pixels in S , is its *LBP relative frequency* and will also be used as a texture feature.

In section 3.4, the LBP mode and the LBP relative frequency features are designated as “pix-lbp-riu2m” and “pix-lbp-riu2f” respectively, for pixel-based analysis. For region-based analysis, they are designated as “reg-lbp-riu2m” and “reg-lbp-riu2f” respectively.

Figure A.11 shows these channels for a portion of a test site.

A.2.4 Structure complexity

Baillard [Bai97] uses the entropy of the histogram of gradient directions, and Guigues’s variant [Gui00] uses a weighted sum of gradient directions based on Medioni’s tensor voting; both take into account the amount of texture present, giving in effect three classes: flat texture (or untextured), unordered orientations (corresponding to natural textures) and ordered orientations (corresponding to artificial objects). Wang, Lu, and Liu [WLL04] give a short description of tensor voting techniques and several applications, including a discontinuity-preserving smoothing method.

Baillard’s method starts by computing the gradient of the source image. This is decomposed into gradient module $M(\mathbf{z})$ and argument $\Theta(\mathbf{z})$. Let $R(\mathbf{z})$ be a mask defining the region of interest. For region-based texture features, $R = 0$ everywhere except in the region’s support, where $R = 1$. For pixel-based texture features, R will be a Gaussian kernel $\mathcal{N}_{\sigma=10}$ (of $\sigma = 10$) centred on the current pixel.

The number N of pixels in the region weighted by the mask R , the set S of pixels whose gradient module is higher than a threshold c , and the number N_v of pixels in S weighted by the mask R , are calculated as

$$N = \sum_{\mathbf{z}} R(\mathbf{z}), \quad (\text{A.38})$$

$$S = \{\mathbf{w} : M(\mathbf{w}) > c\}, \quad (\text{A.39})$$

$$N_v = \sum_{\mathbf{z} \in S} R(\mathbf{z}). \quad (\text{A.40})$$

In [Bai97], the threshold is set to $c = 6$.

Then, a histogram is constructed from the values of Θ in the region S . If the angle domain $[0, \pi)$ is divided into N_a equally-sized bins, the relative histogram count for the i -th bin is $H(i)$,

$$S(i) = \{\mathbf{w} \in S : i\pi/N_a \leq \Theta(\mathbf{w}) < (i+1)\pi/N_a\}, \quad (\text{A.41})$$

$$H(i) = \frac{1}{N_v} \sum_{\mathbf{z} \in S(i)} R(\mathbf{z}), \quad (\text{A.42})$$

and the histogram entropy is computed as

$$E = - \sum_i H(i) \log_2 H(i). \quad (\text{A.43})$$

The use of N_v/N as a texture feature, and the use of a non-constant mask M in pixel analysis are novel additions to Baillard’s algorithm made in this thesis.

In section 3.4, the ratio N_v/N and the values of E are designated as “pix-ent-c” and “pix-ent-e” respectively for pixel-based texture features. For region-based texture features, they are designated as “reg-ent-c” and “reg-ent-e” respectively.

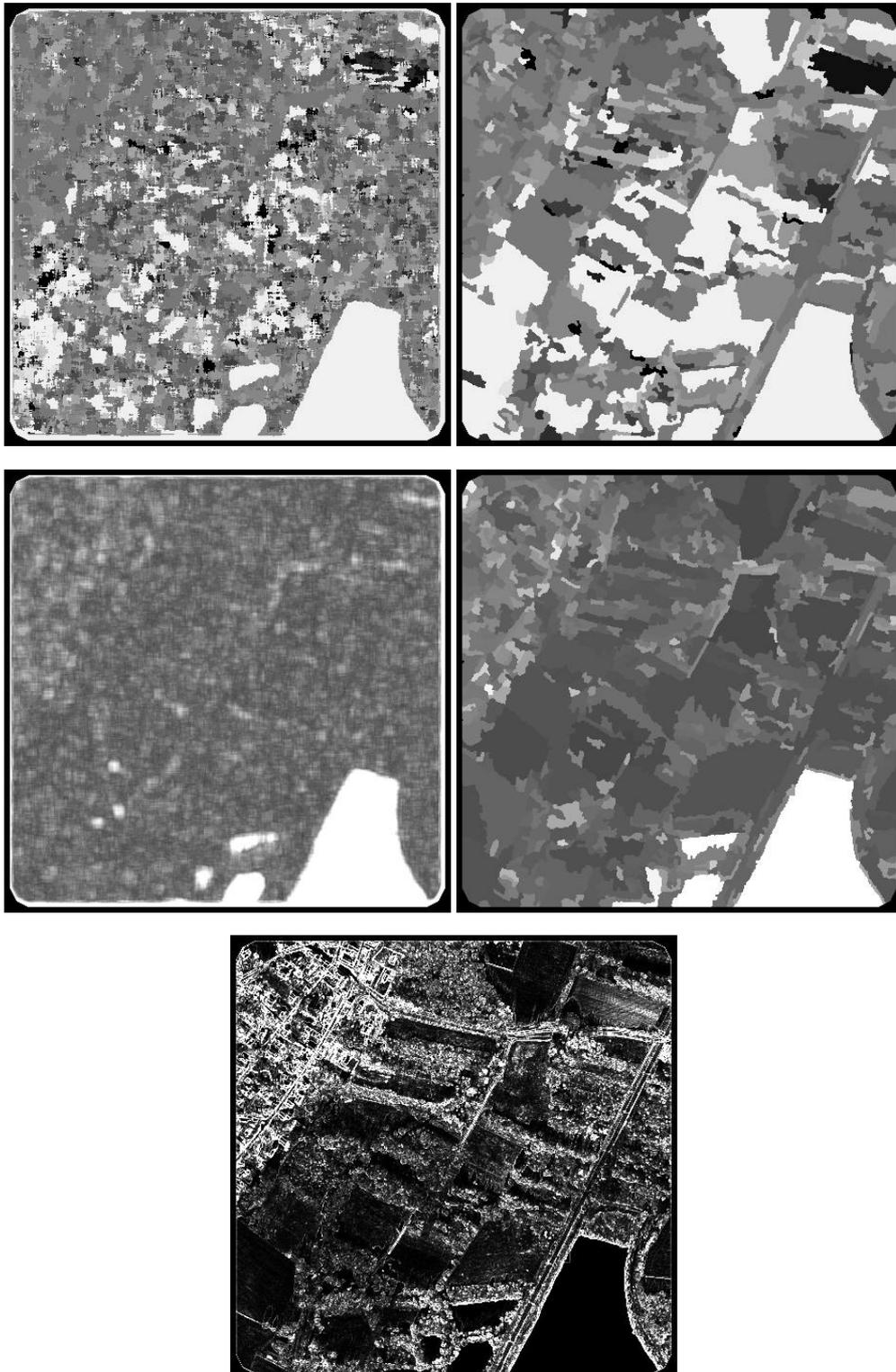


Figure A.11: Local binary pattern channels: LBP mode (top row; pixel-based analysis on the left, region-based on the right), LBP relative frequency (middle row; pixel-based on the left, region-based on the right), and $VAR_{p,R}$ (bottom) for the G003002 section of the Toulouse data set.

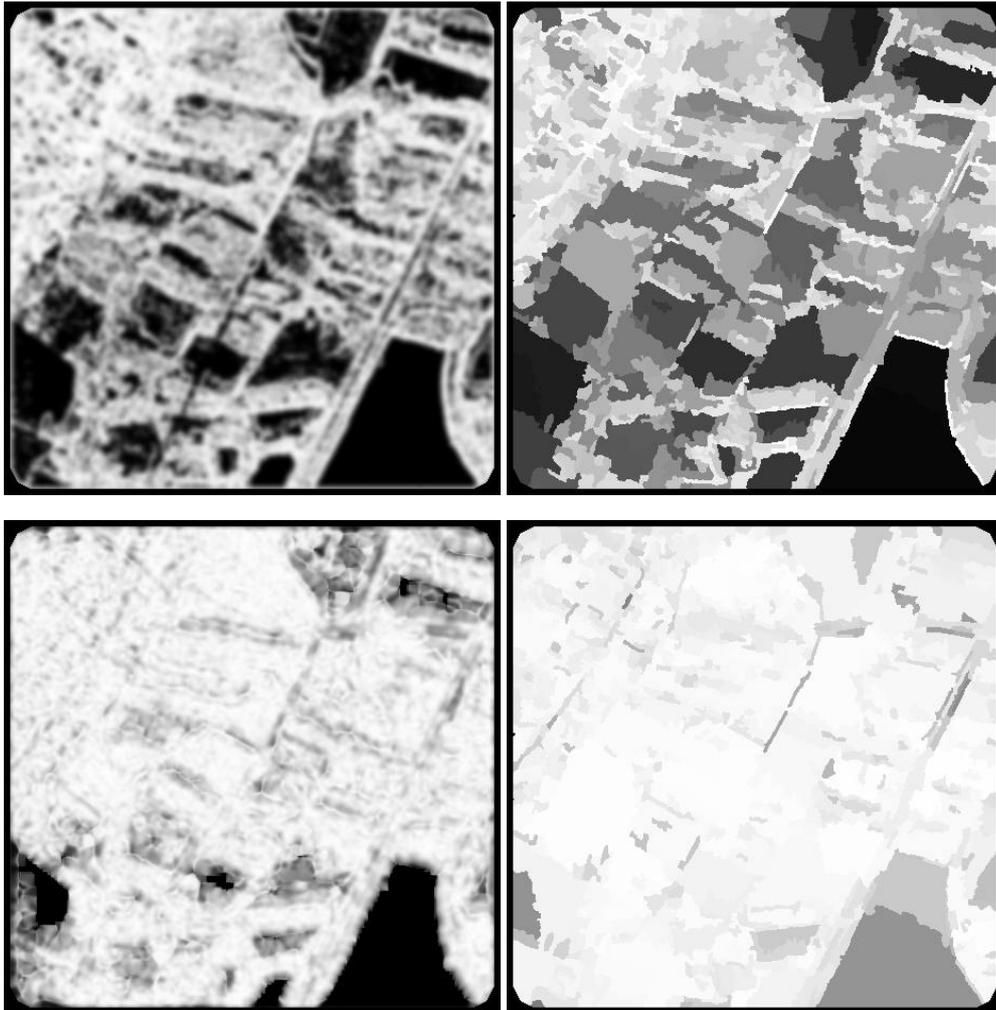


Figure A.12: Structure complexity channels: N_v/N (top; pixel-based analysis on the left, region-based on the right) and E (bottom; pixel-based on the left, region-based on the right) for the G003002 section of the Toulouse data set.

Figure A.12 shows these channels for a portion of a test site.

Guigues's method [Gui00] also starts with the gradient of the source image, in polar form M and Θ .

The gradient angle is multiplied by 4 to obtain a $\pi/2$ invariance,

$$X(z) = M(z) \cos(4\Theta(z)), \quad (\text{A.44})$$

$$Y(z) = M(z) \sin(4\Theta(z)), \quad (\text{A.45})$$

Finally, the gradient module is smoothed by a Gaussian kernel (h_b), and the X and Y functions are used to recompute the gradient module again, after having been *separately* smoothed (h_a) by a Gaussian kernel of $\sigma = 10$:

$$\bar{X}(z) = X(z) * \mathcal{N}_{\sigma=10}(z), \quad (\text{A.46})$$

$$\bar{Y}(z) = Y(z) * \mathcal{N}_{\sigma=10}(z), \quad (\text{A.47})$$

$$a(z) = \sqrt{\bar{X}^2(z) + \bar{Y}^2(z)}, \quad (\text{A.48})$$

$$b(z) = M(z) * \mathcal{N}_{\sigma=10}(z). \quad (\text{A.49})$$

The values b and a/b , for each pixel, are used as pixel-based texture features for that pixel.

In section 3.4, the b and a/b features are designated as "pix-ent-lg-d" and "pix-ent-lg-h" respectively.

Figure A.13 shows these channels for a portion of a test site.

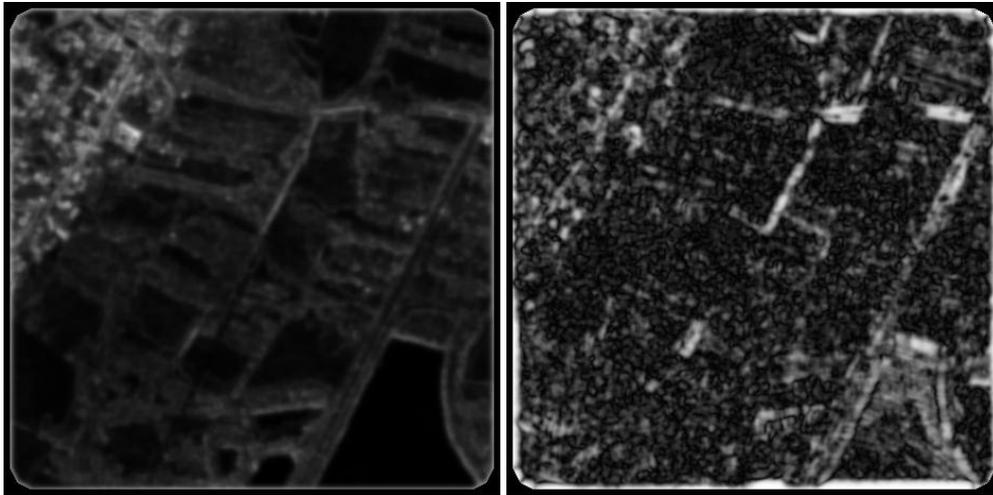


Figure A.13: Structure complexity channels (Guigues' variant): b (left) and a/b (right) for the G003002 section of the Toulouse data set.

A.2.5 Dot-pattern detector

The dot-pattern selective cell operator [KP00] is a biologically motivated texture operator specialized in the detection (and characterization) of dotted textures. We implemented a part of it that seemed useful for land-cover analysis:

To model the centre-surround cells in the primary visual cortex, we define difference-of-Gaussian kernels as follows

$$u_{\xi,\sigma,\gamma,p}(z) = \frac{p}{2\pi\sigma^2} \left(\frac{1}{\gamma^2} e^{-\frac{\|z-\xi\|^2}{2\gamma^2\sigma^2}} - e^{-\frac{\|z-\xi\|^2}{2\sigma^2}} \right), \quad (\text{A.50})$$

where ξ defines the centre of the receptive field, σ is the standard deviation of the surround Gaussian, $\gamma\sigma$ ($\gamma < 1$) is the standard deviation of the centre Gaussian and $p \in \{-1, +1\}$. We used $\gamma = 0.5$. We then compute the kernel response with

$$s_{\sigma,p}(\xi) = \int I(z) u_{\xi,\sigma,p}(x, y) dz, \quad (\text{A.51})$$

where $I(z)$ is the input image. We then perform contrast normalisation by dividing s by the weighted average gray level of the image within the kernel, a ,

$$a_{\sigma}(\xi) = \frac{1}{2\pi\sigma^2} \int I(z) e^{-\frac{\|z-\xi\|^2}{2\sigma^2}} dz, \quad (\text{A.52})$$

to obtain $l_{\sigma,p}(\xi) = s_{\sigma,p}(\xi)/a_{\sigma}(\xi)$, and use the hyperbolic ratio function (with R the maximum response level and C the semi-saturation constant) to obtain the normalized response v ,

$$v_{\sigma,p}(\xi) = \max\left\{0, \frac{l_{\sigma,p}(\xi) \cdot R}{(l_{\sigma,p}(\xi) + C)}\right\}. \quad (\text{A.53})$$

The normalized response v is high for light spots located in the centre of its excitatory region, but it responds also to other features such as light edges. We calculate an excitatory-inhibitory response v' which reacts only at positions of high v response surrounded by low v responses, thereby responding only at spots:

$$v'_{\sigma,p}(\xi) = \begin{cases} v_{\sigma,p}(\xi) & \text{if } \forall i \in \{1..N\} : v_{\sigma,p}(\xi + \delta_i) < \rho v_{\sigma,p}(\xi) \\ 0 & \text{otherwise} \end{cases} \quad (\text{A.54})$$

where $\delta_i = R_{\text{lat}} \cdot (\cos(2\pi i/N), \sin(2\pi i/N))^T$, with $N = 15$, $\rho = 0.8$, and $R_{\text{lat}} = 1.36\sigma$.

We finally calculate, for each position ξ , the σ and p which gives the maximum response ($\sigma_{\text{max}}, p_{\text{max}}$), and this maximum response v'_{max}

$$(\sigma_{\text{max}}(\xi), p_{\text{max}}(\xi)) = \underset{\sigma,p}{\operatorname{argmax}} v'_{\sigma,p}(\xi) \quad (\text{A.55})$$

$$v'_{\text{max}}(\xi) = \max_{\sigma,p} v'_{\sigma,p}(\xi) \quad (\text{A.56})$$

and we use v'_{max} and the product ($p_{\text{max}} \cdot \sigma_{\text{max}}$) as texture parameters.

In section 3.4, the v'_{max} feature is designated as "pix-dot-i". The σ_{max} and p_{max} features are combined into $p_{\text{max}} \cdot \sigma_{\text{max}}$, which is designated as "pix-dot-s".

Figure A.14 shows these channels for a portion of a test site.

A.2.6 Scale-orientation histogram

The scale-orientation histogram [ZXZ03] gives an estimation of the main orientation of a texture, the degree or directionality, and the scale of analysis at which this directionality is found. Of all algorithms, it gives one of the best estimations for orientation; results for degree and scale of directionality are less interesting.

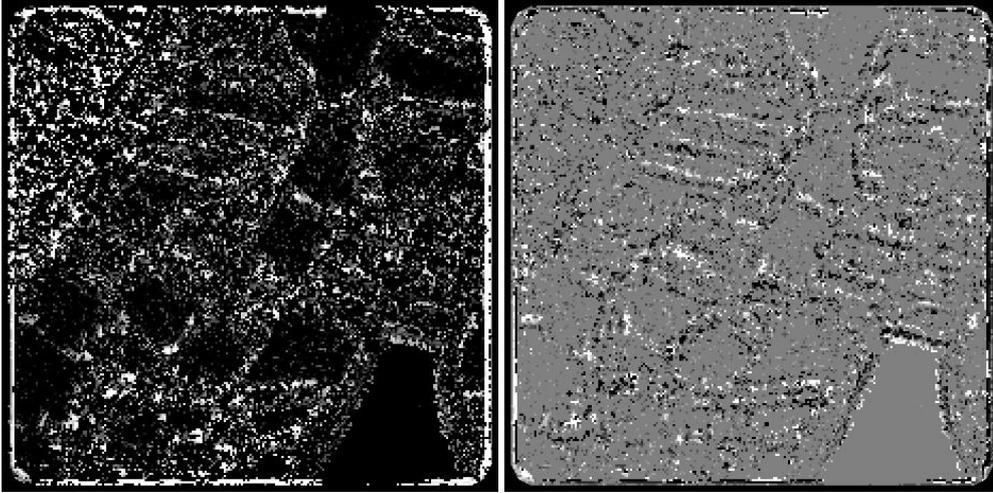


Figure A.14: Dot-pattern detector channels v'_{\max} (left) and $p_{\max} \cdot \sigma_{\max}$ (right) for the G003002 section of the Toulouse data set.

It starts by computing the orientation map θ and the anisotropic strength map g from an intensity image I ,

$$\theta(z, \Omega) = \frac{1}{2} \arctan_2 \frac{S_{\times}(z, \Omega)}{S_{-}(z, \Omega)} + \frac{\pi}{2} \quad (\text{A.57})$$

$$g(z, \Omega) = \frac{S_{-}(z, \Omega)^2 + S_{\times}(z, \Omega)^2}{S_{+}(z, \Omega)^2}, \quad (\text{A.58})$$

where

$$S_{+}(z, \Omega) = \int_{\Omega(z)} (I_x(\mathbf{u})^2 + I_y(\mathbf{u})^2) d\mathbf{u}, \quad (\text{A.59})$$

$$S_{-}(z, \Omega) = \int_{\Omega(z)} (I_x(\mathbf{u})^2 - I_y(\mathbf{u})^2) d\mathbf{u}, \quad (\text{A.60})$$

$$S_{\times}(z, \Omega) = \int_{\Omega(z)} 2I_x(\mathbf{u})^2 I_y(\mathbf{u})^2 d\mathbf{u}, \quad (\text{A.61})$$

$\Omega(x, y)$ is a neighbourhood around pixel (x, y) , \arctan_2 is the 4-quadrant arctangent function, and I_x and I_y are the partial derivatives of $I(x, y)$ over x and y respectively. The anisotropic strength is high for strongly oriented patterns, and close to zero for isotropic regions.

Then, one *scale-orientation histogram* is calculated for the entire image,

$$H(r, a) = \sum_z \{g(z, 2^r \Omega) : \theta(z, 2^r \Omega) = a\}, \quad (\text{A.62})$$

with $r \in \mathbb{R}, a \in [0, \pi)$, where $2^r \Omega(z)$ is a neighbourhood around z of size 2^r that of $\Omega(z)$. Of course, in a real implementation H is calculated as a histogram, adding the values of g for certain bins of r and a .

The original article focuses on producing a single H for an entire image, for use in image retrieval. In order to use that method to describe an arbitrary region of the image, D_i , be it square neighbourhoods around a pixel (for pixel-level analysis) or arbitrary regions (for

region-level analysis), we do the following: For all pixels $z \in D_i$, and for several scales r we calculate $g(z)$ and $\theta(z)$ integrating not over $2^r\Omega(z)$ but over $D_i \cap 2^r\Omega(z)$. We then calculate an $H_i(r, a)$ using only the values of g and θ calculated for D_i . This gives a scale-orientation histogram for every region of analysis D_i .

Since an entire histogram per region is too much data for our purposes, for each histogram $H_i(r, a)$ I propose to calculate an average orientation $\bar{\theta}_i$ and an orientation variance $\sigma_{\theta,i}$: For each r we calculate the average orientation $\bar{\theta}_i(r)$ weighted by $H_i(r, a)$. Because of the periodic nature of a ,

$$\bar{\theta}_i(r) = \frac{\pi}{2} + \frac{1}{2} \arg \sum_a H_i(r, a) \mathbf{u}_a, \quad (\text{A.63})$$

where \mathbf{u}_a is a unit vector in the direction of a , so $\bar{\theta}_i \in [\pi/2, 3\pi/2)$. Then, for each r we calculate the variance of its orientation, also weighted by $H_i(r, a)$,

$$\sigma_{\theta,i}(r) = \frac{\sum_a H_i(r, a) \cdot (a - \frac{\bar{\theta}_i(r)}{\pi})^2}{\sum_a H_i(r, a)}, \quad (\text{A.64})$$

where $\frac{-}{\pi}$ is the π -periodic circular difference, that is (for these values of a and $\bar{\theta}_i$),

$$a - \frac{\bar{\theta}_i}{\pi} := \min\{a - \bar{\theta}_i, a + \pi - \bar{\theta}_i\}. \quad (\text{A.65})$$

For each r , we find $m_i(r) = \max_a H_i(r, a)$. Finally, we calculate $\bar{\theta}_i$ and $\sigma_{\theta,i}$ as

$$\bar{\theta}_i = \frac{\sum_r \bar{\theta}_i(r) \frac{m_i(r)}{\sigma_{\theta,i}(r)}}{\sum_r \frac{m_i(r)}{\sigma_{\theta,i}(r)}}, \quad \sigma_{\theta,i} = \frac{\sum_r \sigma_{\theta,i}(r)}{\sum_r 1}. \quad (\text{A.66})$$

We also found that the raw θ and g were interesting enough. However, since they are calculated at multiple scales we would have too much information. I propose that, for each point $z \in D_i$ the scale r at which the anisotropy g is highest is found. Then, this g and its corresponding θ are selected as texture features:

$$\hat{g}_i(z) = g(z, 2^{\hat{r}_i}\Omega), \quad \hat{\theta}_i(z) = \theta(z, 2^{\hat{r}_i}\Omega), \quad (\text{A.67})$$

where $\hat{r}_i(z) = \operatorname{argmax}_r g(z, 2^r\Omega)$.

The average orientation $\bar{\theta}_i$, the orientation variance $\sigma_{\theta,i}$, \hat{g}_i , and $\hat{\theta}_i$ are designated in section 3.4 as “pix-soh-m”, “pix-soh-v”, “pix-soh-g”, and “pix-soh-th” respectively, for pixel-based texture features. For region-based texture features, they are designated as “reg-soh-m”, “reg-soh-v”, “reg-soh-g”, and “reg-soh-th” respectively.

Figures A.15 and A.16 show these channels for a portion of a test site.

A.2.7 Fourier-based orientation estimator

There are many texture extraction methods that use Fourier analysis. Although in chapter 6 I give some reasons why this may not be a good approach in this case, in this section one of these methods will be presented.

Garnier [Gar02] proposes a simple but effective method that gives the main orientation and spatial period of a texture, when they exist, and also the secondary orientation and period, for textures with at least two prominent directions. This can be used to distinguish among vegetation without a prominent direction (forest, flat fields), those with one prominent direction (ploughed fields and some vineyards), and those with two (some vineyards, planted forest).

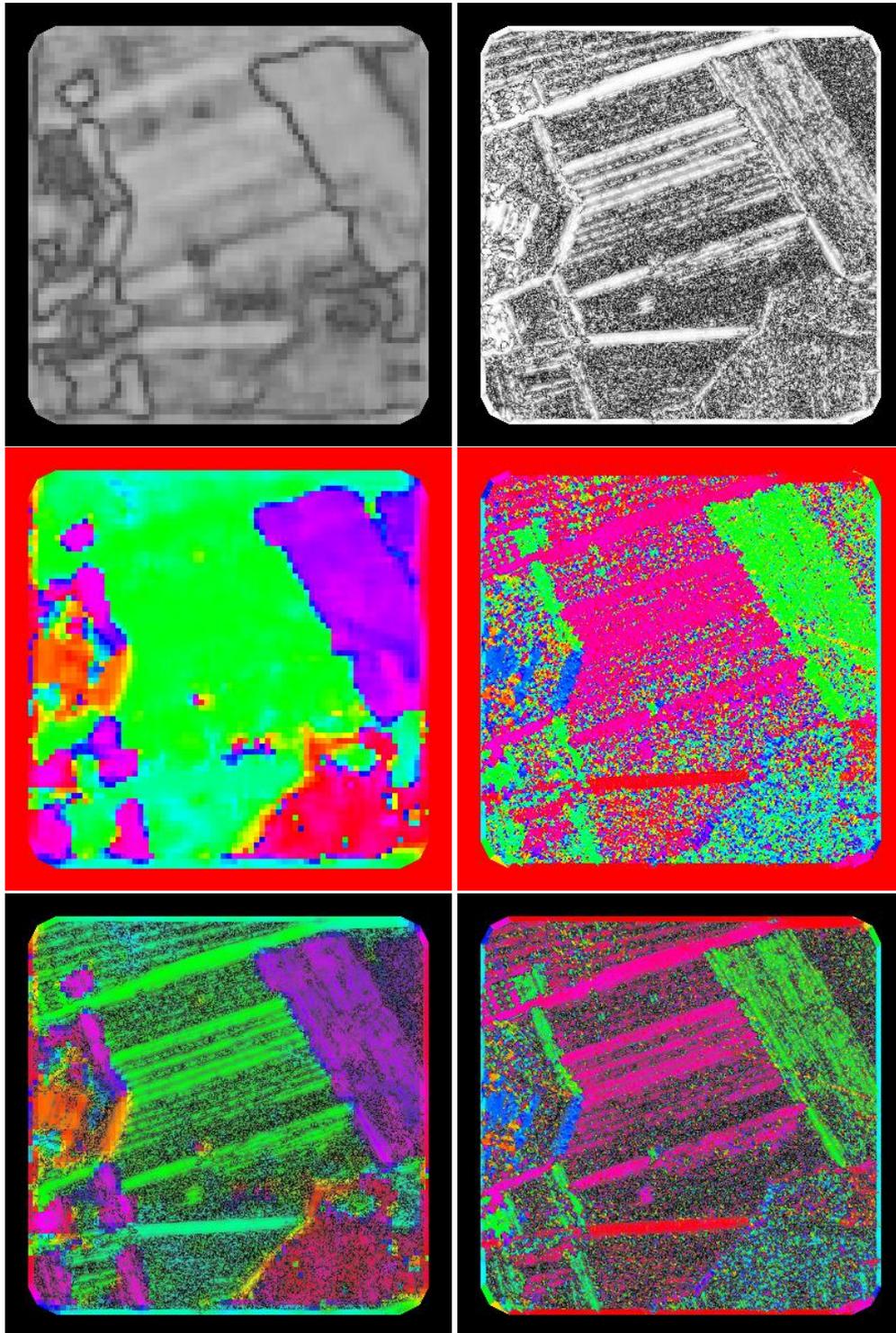


Figure A.15: Scale-orientation histogram channels $\sigma_{\theta,i}$ (top left), $\hat{\theta}_i$ (top right), $\bar{\theta}_i$ (middle left), and $\hat{\theta}_i$ (middle right) for a pixel-level analysis of the G002003 section of the St. Léger data set. The bottom left and right images have $\bar{\theta}_i$ and $\hat{\theta}_i$ as their hue components respectively, and both have $\hat{\theta}_i$ as their value and saturation components. The images in the middle and bottom rows are shown with a circular red-green-blue-red palette.

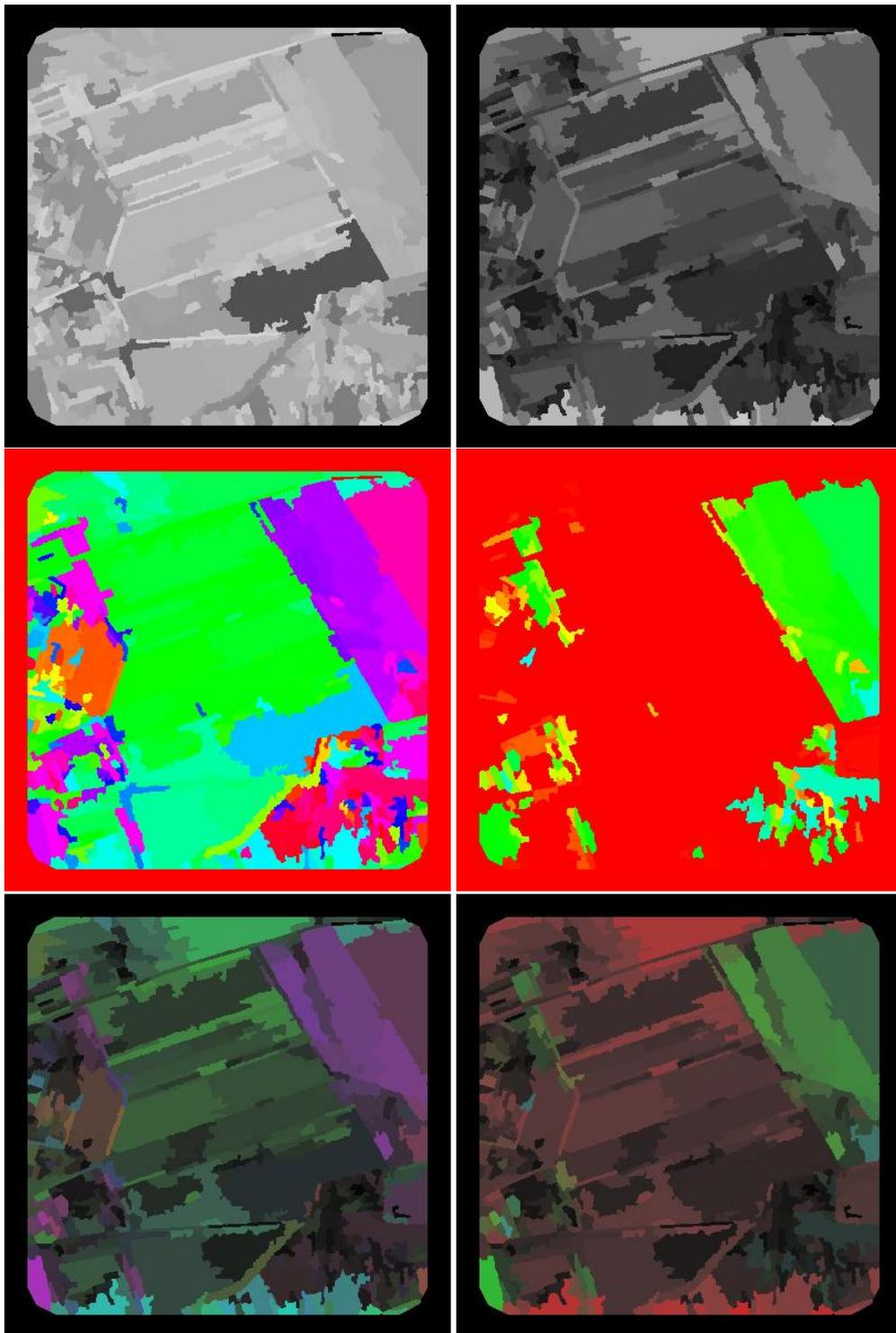


Figure A.16: Scale-orientation histogram channels $\sigma_{\theta,i}$ (top left), \hat{g}_i (top right), $\bar{\theta}_i$ (middle left), and $\hat{\theta}_i$ (middle right) for a region-level analysis of the G002003 section of the St. Léger data set. The bottom left and right images have $\bar{\theta}_i$ and $\hat{\theta}_i$ as their hue components respectively, and both have \hat{g}_i as their value and saturation components. The images in the middle and bottom rows are shown with a circular red-green-blue-red palette.

The period information may be used to differentiate between different kinds of vineyards, for example.

The estimator works as follows:

We start by multiplying the subset of interest of our image by a large Gaussian kernel roughly covering the whole subset. This subset can be a square centred on the pixel of interest (for pixel-level analysis), or an arbitrarily shaped region (for region-level analysis). Then, we calculate the Fourier transform of the result (using the FFT algorithm). Let $F(x, y)$ be this Fourier transform. Multiplying the image by a Gaussian mask prior to the FFT is important to reduce the frequential artifacts that would be produced by the sudden changes at edges of the subset of interest.

The Fourier transform is converted to polar coordinates as

$$F_p(\rho, \theta) = \rho \cdot F(\rho \cos \theta, \rho \sin \theta) \quad (\text{A.68})$$

for $\theta \in [0, \pi)$ and $\rho \in [0, \rho_{\max}]$. Bicubic interpolation is used to obtain values of F at non-integer positions. This is then marginalized over ρ giving

$$F_m(\theta) = \int F_p(\rho, \theta) d\rho, \quad (\text{A.69})$$

and smoothed by circular convolution on $[0, \pi)$ with a Gaussian kernel of $\sigma = 4/3$,

$$F_s(\theta) = F_m(\theta) \circledast \mathcal{N}_{\sigma=4/3}(\theta). \quad (\text{A.70})$$

Note that these calculations are actually made for discretised values of ρ and θ .

Then, the local maxima and minima in $F_s(\theta)$ are determined. Let θ_i be the angle at which the i -th local extrema in $F_s(\theta)$ is. Maxima i for which

$$\frac{F_s(\theta_i)}{F_s(\theta_{i-1})} > c, \quad \frac{F_s(\theta_i)}{F_s(\theta_{i+1})} > c, \quad \text{and } F_s(\theta_i) > s \cdot \frac{1}{\pi} \int_0^\pi F_s(a) da \quad (\text{A.71})$$

are considered for the next step. The values of c and s are set in [Gar02] to $c = 1.2$ and $s = 1.6$. For each such maximum, we calculate its "strength" as

$$S_i = \frac{\int_{\theta_{i-1}}^{\theta_{i+1}} F_s(a) da}{\int_0^\pi F_s(a) da}. \quad (\text{A.72})$$

Let γ_k be the angle in $F_s(\gamma)$ of the k -th maxima that passes the previous tests on c and s , and Σ_k its strength. These maxima correspond to orientations in the input texture.

For each of the maxima γ_k found in the previous step, the periodicity of the texture in that orientation is calculated. This is done, for each γ_k separately, as follows: First, a cut is taken in the polar Fourier transform in the γ_k direction, and this is smoothed by convolution with a Gaussian kernel of $\sigma = 4/3$:

$$F_\gamma(\rho) = F_p(\rho, \gamma_k) * \mathcal{N}_{\sigma=4/3}(\rho). \quad (\text{A.73})$$

Then, using the same approach as in the previous step, the local maxima and minima in $F_\gamma(\rho)$ are determined. Let ρ_i be the radius at which there is the i -th local extrema in $F_\gamma(\rho)$ (the first one having $i = 1$). The maximum with smallest i for which

$$\frac{F_\gamma(\rho_i)}{F_\gamma(\rho_{i-1})} > c, \quad \frac{F_\gamma(\rho_i)}{F_\gamma(\rho_{i+1})} > c, \quad i > 2, \quad \text{and } F_\gamma(\rho_i) > s \cdot \frac{1}{\rho_{\max}} \int_0^{\rho_{\max}} F_\gamma(r) dr \quad (\text{A.74})$$

is taken as the frequency ν_k of the input texture in that direction. Note that it is possible that no such maximum exists; in that case, we set $\nu_k = 0$. The period T_k (in pixels) can be derived from ν and the size of the FFT matrix.

This gives a set of significant directions γ_k , with their strengths Σ_k and periods T_k . The two directions with highest strength Σ are selected. The direction of highest strength is taken as texture feature d_1 , its period p_1 , the direction of second-highest strength is taken as d_2 , and its period p_2 . If less than two significant directions are found, arbitrary angles are used, and periods are set to 0.

The primary direction d_1 , its period p_1 , the secondary direction d_2 , and its period p_2 are designated in section 3.4 as “pix-aurelie-d1”, “pix-aurelie-p1”, “pix-aurelie-d2”, and “pix-aurelie-p2” respectively, for pixel-based texture features. For region-based texture features, they are designated as “reg-aurelie-d1”, “reg-aurelie-p1”, “reg-aurelie-d2”, and “reg-aurelie-p2” respectively.

Figures A.17 and A.18 show these channels for a portion of a test site.

A.2.8 Intensity average

In addition, the average of pixel intensity (channel i in section A.1.3) across all pixels in a region is also calculated as a “region-based” texture feature.

This is designated in section 3.4 as “reg-value”.

Figure A.19 shows this channel for a portion of a test site.

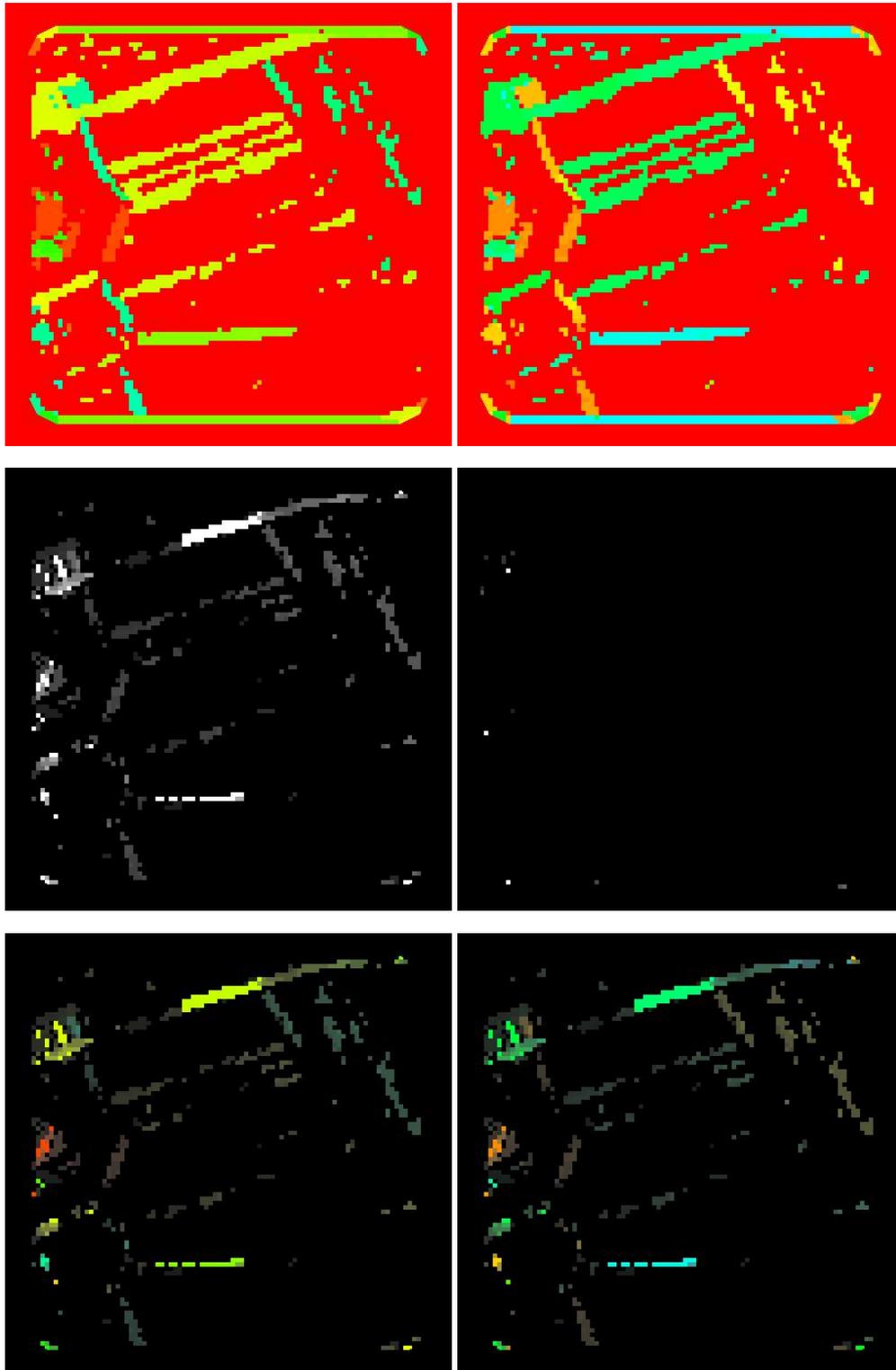


Figure A.17: Garnier's Fourier-based orientation estimator: Channels d_1 (top left), d_2 (top right), p_1 (middle left), and p_2 (middle right) for a pixel-level analysis of the G002003 section of the St. Léger data set. The bottom left and right images have d_1 and d_2 as their hue components respectively, and both have p_1 as their value and saturation components. The images in the top and bottom rows are shown with a circular red-green-blue-red palette.

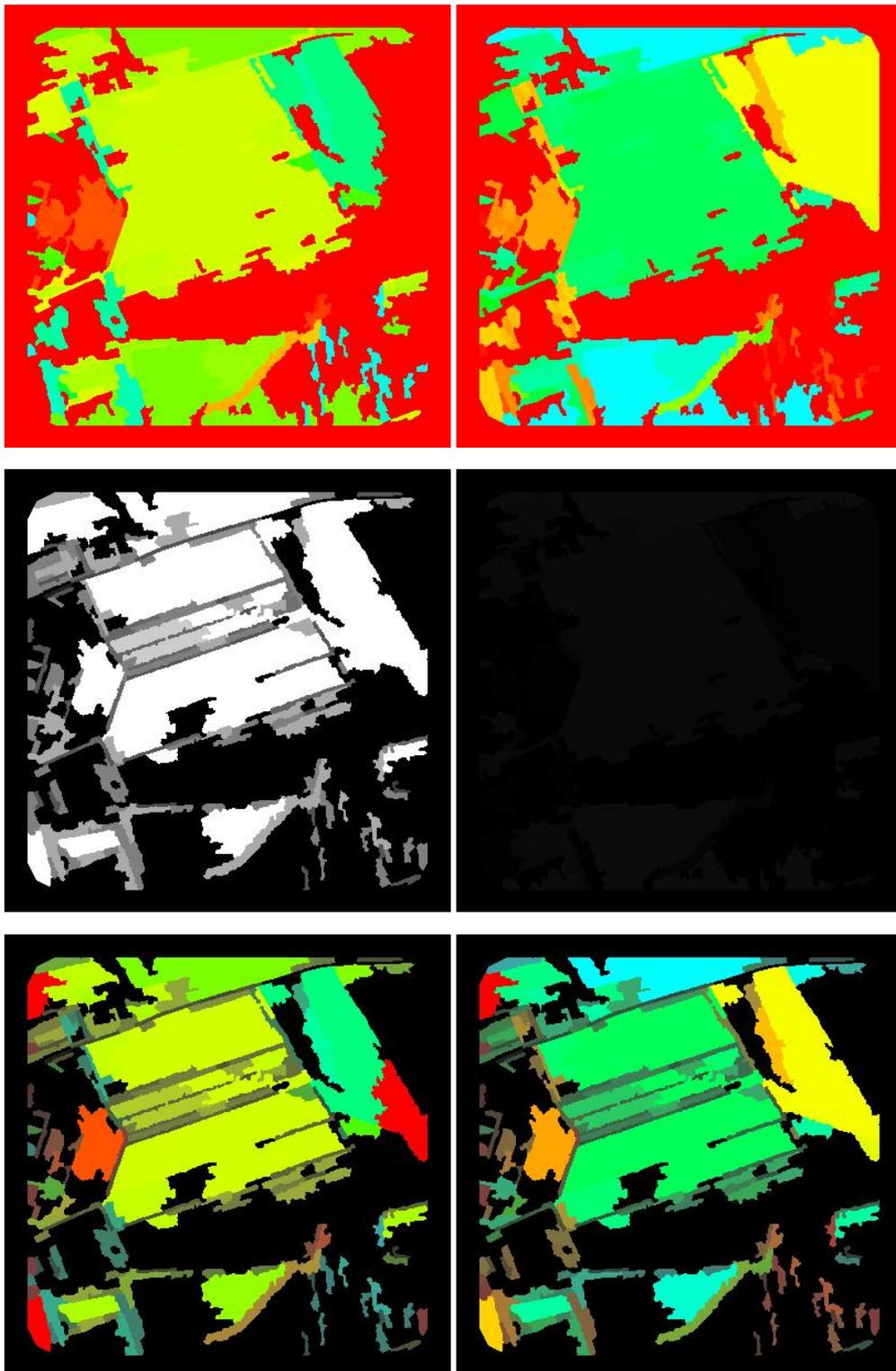


Figure A.18: Garnier's Fourier-based orientation estimator: Channels d_1 (top left), d_2 (top right), p_1 (middle left), and p_2 (middle right) for a region-level analysis of the G002003 section of the St. Léger data set. The bottom left and right images have d_1 and d_2 as their hue components respectively, and both have p_1 as their value and saturation components. The images in the top and bottom rows are shown with a circular red-green-blue-red palette.

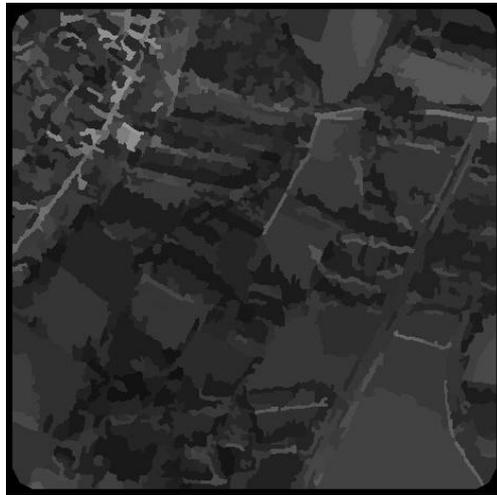


Figure A.19: Average of pixel intensity across all pixels in a region, for the G003002 section of the Toulouse data set.

B | Image segmentation tests

This appendix contains the results of the tests for the selection of the best parameter sets of section 3.4, which, for clarity, are not included in the main text.

In these tables, the M column is the *missed detection* quality measure, and the F column is the *false detection* quality measure, both defined in section 3.3.

Number	Input channels (R_1) / Input channels (R_2)	M	F
1.1	<i>red</i> / \emptyset	0.3476	0.2731
1.3	<i>green</i> / \emptyset	0.3194	0.2570
1.5	<i>blue</i> / \emptyset	0.3538	0.2576
1.7	$\cos(\text{hue}), \sin(\text{hue})$ / \emptyset	0.3186	0.3103
1.9	$\cos(\text{hue})$ / \emptyset	0.3698	0.3407
1.11	$\sin(\text{hue})$ / \emptyset	0.3459	0.3486
1.13	<i>sat</i> / \emptyset	0.3789	0.3447
1.15	<i>value</i> / \emptyset	0.3466	0.2629
1.19	<i>kl2</i> / \emptyset	0.3699	0.3575
1.21	<i>kl3</i> / \emptyset	0.4208	0.3552
1.23	<i>ciex</i> / \emptyset	0.3189	0.2621
1.25	<i>ciey</i> / \emptyset	0.3507	0.2602
1.27	<i>ciez</i> / \emptyset	0.3466	0.2529
1.29	<i>ciel</i> / \emptyset	0.3717	0.2728
1.31	<i>ciea</i> / \emptyset	0.3975	0.3656
1.33	<i>cieb</i> / \emptyset	0.3588	0.3433
1.35	<i>cieu</i> / \emptyset	0.3721	0.3567
1.37	<i>ciev</i> / \emptyset	0.3667	0.3395
1.39	<i>log-rg</i> / \emptyset	0.3999	0.3602
1.41	<i>log-rgb</i> / \emptyset	0.3081	0.3270
1.43	<i>log-rs</i> / \emptyset	0.4061	0.3493
1.45	<i>log-gs</i> / \emptyset	0.4051	0.3521
1.47	<i>chr-rg</i> / \emptyset	0.3855	0.3648
1.49	<i>chr-bg</i> / \emptyset	0.3506	0.3385
1.51	<i>log-bg</i> / \emptyset	0.3551	0.3194
1.53	<i>pix-gabor-m</i> / \emptyset	0.8852	0.7474

Table B.1: Results of the first step of the Stepwise Forward Selection for the selection of the best parameter sets. Because two-phase mode is not used in these tests, region-based input channels cannot be used. Pareto-optimal parameter sets are marked in green. Some close-to-optimal parameter sets, marked in yellow, are also retained for the next step. Parameter sets on white background in a green frame are Pareto-optimal but will not be retained for the next step.

Number	Input channels (R_1) / Input channels (R_2)	M	F
1.57	$\cos(\text{pix-gabor-a}), \sin(\text{pix-gabor-a}) / \emptyset$	0.6679	0.5110
1.59	$\cos(\text{pix-gabor-a}) / \emptyset$	0.6921	0.5991
1.61	$\sin(\text{pix-gabor-a}) / \emptyset$	0.6816	0.5430
1.63	$\text{pix-fractal} / \emptyset$	0.5278	0.3471
1.65	$\text{pix-lbp-var} / \emptyset$	0.3942	0.3002
1.67	$\text{pix-lbp-riu2m} / \emptyset$	0.8614	0.7357
1.69	$\text{pix-lbp-riu2f} / \emptyset$	0.6951	0.5496
1.71	$\text{pix-ent-e} / \emptyset$	0.6669	0.4503
1.73	$\text{pix-ent-c} / \emptyset$	0.6672	0.3892
1.75	$\text{pix-ent-lg-h} / \emptyset$	0.6124	0.4352
1.77	$\text{pix-ent-lg-d} / \emptyset$	0.5534	0.2202
1.79	$\text{pix-dot-i} / \emptyset$	0.6461	0.4688
1.83	$\cos(\text{pix-soh-th}), \sin(\text{pix-soh-th}) / \emptyset$	0.6092	0.5599
1.85	$\cos(\text{pix-soh-th}) / \emptyset$	0.6814	0.6154
1.87	$\sin(\text{pix-soh-th}) / \emptyset$	0.7165	0.6210
1.89	$\text{pix-soh-g} / \emptyset$	0.6211	0.5130
1.91	$\cos(\text{pix-soh-m}), \sin(\text{pix-soh-m}) / \emptyset$	0.8493	0.6871
1.95	$\sin(\text{pix-soh-m}) / \emptyset$	0.8554	0.6885
1.97	$\text{pix-soh-v} / \emptyset$	0.8220	0.6803
1.99	$\cos(\text{pix-aurelie-d1}), \sin(\text{pix-aurelie-d1}) / \emptyset$	0.8179	0.5138
1.101	$\cos(\text{pix-aurelie-d1}) / \emptyset$	0.8264	0.5364
1.103	$\sin(\text{pix-aurelie-d1}) / \emptyset$	0.7505	0.5200
1.105	$\text{pix-aurelie-p1} / \emptyset$	0.7765	0.3492
1.107	$\cos(\text{pix-aurelie-d2}), \sin(\text{pix-aurelie-d2}) / \emptyset$	0.8263	0.5209
1.109	$\cos(\text{pix-aurelie-d2}) / \emptyset$	0.8347	0.5402
1.111	$\sin(\text{pix-aurelie-d2}) / \emptyset$	0.7951	0.5232
1.113	$\text{pix-aurelie-p2} / \emptyset$	0.9834	0.7284
1.115	$\cos(\text{sathue}), \sin(\text{sathue}) / \emptyset$	0.4165	0.3715
1.119	$\sin(\text{sathue}) / \emptyset$	0.4243	0.3714
1.121	$\text{satint} / \emptyset$	0.3425	0.2612
1.123	tgdvi / \emptyset	0.9208	0.5765
1.125	wi / \emptyset	0.3971	0.3623
1.127	$\text{red-z} / \emptyset$	0.3539	0.2849
1.129	$\text{green-z} / \emptyset$	0.3361	0.2698
1.131	$\text{blue-z} / \emptyset$	0.3306	0.2573

Table B.1: Results of the first step of the Stepwise Forward Selection for the selection of the best parameter sets. Because two-phase mode is not used in these tests, region-based input channels cannot be used. Pareto-optimal parameter sets are marked in green. Some close-to-optimal parameter sets, marked in yellow, are also retained for the next step. Parameter sets on white background in a green frame are Pareto-optimal but will not be retained for the next step.

Number	Input channels (R_1) / Input channels (R_2)	M	F
2.1	$\text{green}, \text{red} / \emptyset$	0.3311	0.2616

Table B.2: Results of the second step of the Stepwise Forward Selection for the selection of the best parameter sets. Pareto-optimal parameter sets are marked in green.

Number	Input channels (R_1) / Input channels (R_2)	M	F
2.5	<i>green, blue</i> / \emptyset	0.3649	0.2514
2.8	<i>green, cos(hue), sin(hue)</i> / \emptyset	0.3552	0.2820
2.10	<i>green, cos(hue)</i> / \emptyset	0.3541	0.3010
2.12	<i>green, sin(hue)</i> / \emptyset	0.3754	0.2918
2.14	<i>green, green · cos(hue), green · sin(hue)</i> / \emptyset	0.3811	0.2784
2.16	<i>green, green · cos(hue)</i> / \emptyset	0.3585	0.2886
2.18	<i>green, green · sin(hue)</i> / \emptyset	0.4177	0.3089
2.20	<i>green, (1 - green) · cos(hue), (1 - green) · sin(hue)</i> / \emptyset	0.3862	0.2944
2.22	<i>green, (1 - green) · cos(hue)</i> / \emptyset	0.3624	0.2758
2.24	<i>green, (1 - green) · sin(hue)</i> / \emptyset	0.3536	0.2903
2.29	<i>green, sat</i> / \emptyset	0.3211	0.2859
2.32	<i>green, value</i> / \emptyset	0.3484	0.2603
2.35	<i>green, kl2</i> / \emptyset	0.3434	0.3052
2.38	<i>green, ciex</i> / \emptyset	0.4336	0.2509
2.41	<i>green, ciey</i> / \emptyset	0.3262	0.2688
2.44	<i>green, ciez</i> / \emptyset	0.3218	0.2520
2.50	<i>green, cieb</i> / \emptyset	0.3487	0.3045
2.53	<i>green, cieu</i> / \emptyset	0.3394	0.2699
2.56	<i>green, ciev</i> / \emptyset	0.3673	0.3068
2.59	<i>green, log-rgbb</i> / \emptyset	0.3537	0.2967
2.62	<i>green, log-rs</i> / \emptyset	0.3457	0.2719
2.65	<i>green, chr-rg</i> / \emptyset	0.3547	0.2459
2.68	<i>green, chr-bg</i> / \emptyset	0.3456	0.2535
2.71	<i>green, log-bg</i> / \emptyset	0.3463	0.2612
2.74	<i>green, pix-lbp-var</i> / \emptyset	0.4113	0.2775
2.77	<i>green, satint</i> / \emptyset	0.3488	0.2487
2.102	<i>green, red-z</i> / \emptyset	0.3426	0.2584
2.105	<i>green, green-z</i> / \emptyset	0.3550	0.2668
2.108	<i>green, blue-z</i> / \emptyset	0.3879	0.2382
2.111	<i>cos(hue), sin(hue), red</i> / \emptyset	0.3088	0.2795
2.115	<i>cos(hue), sin(hue), blue</i> / \emptyset	0.3317	0.2801
2.117	<i>cos(hue), sin(hue), sat</i> / \emptyset	0.3305	0.3040
2.119	<i>cos(hue), sin(hue), value</i> / \emptyset	0.3575	0.2796
2.121	<i>cos(hue), sin(hue), kl2</i> / \emptyset	0.3220	0.3103
2.123	<i>cos(hue), sin(hue), ciex</i> / \emptyset	0.4275	0.2852
2.125	<i>cos(hue), sin(hue), ciey</i> / \emptyset	0.3397	0.2855
2.127	<i>cos(hue), sin(hue), ciez</i> / \emptyset	0.3559	0.2792
2.129	<i>cos(hue), sin(hue), ciel</i> / \emptyset	0.3632	0.2766
2.131	<i>cos(hue), sin(hue), cieb</i> / \emptyset	0.3355	0.3007
2.133	<i>cos(hue), sin(hue), cieu</i> / \emptyset	0.3342	0.3118
2.135	<i>cos(hue), sin(hue), ciev</i> / \emptyset	0.3209	0.2925
2.137	<i>cos(hue), sin(hue), log-rgbb</i> / \emptyset	0.3320	0.2976
2.139	<i>cos(hue), sin(hue), log-rs</i> / \emptyset	0.3661	0.3211
2.141	<i>cos(hue), sin(hue), chr-rg</i> / \emptyset	0.3586	0.3108
2.143	<i>cos(hue), sin(hue), chr-bg</i> / \emptyset	0.3660	0.3033
2.145	<i>cos(hue), sin(hue), log-bg</i> / \emptyset	0.3259	0.2985
2.147	<i>cos(hue), sin(hue), pix-lbp-var</i> / \emptyset	0.4407	0.2816
2.149	<i>cos(hue), sin(hue), satint</i> / \emptyset	0.3557	0.2771

Table B.2: Results of the second step of the Stepwise Forward Selection for the selection of the best parameter sets. Pareto-optimal parameter sets are marked in green.

Number	Input channels (R_1) / Input channels (R_2)	M	F
2.151	$\cos(\text{hue}), \sin(\text{hue}), \text{red-z} / \emptyset$	0.3343	0.2779
2.153	$\cos(\text{hue}), \sin(\text{hue}), \text{green-z} / \emptyset$	0.3474	0.2808
2.155	$\cos(\text{hue}), \sin(\text{hue}), \text{blue-z} / \emptyset$	0.4252	0.2776
2.157	$\text{ciex}, \text{red} / \emptyset$	0.3509	0.2608
2.160	$\text{ciex}, \text{blue} / \emptyset$	0.3801	0.2560
2.162	$\text{ciex}, \cos(\text{hue}) / \emptyset$	0.3482	0.3041
2.164	$\text{ciex}, \sin(\text{hue}) / \emptyset$	0.3683	0.2902
2.166	$\text{ciex}, \text{ciex} \cdot \cos(\text{hue}), \text{ciex} \cdot \sin(\text{hue}) / \emptyset$	0.3604	0.2804
2.168	$\text{ciex}, \text{ciex} \cdot \cos(\text{hue}) / \emptyset$	0.3873	0.2883
2.170	$\text{ciex}, \text{ciex} \cdot \sin(\text{hue}) / \emptyset$	0.4118	0.2977
2.174	$\text{ciex}, (1 - \text{ciex}) \cdot \cos(\text{hue}) / \emptyset$	0.3906	0.2856
2.176	$\text{ciex}, (1 - \text{ciex}) \cdot \sin(\text{hue}) / \emptyset$	0.4475	0.2876
2.178	$\text{ciex}, \text{sat} / \emptyset$	0.3471	0.2834
2.180	$\text{ciex}, \text{value} / \emptyset$	0.3264	0.2630
2.182	$\text{ciex}, \text{kl2} / \emptyset$	0.3433	0.3069
2.184	$\text{ciex}, \text{ciey} / \emptyset$	0.3131	0.2521
2.186	$\text{ciex}, \text{ciez} / \emptyset$	0.3405	0.2593
2.188	$\text{ciex}, \text{ciel} / \emptyset$	0.3857	0.2673
2.190	$\text{ciex}, \text{cieb} / \emptyset$	0.3534	0.3080
2.192	$\text{ciex}, \text{cieu} / \emptyset$	0.3691	0.2746
2.194	$\text{ciex}, \text{ciev} / \emptyset$	0.3487	0.3046
2.196	$\text{ciex}, \log\text{-rgbb} / \emptyset$	0.3290	0.2985
2.198	$\text{ciex}, \log\text{-rs} / \emptyset$	0.3627	0.2772
2.200	$\text{ciex}, \text{chr-rg} / \emptyset$	0.3851	0.2578
2.202	$\text{ciex}, \text{chr-bg} / \emptyset$	0.3125	0.2648
2.204	$\text{ciex}, \log\text{-bg} / \emptyset$	0.3485	0.2704
2.206	$\text{ciex}, \text{pix-lbp-var} / \emptyset$	0.3847	0.2825
2.208	$\text{ciex}, \text{satint} / \emptyset$	0.2977	0.2614
2.210	$\text{ciex}, \text{red-z} / \emptyset$	0.3580	0.2630
2.212	$\text{ciex}, \text{green-z} / \emptyset$	0.3614	0.2710
2.214	$\text{ciex}, \text{blue-z} / \emptyset$	0.4279	0.2377
2.216	$\text{ciez}, \text{red} / \emptyset$	0.3477	0.2530
2.219	$\text{ciez}, \text{blue} / \emptyset$	0.3590	0.2579
2.221	$\text{ciez}, \cos(\text{hue}) / \emptyset$	0.3686	0.3040
2.223	$\text{ciez}, \sin(\text{hue}) / \emptyset$	0.3816	0.3045
2.225	$\text{ciez}, \text{ciez} \cdot \cos(\text{hue}), \text{ciez} \cdot \sin(\text{hue}) / \emptyset$	0.3459	0.2749
2.227	$\text{ciez}, \text{ciez} \cdot \cos(\text{hue}) / \emptyset$	0.3645	0.2764
2.229	$\text{ciez}, \text{ciez} \cdot \sin(\text{hue}) / \emptyset$	0.4032	0.2856
2.231	$\text{ciez}, (1 - \text{ciez}) \cdot \cos(\text{hue}), (1 - \text{ciez}) \cdot \sin(\text{hue}) / \emptyset$	0.3662	0.2731
2.233	$\text{ciez}, (1 - \text{ciez}) \cdot \cos(\text{hue}) / \emptyset$	0.3822	0.2785
2.235	$\text{ciez}, (1 - \text{ciez}) \cdot \sin(\text{hue}) / \emptyset$	0.3921	0.2852
2.237	$\text{ciez}, \text{sat} / \emptyset$	0.3176	0.2641
2.239	$\text{ciez}, \text{value} / \emptyset$	0.3614	0.2521
2.241	$\text{ciez}, \text{kl2} / \emptyset$	0.3351	0.2939
2.243	$\text{ciez}, \text{ciey} / \emptyset$	0.3493	0.2531
2.245	$\text{ciez}, \text{ciel} / \emptyset$	0.4221	0.2488
2.247	$\text{ciez}, \text{cieb} / \emptyset$	0.3264	0.3029
2.249	$\text{ciez}, \text{cieu} / \emptyset$	0.3465	0.2596
2.251	$\text{ciez}, \text{ciev} / \emptyset$	0.3797	0.2933

Table B.2: Results of the second step of the Stepwise Forward Selection for the selection of the best parameter sets. Pareto-optimal parameter sets are marked in green.

Number	Input channels (R_1) / Input channels (R_2)	M	F
2.253	<i>ciez, log-rgbb</i> / \emptyset	0.3272	0.2822
2.255	<i>ciez, log-rs</i> / \emptyset	0.3279	0.2511
2.257	<i>ciez, chr-rg</i> / \emptyset	0.3433	0.2515
2.259	<i>ciez, chr-bg</i> / \emptyset	0.3273	0.2664
2.261	<i>ciez, log-bg</i> / \emptyset	0.3611	0.2621
2.263	<i>ciez, pix-lbp-var</i> / \emptyset	0.4383	0.2640
2.265	<i>ciez, satint</i> / \emptyset	0.3361	0.2561
2.267	<i>ciez, red-z</i> / \emptyset	0.3387	0.2547
2.269	<i>ciez, green-z</i> / \emptyset	0.3397	0.2526
2.271	<i>ciez, blue-z</i> / \emptyset	0.4070	0.2521
2.273	<i>value, red</i> / \emptyset	0.3477	0.2628
2.276	<i>value, blue</i> / \emptyset	0.3538	0.2504
2.278	<i>value, cos(hue)</i> / \emptyset	0.3708	0.3051
2.280	<i>value, sin(hue)</i> / \emptyset	0.3799	0.2956
2.282	<i>value, value · cos(hue), value · sin(hue)</i> / \emptyset	0.3705	0.2759
2.284	<i>value, value · cos(hue)</i> / \emptyset	0.3631	0.2850
2.286	<i>value, value · sin(hue)</i> / \emptyset	0.4125	0.2889
2.288	<i>value, (1 - value) · cos(hue), (1 - value) · sin(hue)</i> / \emptyset	0.3956	0.2835
2.290	<i>value, (1 - value) · cos(hue)</i> / \emptyset	0.3712	0.2862
2.292	<i>value, (1 - value) · sin(hue)</i> / \emptyset	0.3916	0.2795
2.294	<i>value, sat</i> / \emptyset	0.3137	0.2736
2.296	<i>value, kl2</i> / \emptyset	0.3528	0.3105
2.298	<i>value, ciey</i> / \emptyset	0.3437	0.2607
2.300	<i>value, ciel</i> / \emptyset	0.4172	0.2529
2.302	<i>value, cieb</i> / \emptyset	0.3895	0.3043
2.304	<i>value, cieu</i> / \emptyset	0.3686	0.2671
2.306	<i>value, cieν</i> / \emptyset	0.3238	0.3064
2.308	<i>value, log-rgbb</i> / \emptyset	0.3477	0.2976
2.310	<i>value, log-rs</i> / \emptyset	0.3585	0.2676
2.312	<i>value, chr-rg</i> / \emptyset	0.3358	0.2511
2.314	<i>value, chr-bg</i> / \emptyset	0.3408	0.2631
2.316	<i>value, log-bg</i> / \emptyset	0.3209	0.2595
2.318	<i>value, pix-lbp-var</i> / \emptyset	0.4199	0.2722
2.320	<i>value, satint</i> / \emptyset	0.3493	0.2631
2.322	<i>value, red-z</i> / \emptyset	0.3465	0.2614
2.324	<i>value, green-z</i> / \emptyset	0.3465	0.2584
2.326	<i>value, blue-z</i> / \emptyset	0.4370	0.2522
2.328	<i>ciey, red</i> / \emptyset	0.3667	0.2601
2.331	<i>ciey, blue</i> / \emptyset	0.3241	0.2551
2.333	<i>ciey, cos(hue)</i> / \emptyset	0.3583	0.3025
2.335	<i>ciey, sin(hue)</i> / \emptyset	0.3457	0.2913
2.337	<i>ciey, ciey · cos(hue), ciey · sin(hue)</i> / \emptyset	0.3854	0.2777
2.339	<i>ciey, ciey · cos(hue)</i> / \emptyset	0.3522	0.2895
2.341	<i>ciey, ciey · sin(hue)</i> / \emptyset	0.4109	0.3077
2.343	<i>ciey, (1 - ciey) · cos(hue), (1 - ciey) · sin(hue)</i> / \emptyset	0.3751	0.2931
2.345	<i>ciey, (1 - ciey) · cos(hue)</i> / \emptyset	0.3932	0.2866
2.347	<i>ciey, (1 - ciey) · sin(hue)</i> / \emptyset	0.3888	0.2865
2.349	<i>ciey, sat</i> / \emptyset	0.3511	0.2880
2.351	<i>ciey, kl2</i> / \emptyset	0.3346	0.3069

Table B.2: Results of the second step of the Stepwise Forward Selection for the selection of the best parameter sets. Pareto-optimal parameter sets are marked in green.

Number	Input channels (R_1) / Input channels (R_2)	M	F
2.353	<i>ciey, ciel</i> / \emptyset	0.4139	0.2611
2.355	<i>ciey, cie b</i> / \emptyset	0.3627	0.2963
2.357	<i>ciey, cie u</i> / \emptyset	0.3626	0.2697
2.359	<i>ciey, cie v</i> / \emptyset	0.3559	0.3101
2.361	<i>ciey, log-rgbb</i> / \emptyset	0.3279	0.2917
2.363	<i>ciey, log-rs</i> / \emptyset	0.3330	0.2796
2.365	<i>ciey, chr-rg</i> / \emptyset	0.3554	0.2582
2.367	<i>ciey, chr-bg</i> / \emptyset	0.3537	0.2572
2.369	<i>ciey, log-bg</i> / \emptyset	0.3617	0.2597
2.371	<i>ciey, pix-lbp-var</i> / \emptyset	0.3854	0.2797
2.373	<i>ciey, satint</i> / \emptyset	0.3204	0.2541
2.375	<i>ciey, red-z</i> / \emptyset	0.3523	0.2576
2.377	<i>ciey, green-z</i> / \emptyset	0.3256	0.2704
2.379	<i>ciey, blue-z</i> / \emptyset	0.4316	0.2505
2.381	<i>satint, red</i> / \emptyset	0.3294	0.2547
2.384	<i>satint, blue</i> / \emptyset	0.3385	0.2534
2.386	<i>satint, cos(hue)</i> / \emptyset	0.3503	0.3054
2.388	<i>satint, sin(hue)</i> / \emptyset	0.3583	0.3003
2.390	<i>satint, satint · cos(hue), satint · sin(hue)</i> / \emptyset	0.3445	0.2890
2.392	<i>satint, satint · cos(hue)</i> / \emptyset	0.3455	0.2903
2.394	<i>satint, satint · sin(hue)</i> / \emptyset	0.4004	0.2908
2.398	<i>satint, (1 – satint) · cos(hue)</i> / \emptyset	0.3601	0.2749
2.400	<i>satint, (1 – satint) · sin(hue)</i> / \emptyset	0.3772	0.2968
2.402	<i>satint, sat</i> / \emptyset	0.3586	0.2726
2.404	<i>satint, kl2</i> / \emptyset	0.3608	0.3058
2.406	<i>satint, ciel</i> / \emptyset	0.3840	0.2502
2.408	<i>satint, cie b</i> / \emptyset	0.3764	0.3166
2.410	<i>satint, cie u</i> / \emptyset	0.3620	0.2663
2.412	<i>satint, cie v</i> / \emptyset	0.3483	0.3025
2.414	<i>satint, log-rgbb</i> / \emptyset	0.3509	0.2955
2.416	<i>satint, log-rs</i> / \emptyset	0.3737	0.2563
2.418	<i>satint, chr-rg</i> / \emptyset	0.3367	0.2428
2.420	<i>satint, chr-bg</i> / \emptyset	0.3082	0.2628
2.422	<i>satint, log-bg</i> / \emptyset	0.3479	0.2673
2.424	<i>satint, pix-lbp-var</i> / \emptyset	0.4428	0.2846
2.426	<i>satint, red-z</i> / \emptyset	0.3386	0.2506
2.428	<i>satint, green-z</i> / \emptyset	0.3400	0.2503
2.430	<i>satint, blue-z</i> / \emptyset	0.3914	0.2472
2.432	<i>green-z, red</i> / \emptyset	0.3299	0.2617
2.435	<i>green-z, blue</i> / \emptyset	0.2931	0.2562
2.437	<i>green-z, cos(hue)</i> / \emptyset	0.3598	0.3070
2.439	<i>green-z, sin(hue)</i> / \emptyset	0.3631	0.2874
2.441	<i>green-z, green-z · cos(hue), green-z · sin(hue)</i> / \emptyset	0.3491	0.2817
2.443	<i>green-z, green-z · cos(hue)</i> / \emptyset	0.3359	0.2992
2.445	<i>green-z, green-z · sin(hue)</i> / \emptyset	0.4249	0.3003
2.447	<i>green-z, (1 – green-z) · cos(hue), (1 – green-z) · sin(hue)</i> / \emptyset	0.3709	0.2964
2.449	<i>green-z, (1 – green-z) · cos(hue)</i> / \emptyset	0.3871	0.2917
2.451	<i>green-z, (1 – green-z) · sin(hue)</i> / \emptyset	0.3785	0.2867

Table B.2: Results of the second step of the Stepwise Forward Selection for the selection of the best parameter sets. Pareto-optimal parameter sets are marked in green.

Number	Input channels (R_1) / Input channels (R_2)	M	F
2.453	<i>green-z, sat</i> / \emptyset	0.3351	0.2949
2.455	<i>green-z, kl2</i> / \emptyset	0.3080	0.3024
2.457	<i>green-z, ciel</i> / \emptyset	0.3817	0.2649
2.459	<i>green-z, cie b</i> / \emptyset	0.3631	0.3027
2.461	<i>green-z, cie u</i> / \emptyset	0.3587	0.2724
2.463	<i>green-z, cie v</i> / \emptyset	0.3687	0.3007
2.465	<i>green-z, log-rgbb</i> / \emptyset	0.3815	0.3039
2.467	<i>green-z, log-rs</i> / \emptyset	0.3362	0.2706
2.469	<i>green-z, chr-rg</i> / \emptyset	0.3432	0.2455
2.471	<i>green-z, chr-bg</i> / \emptyset	0.3518	0.2552
2.473	<i>green-z, log-bg</i> / \emptyset	0.3515	0.2578
2.477	<i>green-z, red-z</i> / \emptyset	0.3331	0.2631
2.479	<i>green-z, blue-z</i> / \emptyset	0.4047	0.2353
2.481	<i>blue-z, red</i> / \emptyset	0.3434	0.2439
2.484	<i>blue-z, blue</i> / \emptyset	0.3224	0.2616
2.486	<i>blue-z, cos(hue)</i> / \emptyset	0.3555	0.3070
2.488	<i>blue-z, sin(hue)</i> / \emptyset	0.3707	0.3016
2.490	<i>blue-z, blue-z · cos(hue), blue-z · sin(hue)</i> / \emptyset	0.4228	0.2771
2.492	<i>blue-z, blue-z · cos(hue)</i> / \emptyset	0.4605	0.2777
2.494	<i>blue-z, blue-z · sin(hue)</i> / \emptyset	0.3875	0.2916
2.496	<i>blue-z, (1 - blue-z) · cos(hue), (1 - blue-z) · sin(hue)</i> / \emptyset	0.3407	0.2789
2.498	<i>blue-z, (1 - blue-z) · cos(hue)</i> / \emptyset	0.3729	0.2798
2.500	<i>blue-z, (1 - blue-z) · sin(hue)</i> / \emptyset	0.3949	0.3000
2.502	<i>blue-z, sat</i> / \emptyset	0.3120	0.2615
2.504	<i>blue-z, kl2</i> / \emptyset	0.3413	0.3080
2.506	<i>blue-z, ciel</i> / \emptyset	0.4188	0.2476
2.508	<i>blue-z, cie b</i> / \emptyset	0.3917	0.3074
2.510	<i>blue-z, cie u</i> / \emptyset	0.3489	0.2554
2.512	<i>blue-z, cie v</i> / \emptyset	0.3559	0.3046
2.514	<i>blue-z, log-rgbb</i> / \emptyset	0.3402	0.2854
2.516	<i>blue-z, log-rs</i> / \emptyset	0.3692	0.2556
2.518	<i>blue-z, chr-rg</i> / \emptyset	0.3646	0.2451
2.520	<i>blue-z, chr-bg</i> / \emptyset	0.3513	0.2663
2.522	<i>blue-z, log-bg</i> / \emptyset	0.3401	0.2554
2.524	<i>blue-z, pix-lbp-var</i> / \emptyset	0.4184	0.2687
2.526	<i>blue-z, red-z</i> / \emptyset	0.3282	0.2538
2.528.1	<i>cos(hue), sin(hue)</i> / <i>red</i>	0.4846	0.2378
2.528.2	<i>cos(hue), sin(hue)</i> / <i>green</i>	0.4541	0.2288
2.528.3	<i>cos(hue), sin(hue)</i> / <i>blue</i>	0.4443	0.2254
2.528.4	<i>cos(hue), sin(hue)</i> / <i>sat</i>	0.5360	0.2739
2.528.5	<i>cos(hue), sin(hue)</i> / <i>value</i>	0.4361	0.2231
2.528.6	<i>cos(hue), sin(hue)</i> / <i>kl2</i>	0.4782	0.2639
2.528.7	<i>cos(hue), sin(hue)</i> / <i>ci x</i>	0.4748	0.2281
2.528.8	<i>cos(hue), sin(hue)</i> / <i>cie y</i>	0.4592	0.2227
2.528.9	<i>cos(hue), sin(hue)</i> / <i>cie z</i>	0.4325	0.2261
2.528.10	<i>cos(hue), sin(hue)</i> / <i>ciel</i>	0.4821	0.2297
2.528.11	<i>cos(hue), sin(hue)</i> / <i>cie b</i>	0.4608	0.2518
2.528.12	<i>cos(hue), sin(hue)</i> / <i>cie u</i>	0.4813	0.2684
2.528.13	<i>cos(hue), sin(hue)</i> / <i>cie v</i>	0.4606	0.2556

Table B.2: Results of the second step of the Stepwise Forward Selection for the selection of the best parameter sets. Pareto-optimal parameter sets are marked in green.

Number	Input channels (R_1) / Input channels (R_2)	M	F
2.528.14	$\cos(\text{hue}), \sin(\text{hue}) / \log\text{-rgbb}$	0.4445	0.2426
2.528.15	$\cos(\text{hue}), \sin(\text{hue}) / \log\text{-rs}$	0.4672	0.2619
2.528.16	$\cos(\text{hue}), \sin(\text{hue}) / \text{chr}\text{-rg}$	0.5143	0.2785
2.528.17	$\cos(\text{hue}), \sin(\text{hue}) / \text{chr}\text{-bg}$	0.5355	0.2607
2.528.18	$\cos(\text{hue}), \sin(\text{hue}) / \log\text{-bg}$	0.5085	0.2426
2.528.19	$\cos(\text{hue}), \sin(\text{hue}) / \text{pix}\text{-lbp}\text{-var}$	0.5885	0.3799
2.528.20	$\cos(\text{hue}), \sin(\text{hue}) / \text{satint}$	0.4226	0.2211
2.528.21	$\cos(\text{hue}), \sin(\text{hue}) / \text{reg}\text{-lbp}\text{-riu}2\text{m}$	0.7300	0.5332
2.528.22	$\cos(\text{hue}), \sin(\text{hue}) / \text{reg}\text{-lbp}\text{-riu}2\text{f}$	0.6941	0.4656
2.528.24	$\cos(\text{hue}), \sin(\text{hue}) / \text{reg}\text{-ent}\text{-e}$	0.7017	0.5613
2.528.25	$\cos(\text{hue}), \sin(\text{hue}) / \text{reg}\text{-ent}\text{-c}$	0.5542	0.3948
2.528.27	$\cos(\text{hue}), \sin(\text{hue}) / \cos(\text{reg}\text{-soh}\text{-th})$	0.7380	0.5859
2.528.28	$\cos(\text{hue}), \sin(\text{hue}) / \sin(\text{reg}\text{-soh}\text{-th})$	0.6807	0.5658
2.528.29	$\cos(\text{hue}), \sin(\text{hue}) / \text{reg}\text{-soh}\text{-g}$	0.7047	0.5096
2.528.30	$\cos(\text{hue}), \sin(\text{hue}) / \cos(\text{reg}\text{-soh}\text{-m}), \sin(\text{reg}\text{-soh}\text{-m})$	0.6290	0.5096
2.528.31	$\cos(\text{hue}), \sin(\text{hue}) / \cos(\text{reg}\text{-soh}\text{-m})$	0.6898	0.5261
2.528.32	$\cos(\text{hue}), \sin(\text{hue}) / \sin(\text{reg}\text{-soh}\text{-m})$	0.6324	0.5478
2.528.33	$\cos(\text{hue}), \sin(\text{hue}) / \text{reg}\text{-soh}\text{-v}$	0.7288	0.6086
2.528.37	$\cos(\text{hue}), \sin(\text{hue}) / \text{reg}\text{-aurelie}\text{-p1}$	0.3106	0.7080
2.528.41	$\cos(\text{hue}), \sin(\text{hue}) / \text{reg}\text{-aurelie}\text{-p2}$	0.5317	0.7088
2.528.43	$\cos(\text{hue}), \sin(\text{hue}) / \text{red}\text{-z}$	0.4792	0.2495
2.528.44	$\cos(\text{hue}), \sin(\text{hue}) / \text{green}\text{-z}$	0.4046	0.2298
2.528.45	$\cos(\text{hue}), \sin(\text{hue}) / \text{blue}\text{-z}$	0.4645	0.2275
2.530.1	$\text{satint} / \text{red}$	0.4456	0.2190
2.530.2	$\text{satint} / \text{green}$	0.4401	0.2159
2.530.3	$\text{satint} / \text{blue}$	0.4271	0.2000
2.530.4	$\text{satint} / \cos(\text{hue}), \sin(\text{hue})$	0.4912	0.2423
2.530.5	$\text{satint} / \cos(\text{hue})$	0.4756	0.2651
2.530.6	$\text{satint} / \sin(\text{hue})$	0.4104	0.2470
2.530.7	$\text{satint} / \text{sat}$	0.4617	0.2612
2.530.8	$\text{satint} / \text{value}$	0.4099	0.1978
2.530.9	$\text{satint} / \text{kl}2$	0.4827	0.2682
2.530.10	$\text{satint} / \text{ciex}$	0.4163	0.2020
2.530.11	$\text{satint} / \text{ciey}$	0.4293	0.2122
2.530.12	$\text{satint} / \text{ciez}$	0.3948	0.1923
2.530.13	$\text{satint} / \text{ciel}$	0.4127	0.2197
2.530.14	$\text{satint} / \text{cieb}$	0.4722	0.2302
2.530.15	$\text{satint} / \text{cieu}$	0.5106	0.2774
2.530.16	$\text{satint} / \text{ciev}$	0.4424	0.2348
2.530.17	$\text{satint} / \log\text{-rgbb}$	0.3980	0.2443
2.530.18	$\text{satint} / \log\text{-rs}$	0.4762	0.2675
2.530.19	$\text{satint} / \text{chr}\text{-rg}$	0.5127	0.2758
2.530.20	$\text{satint} / \text{chr}\text{-bg}$	0.4680	0.2481
2.530.21	$\text{satint} / \log\text{-bg}$	0.4869	0.2413
2.530.22	$\text{satint} / \text{pix}\text{-lbp}\text{-var}$	0.4907	0.3612
2.530.23	$\text{satint} / \text{reg}\text{-lbp}\text{-riu}2\text{m}$	0.7311	0.5269
2.530.24	$\text{satint} / \text{reg}\text{-lbp}\text{-riu}2\text{f}$	0.6566	0.4115
2.530.26	$\text{satint} / \text{reg}\text{-ent}\text{-e}$	0.7119	0.5392
2.530.27	$\text{satint} / \text{reg}\text{-ent}\text{-c}$	0.5863	0.3852

Table B.2: Results of the second step of the Stepwise Forward Selection for the selection of the best parameter sets. Pareto-optimal parameter sets are marked in green.

Number	Input channels (R_1) / Input channels (R_2)	M	F
2.530.31	<i>satint</i> / <i>reg-soh-g</i>	0.7190	0.5315
2.530.32	<i>satint</i> / $\cos(\text{reg-soh-m}), \sin(\text{reg-soh-m})$	0.6362	0.4920
2.530.33	<i>satint</i> / $\cos(\text{reg-soh-m})$	0.6451	0.4935
2.530.34	<i>satint</i> / $\sin(\text{reg-soh-m})$	0.6653	0.5375
2.530.35	<i>satint</i> / <i>reg-soh-v</i>	0.7988	0.6737
2.530.39	<i>satint</i> / <i>reg-aurelie-p1</i>	0.5429	0.7019
2.530.43	<i>satint</i> / <i>reg-aurelie-p2</i>	0.5347	0.7047
2.530.45	<i>satint</i> / <i>red-z</i>	0.4597	0.2125
2.530.46	<i>satint</i> / <i>green-z</i>	0.4398	0.2129
2.530.47	<i>satint</i> / <i>blue-z</i>	0.4440	0.1956
2.532.1	<i>value</i> / <i>red</i>	0.4531	0.2191
2.532.2	<i>value</i> / <i>green</i>	0.4295	0.2098
2.532.3	<i>value</i> / <i>blue</i>	0.4554	0.1975
2.532.4	<i>value</i> / $\cos(\text{hue}), \sin(\text{hue})$	0.4923	0.2355
2.532.5	<i>value</i> / $\cos(\text{hue})$	0.5030	0.2570
2.532.6	<i>value</i> / $\sin(\text{hue})$	0.4367	0.2411
2.532.7	<i>value</i> / <i>sat</i>	0.4953	0.2725
2.532.8	<i>value</i> / <i>kl2</i>	0.4480	0.2483
2.532.9	<i>value</i> / <i>ciex</i>	0.3958	0.2003
2.532.10	<i>value</i> / <i>ciex</i>	0.4301	0.1945
2.532.11	<i>value</i> / <i>ciez</i>	0.4269	0.1996
2.532.12	<i>value</i> / <i>ciel</i>	0.4245	0.2114
2.532.13	<i>value</i> / <i>cieb</i>	0.4975	0.2335
2.532.14	<i>value</i> / <i>cieu</i>	0.5402	0.2743
2.532.15	<i>value</i> / <i>ciev</i>	0.4848	0.2510
2.532.16	<i>value</i> / <i>log-rgbb</i>	0.4451	0.2323
2.532.17	<i>value</i> / <i>log-rs</i>	0.4780	0.2631
2.532.18	<i>value</i> / <i>chr-rg</i>	0.5538	0.2809
2.532.19	<i>value</i> / <i>chr-bg</i>	0.5035	0.2341
2.532.20	<i>value</i> / <i>log-bg</i>	0.5091	0.2199
2.532.21	<i>value</i> / <i>pix-lbp-var</i>	0.5024	0.3355
2.532.22	<i>value</i> / <i>satint</i>	0.4124	0.1980
2.532.23	<i>value</i> / <i>reg-lbp-riu2m</i>	0.7665	0.5439
2.532.24	<i>value</i> / <i>reg-lbp-riu2f</i>	0.6754	0.4502
2.532.26	<i>value</i> / <i>reg-ent-e</i>	0.7042	0.5577
2.532.27	<i>value</i> / <i>reg-ent-c</i>	0.5747	0.3797
2.532.28	<i>value</i> / $\cos(\text{reg-soh-th}), \sin(\text{reg-soh-th})$	0.6726	0.5271
2.532.29	<i>value</i> / $\cos(\text{reg-soh-th})$	0.7367	0.5659
2.532.30	<i>value</i> / $\sin(\text{reg-soh-th})$	0.6572	0.5361
2.532.31	<i>value</i> / <i>reg-soh-g</i>	0.7598	0.5559
2.532.32	<i>value</i> / $\cos(\text{reg-soh-m}), \sin(\text{reg-soh-m})$	0.6335	0.5026
2.532.33	<i>value</i> / $\cos(\text{reg-soh-m})$	0.6730	0.5220
2.532.34	<i>value</i> / $\sin(\text{reg-soh-m})$	0.6348	0.5441
2.532.35	<i>value</i> / <i>reg-soh-v</i>	0.8148	0.6974
2.532.39	<i>value</i> / <i>reg-aurelie-p1</i>	0.5624	0.7022
2.532.43	<i>value</i> / <i>reg-aurelie-p2</i>	0.5611	0.7057
2.532.45	<i>value</i> / <i>red-z</i>	0.4884	0.2227
2.532.46	<i>value</i> / <i>green-z</i>	0.4457	0.2040
2.532.47	<i>value</i> / <i>blue-z</i>	0.4165	0.1946

Table B.2: Results of the second step of the Stepwise Forward Selection for the selection of the best parameter sets. Pareto-optimal parameter sets are marked in green.

Number	Input channels (R_1) / Input channels (R_2)	M	F
2.534.1	<i>blue-z / red</i>	0.4542	0.2190
2.534.2	<i>blue-z / green</i>	0.4044	0.2067
2.534.3	<i>blue-z / blue</i>	0.4113	0.2168
2.534.4	<i>blue-z / cos(hue), sin(hue)</i>	0.4400	0.2311
2.534.5	<i>blue-z / cos(hue)</i>	0.4643	0.2553
2.534.6	<i>blue-z / sin(hue)</i>	0.3969	0.2335
2.534.7	<i>blue-z / sat</i>	0.4226	0.2790
2.534.8	<i>blue-z / value</i>	0.4217	0.2004
2.534.9	<i>blue-z / kl2</i>	0.4711	0.2533
2.534.10	<i>blue-z / ciex</i>	0.4100	0.2114
2.534.11	<i>blue-z / ciey</i>	0.4007	0.2154
2.534.12	<i>blue-z / ciez</i>	0.4141	0.2081
2.534.13	<i>blue-z / ciel</i>	0.4284	0.2078
2.534.14	<i>blue-z / cieb</i>	0.4433	0.2267
2.534.15	<i>blue-z / cieu</i>	0.4994	0.2670
2.534.16	<i>blue-z / ciev</i>	0.5084	0.2284
2.534.17	<i>blue-z / log-rgbb</i>	0.4030	0.2478
2.534.18	<i>blue-z / log-rs</i>	0.4815	0.2572
2.534.19	<i>blue-z / chr-rg</i>	0.5166	0.2824
2.534.20	<i>blue-z / chr-bg</i>	0.4455	0.2419
2.534.21	<i>blue-z / log-bg</i>	0.4824	0.2413
2.534.22	<i>blue-z / pix-lbp-var</i>	0.4796	0.3527
2.534.23	<i>blue-z / satint</i>	0.4163	0.2114
2.534.24	<i>blue-z / reg-lbp-riu2m</i>	0.7571	0.5338
2.534.25	<i>blue-z / reg-lbp-riu2f</i>	0.6572	0.4318
2.534.27	<i>blue-z / reg-ent-e</i>	0.6588	0.5213
2.534.28	<i>blue-z / reg-ent-c</i>	0.5935	0.4044
2.534.32	<i>blue-z / reg-soh-g</i>	0.7217	0.5280
2.534.33	<i>blue-z / cos(reg-soh-m), sin(reg-soh-m)</i>	0.6307	0.4977
2.534.34	<i>blue-z / cos(reg-soh-m)</i>	0.6445	0.5035
2.534.35	<i>blue-z / sin(reg-soh-m)</i>	0.6594	0.5321
2.534.36	<i>blue-z / reg-soh-v</i>	0.7781	0.6835
2.534.46	<i>blue-z / red-z</i>	0.4296	0.2187
2.534.47	<i>blue-z / green-z</i>	0.3907	0.2042
2.536.1	<i>green-z / red</i>	0.4480	0.2186
2.536.2	<i>green-z / green</i>	0.4153	0.2041
2.536.3	<i>green-z / blue</i>	0.4424	0.1920
2.536.4	<i>green-z / cos(hue), sin(hue)</i>	0.4569	0.2418
2.536.5	<i>green-z / cos(hue)</i>	0.4586	0.2661
2.536.6	<i>green-z / sin(hue)</i>	0.4625	0.2654
2.536.7	<i>green-z / sat</i>	0.4476	0.2699
2.536.8	<i>green-z / value</i>	0.4389	0.2044
2.536.9	<i>green-z / kl2</i>	0.4670	0.2565
2.536.10	<i>green-z / ciex</i>	0.4350	0.2040
2.536.11	<i>green-z / ciey</i>	0.4638	0.1988
2.536.12	<i>green-z / ciez</i>	0.4397	0.1946
2.536.13	<i>green-z / ciel</i>	0.4592	0.2152
2.536.14	<i>green-z / cieb</i>	0.4723	0.2375
2.536.15	<i>green-z / cieu</i>	0.5357	0.2754

Table B.2: Results of the second step of the Stepwise Forward Selection for the selection of the best parameter sets. Pareto-optimal parameter sets are marked in green.

Number	Input channels (R_1) / Input channels (R_2)	M	F
2.536.16	<i>green-z / ciev</i>	0.4608	0.2486
2.536.17	<i>green-z / log-rgbb</i>	0.5018	0.2384
2.536.18	<i>green-z / log-rs</i>	0.5249	0.2640
2.536.19	<i>green-z / chr-rg</i>	0.5217	0.2709
2.536.20	<i>green-z / chr-bg</i>	0.5060	0.2465
2.536.21	<i>green-z / log-bg</i>	0.4866	0.2366
2.536.22	<i>green-z / pix-lbp-var</i>	0.5136	0.3566
2.536.23	<i>green-z / satint</i>	0.4400	0.2044
2.536.24	<i>green-z / reg-lbp-riu2m</i>	0.7542	0.5331
2.536.25	<i>green-z / reg-lbp-riu2f</i>	0.6771	0.4311
2.536.27	<i>green-z / reg-ent-e</i>	0.6797	0.5539
2.536.28	<i>green-z / reg-ent-c</i>	0.5599	0.3770
2.536.29	<i>green-z / cos(reg-soh-th), sin(reg-soh-th)</i>	0.7002	0.5487
2.536.31	<i>green-z / sin(reg-soh-th)</i>	0.6737	0.5572
2.536.32	<i>green-z / reg-soh-g</i>	0.7532	0.5400
2.536.33	<i>green-z / cos(reg-soh-m), sin(reg-soh-m)</i>	0.6358	0.4991
2.536.34	<i>green-z / cos(reg-soh-m)</i>	0.6495	0.5004
2.536.35	<i>green-z / sin(reg-soh-m)</i>	0.6471	0.5301
2.536.36	<i>green-z / reg-soh-v</i>	0.7937	0.6705
2.536.40	<i>green-z / reg-aurelie-p1</i>	0.4304	0.6933
2.536.46	<i>green-z / red-z</i>	0.4627	0.2258
2.536.47	<i>green-z / blue-z</i>	0.4340	0.1997
2.538.1	<i>ciey / red</i>	0.4527	0.2110
2.538.2	<i>ciey / green</i>	0.4384	0.2014
2.538.3	<i>ciey / blue</i>	0.4360	0.1828
2.538.4	<i>ciey / cos(hue), sin(hue)</i>	0.4364	0.2342
2.538.5	<i>ciey / cos(hue)</i>	0.4506	0.2702
2.538.6	<i>ciey / sin(hue)</i>	0.4162	0.2540
2.538.7	<i>ciey / sat</i>	0.5085	0.2425
2.538.8	<i>ciey / value</i>	0.4453	0.1898
2.538.9	<i>ciey / kl2</i>	0.4543	0.2478
2.538.10	<i>ciey / ciex</i>	0.4567	0.1971
2.538.11	<i>ciey / ciez</i>	0.4228	0.1820
2.538.12	<i>ciey / ciel</i>	0.4118	0.2088
2.538.13	<i>ciey / cie b</i>	0.5002	0.2316
2.538.14	<i>ciey / cie u</i>	0.5412	0.2908
2.538.15	<i>ciey / cie v</i>	0.4681	0.2340
2.538.16	<i>ciey / log-rgbb</i>	0.4637	0.2277
2.538.17	<i>ciey / log-rs</i>	0.4866	0.2643
2.538.18	<i>ciey / chr-rg</i>	0.5446	0.2818
2.538.19	<i>ciey / chr-bg</i>	0.4892	0.2421
2.538.20	<i>ciey / log-bg</i>	0.4749	0.2315
2.538.21	<i>ciey / pix-lbp-var</i>	0.5441	0.3760
2.538.22	<i>ciey / satint</i>	0.4461	0.1940
2.538.23	<i>ciey / reg-lbp-riu2m</i>	0.7572	0.5410
2.538.24	<i>ciey / reg-lbp-riu2f</i>	0.6724	0.4491
2.538.26	<i>ciey / reg-ent-e</i>	0.7081	0.5617
2.538.27	<i>ciey / reg-ent-c</i>	0.5717	0.3927
2.538.29	<i>ciey / cos(reg-soh-th)</i>	0.7344	0.5544

Table B.2: Results of the second step of the Stepwise Forward Selection for the selection of the best parameter sets. Pareto-optimal parameter sets are marked in green.

Number	Input channels (R_1) / Input channels (R_2)	M	F
2.538.31	<i>ciey</i> / <i>reg-soh-g</i>	0.7482	0.5435
2.538.32	<i>ciey</i> / $\cos(\text{reg-soh-m}), \sin(\text{reg-soh-m})$	0.6412	0.4970
2.538.33	<i>ciey</i> / $\cos(\text{reg-soh-m})$	0.6603	0.5116
2.538.34	<i>ciey</i> / $\sin(\text{reg-soh-m})$	0.6723	0.5502
2.538.35	<i>ciey</i> / <i>reg-soh-v</i>	0.7671	0.6636
2.538.39	<i>ciey</i> / <i>reg-aurelie-p1</i>	0.6445	0.7004
2.538.45	<i>ciey</i> / <i>red-z</i>	0.4946	0.2155
2.538.46	<i>ciey</i> / <i>green-z</i>	0.4213	0.2026
2.538.47	<i>ciey</i> / <i>blue-z</i>	0.4270	0.1892
2.540.1	<i>ciez</i> / <i>red</i>	0.4534	0.2136
2.540.2	<i>ciez</i> / <i>green</i>	0.4095	0.2003
2.540.3	<i>ciez</i> / <i>blue</i>	0.4555	0.1951
2.540.4	<i>ciez</i> / $\cos(\text{hue}), \sin(\text{hue})$	0.4210	0.2175
2.540.5	<i>ciez</i> / $\cos(\text{hue})$	0.5007	0.2608
2.540.6	<i>ciez</i> / $\sin(\text{hue})$	0.4137	0.2400
2.540.7	<i>ciez</i> / <i>sat</i>	0.4698	0.2435
2.540.8	<i>ciez</i> / <i>value</i>	0.4537	0.2004
2.540.9	<i>ciez</i> / <i>kl2</i>	0.4897	0.2636
2.540.10	<i>ciez</i> / <i>ciex</i>	0.4823	0.2019
2.540.11	<i>ciez</i> / <i>ciey</i>	0.4188	0.2102
2.540.12	<i>ciez</i> / <i>ciel</i>	0.4216	0.2156
2.540.13	<i>ciez</i> / <i>cieb</i>	0.4649	0.2347
2.540.14	<i>ciez</i> / <i>cieu</i>	0.4871	0.2739
2.540.15	<i>ciez</i> / <i>ciev</i>	0.4503	0.2289
2.540.16	<i>ciez</i> / <i>log-rgbb</i>	0.3996	0.2344
2.540.17	<i>ciez</i> / <i>log-rs</i>	0.4902	0.2639
2.540.18	<i>ciez</i> / <i>chr-rg</i>	0.5321	0.2679
2.540.19	<i>ciez</i> / <i>chr-bg</i>	0.4695	0.2248
2.540.20	<i>ciez</i> / <i>log-bg</i>	0.4665	0.2353
2.540.21	<i>ciez</i> / <i>pix-lbp-var</i>	0.5170	0.3553
2.540.22	<i>ciez</i> / <i>satint</i>	0.4186	0.1920
2.540.23	<i>ciez</i> / <i>reg-lbp-riu2m</i>	0.7456	0.5233
2.540.24	<i>ciez</i> / <i>reg-lbp-riu2f</i>	0.6777	0.4406
2.540.26	<i>ciez</i> / <i>reg-ent-e</i>	0.6867	0.5399
2.540.27	<i>ciez</i> / <i>reg-ent-c</i>	0.5659	0.3734
2.540.31	<i>ciez</i> / <i>reg-soh-g</i>	0.7440	0.5414
2.540.32	<i>ciez</i> / $\cos(\text{reg-soh-m}), \sin(\text{reg-soh-m})$	0.6218	0.4843
2.540.33	<i>ciez</i> / $\cos(\text{reg-soh-m})$	0.6497	0.4834
2.540.34	<i>ciez</i> / $\sin(\text{reg-soh-m})$	0.6474	0.5347
2.540.35	<i>ciez</i> / <i>reg-soh-v</i>	0.7794	0.6550
2.540.39	<i>ciez</i> / <i>reg-aurelie-p1</i>	0.5736	0.7057
2.540.43	<i>ciez</i> / <i>reg-aurelie-p2</i>	0.5220	0.7031
2.540.45	<i>ciez</i> / <i>red-z</i>	0.4531	0.2168
2.540.46	<i>ciez</i> / <i>green-z</i>	0.4482	0.2110
2.540.47	<i>ciez</i> / <i>blue-z</i>	0.4478	0.1968
2.542.1	<i>ciex</i> / <i>red</i>	0.4472	0.2121
2.542.2	<i>ciex</i> / <i>green</i>	0.4137	0.1990
2.542.3	<i>ciex</i> / <i>blue</i>	0.4291	0.1961
2.542.4	<i>ciex</i> / $\cos(\text{hue}), \sin(\text{hue})$	0.4710	0.2192

Table B.2: Results of the second step of the Stepwise Forward Selection for the selection of the best parameter sets. Pareto-optimal parameter sets are marked in green.

Number	Input channels (R_1) / Input channels (R_2)	M	F
2.542.5	<i>ciex</i> / $\cos(\text{hue})$	0.4734	0.2549
2.542.6	<i>ciex</i> / $\sin(\text{hue})$	0.4328	0.2348
2.542.7	<i>ciex</i> / <i>sat</i>	0.4800	0.2454
2.542.8	<i>ciex</i> / <i>value</i>	0.3983	0.1999
2.542.9	<i>ciex</i> / <i>kl2</i>	0.4981	0.2486
2.542.10	<i>ciex</i> / <i>ciex</i>	0.4191	0.1937
2.542.11	<i>ciex</i> / <i>ciez</i>	0.4108	0.1937
2.542.12	<i>ciex</i> / <i>ciel</i>	0.4243	0.2111
2.542.13	<i>ciex</i> / <i>cieb</i>	0.4773	0.2219
2.542.14	<i>ciex</i> / <i>cieu</i>	0.5397	0.2807
2.542.15	<i>ciex</i> / <i>ciev</i>	0.4615	0.2308
2.542.16	<i>ciex</i> / <i>log-rgbb</i>	0.4168	0.2321
2.542.17	<i>ciex</i> / <i>log-rs</i>	0.4797	0.2515
2.542.18	<i>ciex</i> / <i>chr-rg</i>	0.5125	0.2729
2.542.19	<i>ciex</i> / <i>chr-bg</i>	0.4584	0.2324
2.542.20	<i>ciex</i> / <i>log-bg</i>	0.5143	0.2117
2.542.21	<i>ciex</i> / <i>pix-lbp-var</i>	0.5342	0.3540
2.542.22	<i>ciex</i> / <i>satint</i>	0.4145	0.1919
2.542.23	<i>ciex</i> / <i>reg-lbp-riu2m</i>	0.7633	0.5471
2.542.24	<i>ciex</i> / <i>reg-lbp-riu2f</i>	0.6725	0.4364
2.542.26	<i>ciex</i> / <i>reg-ent-e</i>	0.6797	0.5347
2.542.27	<i>ciex</i> / <i>reg-ent-c</i>	0.5841	0.3818
2.542.28	<i>ciex</i> / $\cos(\text{reg-soh-th}), \sin(\text{reg-soh-th})$	0.7052	0.5454
2.542.29	<i>ciex</i> / $\cos(\text{reg-soh-th})$	0.7256	0.5453
2.542.30	<i>ciex</i> / $\sin(\text{reg-soh-th})$	0.6774	0.5576
2.542.31	<i>ciex</i> / <i>reg-soh-g</i>	0.7462	0.5454
2.542.32	<i>ciex</i> / $\cos(\text{reg-soh-m}), \sin(\text{reg-soh-m})$	0.6406	0.5137
2.542.33	<i>ciex</i> / $\cos(\text{reg-soh-m})$	0.6511	0.5136
2.542.34	<i>ciex</i> / $\sin(\text{reg-soh-m})$	0.6464	0.5235
2.542.35	<i>ciex</i> / <i>reg-soh-v</i>	0.7811	0.6699
2.542.45	<i>ciex</i> / <i>red-z</i>	0.4166	0.2052
2.542.46	<i>ciex</i> / <i>green-z</i>	0.4211	0.1976
2.542.47	<i>ciex</i> / <i>blue-z</i>	0.4208	0.1823
2.544.1	<i>green</i> / <i>red</i>	0.4425	0.2174
2.544.2	<i>green</i> / <i>blue</i>	0.4277	0.1961
2.544.3	<i>green</i> / $\cos(\text{hue}), \sin(\text{hue})$	0.4572	0.2380
2.544.4	<i>green</i> / $\cos(\text{hue})$	0.4600	0.2758
2.544.5	<i>green</i> / $\sin(\text{hue})$	0.4601	0.2650
2.544.6	<i>green</i> / <i>sat</i>	0.4567	0.2499
2.544.7	<i>green</i> / <i>value</i>	0.4285	0.2025
2.544.8	<i>green</i> / <i>kl2</i>	0.4664	0.2611
2.544.9	<i>green</i> / <i>ciex</i>	0.4574	0.2013
2.544.10	<i>green</i> / <i>ciex</i>	0.4276	0.2084
2.544.11	<i>green</i> / <i>ciez</i>	0.4072	0.1944
2.544.12	<i>green</i> / <i>ciel</i>	0.4352	0.2092
2.544.13	<i>green</i> / <i>cieb</i>	0.5000	0.2427
2.544.14	<i>green</i> / <i>cieu</i>	0.4768	0.2710
2.544.15	<i>green</i> / <i>ciev</i>	0.4827	0.2550

Table B.2: Results of the second step of the Stepwise Forward Selection for the selection of the best parameter sets. Pareto-optimal parameter sets are marked in green.

Number	Input channels (R_1) / Input channels (R_2)	M	F
2.544.16	<i>green</i> / <i>log-rgbb</i>	0.4674	0.2332
2.544.17	<i>green</i> / <i>log-rs</i>	0.4849	0.2717
2.544.18	<i>green</i> / <i>chr-rg</i>	0.4818	0.2718
2.544.19	<i>green</i> / <i>chr-bg</i>	0.4739	0.2420
2.544.20	<i>green</i> / <i>log-bg</i>	0.4732	0.2271
2.544.21	<i>green</i> / <i>pix-lbp-var</i>	0.5117	0.3719
2.544.22	<i>green</i> / <i>satint</i>	0.4453	0.1968
2.544.23	<i>green</i> / <i>reg-lbp-riu2m</i>	0.7626	0.5418
2.544.24	<i>green</i> / <i>reg-lbp-riu2f</i>	0.6860	0.4409
2.544.26	<i>green</i> / <i>reg-ent-e</i>	0.6782	0.5415
2.544.27	<i>green</i> / <i>reg-ent-c</i>	0.5755	0.3885
2.544.28	<i>green</i> / $\cos(\text{reg-soh-th}), \sin(\text{reg-soh-th})$	0.6785	0.5589
2.544.30	<i>green</i> / $\sin(\text{reg-soh-th})$	0.6634	0.5600
2.544.31	<i>green</i> / <i>reg-soh-g</i>	0.7487	0.5507
2.544.32	<i>green</i> / $\cos(\text{reg-soh-m}), \sin(\text{reg-soh-m})$	0.6547	0.5048
2.544.33	<i>green</i> / $\cos(\text{reg-soh-m})$	0.6569	0.5019
2.544.34	<i>green</i> / $\sin(\text{reg-soh-m})$	0.6374	0.5245
2.544.35	<i>green</i> / <i>reg-soh-v</i>	0.7721	0.6471
2.544.43	<i>green</i> / <i>reg-aurelie-p2</i>	0.5665	0.7071
2.544.45	<i>green</i> / <i>red-z</i>	0.4340	0.2153
2.544.46	<i>green</i> / <i>green-z</i>	0.4395	0.2050
2.544.47	<i>green</i> / <i>blue-z</i>	0.4571	0.1967

Table B.2: Results of the second step of the Stepwise Forward Selection for the selection of the best parameter sets. Pareto-optimal parameter sets are marked in green.

Number	Input channels (R_1) / Input channels (R_2)	M	F
3.1	<i>green, ciez, red</i> / \emptyset	0.3379	0.2493
3.5	<i>green, ciez, blue</i> / \emptyset	0.3178	0.2529
3.8	<i>green, ciez, cos(hue), sin(hue)</i> / \emptyset	0.3373	0.2785
3.10	<i>green, ciez, cos(hue)</i> / \emptyset	0.3728	0.2855
3.12	<i>green, ciez, sin(hue)</i> / \emptyset	0.3624	0.2819
3.14	<i>green, ciez, green · cos(hue), green · sin(hue)</i> / \emptyset	0.3335	0.2778
3.16	<i>green, ciez, green · cos(hue)</i> / \emptyset	0.3504	0.2611
3.18	<i>green, ciez, green · sin(hue)</i> / \emptyset	0.3560	0.2632
3.20	<i>green, ciez, (1 - green) · cos(hue), (1 - green) · sin(hue)</i> / \emptyset	0.3502	0.2857
3.22	<i>green, ciez, (1 - green) · cos(hue)</i> / \emptyset	0.3295	0.2566
3.24	<i>green, ciez, (1 - green) · sin(hue)</i> / \emptyset	0.3483	0.2738
3.26	<i>green, ciez, ciez · cos(hue), ciez · sin(hue)</i> / \emptyset	0.3687	0.2934
3.28	<i>green, ciez, ciez · cos(hue)</i> / \emptyset	0.3197	0.2563
3.30	<i>green, ciez, ciez · sin(hue)</i> / \emptyset	0.3711	0.2762
3.32	<i>green, ciez, (1 - ciez) · cos(hue), (1 - ciez) · sin(hue)</i> / \emptyset	0.3604	0.2826
3.34	<i>green, ciez, (1 - ciez) · cos(hue)</i> / \emptyset	0.3558	0.2692
3.36	<i>green, ciez, (1 - ciez) · sin(hue)</i> / \emptyset	0.3081	0.2891
3.41	<i>green, ciez, sat</i> / \emptyset	0.3540	0.2577

Table B.3: Results of the third step of the Stepwise Forward Selection for the selection of the best parameter sets. Pareto-optimal parameter sets are marked in green. Some close-to-optimal parameter sets, marked in yellow, are also retained for the next step.

Number	Input channels (R_1) / Input channels (R_2)	M	F
3.44	<i>green, ciez, value</i> / \emptyset	0.3575	0.2561
3.47	<i>green, ciez, kl2</i> / \emptyset	0.3668	0.2670
3.50	<i>green, ciez, ciex</i> / \emptyset	0.4355	0.2467
3.53	<i>green, ciez, ciej</i> / \emptyset	0.3563	0.2597
3.56	<i>green, ciez, ciel</i> / \emptyset	0.4299	0.2459
3.59	<i>green, ciez, cieb</i> / \emptyset	0.3400	0.2806
3.62	<i>green, ciez, cieu</i> / \emptyset	0.3201	0.2461
3.65	<i>green, ciez, ciev</i> / \emptyset	0.3148	0.2749
3.68	<i>green, ciez, log-rgbb</i> / \emptyset	0.3045	0.2635
3.71	<i>green, ciez, log-rs</i> / \emptyset	0.3583	0.2492
3.74	<i>green, ciez, chr-rg</i> / \emptyset	0.3554	0.2516
3.77	<i>green, ciez, chr-bg</i> / \emptyset	0.3284	0.2567
3.80	<i>green, ciez, log-bg</i> / \emptyset	0.3359	0.2494
3.83	<i>green, ciez, pix-lbp-var</i> / \emptyset	0.3986	0.2793
3.86	<i>green, ciez, satint</i> / \emptyset	0.3140	0.2542
3.91	<i>green, ciez, red-z</i> / \emptyset	0.3598	0.2534
3.94	<i>green, ciez, green-z</i> / \emptyset	0.3169	0.2532
3.97	<i>green, ciez, blue-z</i> / \emptyset	0.4501	0.2417
3.100	<i>green, blue-z, red</i> / \emptyset	0.4001	0.2382
3.103	<i>green, blue-z, blue</i> / \emptyset	0.3577	0.2452
3.105	<i>green, blue-z, cos(hue), sin(hue)</i> / \emptyset	0.4269	0.2813
3.107	<i>green, blue-z, cos(hue)</i> / \emptyset	0.4430	0.2877
3.109	<i>green, blue-z, sin(hue)</i> / \emptyset	0.3868	0.2633
3.111	<i>green, blue-z, green · cos(hue), green · sin(hue)</i> / \emptyset	0.3694	0.2837
3.113	<i>green, blue-z, green · cos(hue)</i> / \emptyset	0.3740	0.2526
3.115	<i>green, blue-z, green · sin(hue)</i> / \emptyset	0.4098	0.2527
3.117	<i>green, blue-z, (1 - green) · cos(hue), (1 - green) · sin(hue)</i> / \emptyset	0.4131	0.2859
3.119	<i>green, blue-z, (1 - green) · cos(hue)</i> / \emptyset	0.4169	0.2601
3.121	<i>green, blue-z, (1 - green) · sin(hue)</i> / \emptyset	0.4133	0.2678
3.123	<i>green, blue-z, blue-z · cos(hue), blue-z · sin(hue)</i> / \emptyset	0.3914	0.2707
3.125	<i>green, blue-z, blue-z · cos(hue)</i> / \emptyset	0.4299	0.2512
3.127	<i>green, blue-z, blue-z · sin(hue)</i> / \emptyset	0.4388	0.2583
3.129	<i>green, blue-z, (1 - blue-z) · cos(hue), (1 - blue-z) · sin(hue)</i> / \emptyset	0.3998	0.2713
3.131	<i>green, blue-z, (1 - blue-z) · cos(hue)</i> / \emptyset	0.4100	0.2789
3.133	<i>green, blue-z, (1 - blue-z) · sin(hue)</i> / \emptyset	0.4325	0.2826
3.135	<i>green, blue-z, sat</i> / \emptyset	0.4474	0.2436
3.137	<i>green, blue-z, value</i> / \emptyset	0.4503	0.2314
3.139	<i>green, blue-z, kl2</i> / \emptyset	0.3928	0.2645
3.141	<i>green, blue-z, ciex</i> / \emptyset	0.4237	0.2381
3.143	<i>green, blue-z, ciej</i> / \emptyset	0.3885	0.2461
3.146	<i>green, blue-z, ciel</i> / \emptyset	0.4201	0.2410
3.148	<i>green, blue-z, cieb</i> / \emptyset	0.4330	0.2768
3.150	<i>green, blue-z, cieu</i> / \emptyset	0.4130	0.2347
3.152	<i>green, blue-z, ciev</i> / \emptyset	0.3712	0.2638
3.154	<i>green, blue-z, log-rgbb</i> / \emptyset	0.4118	0.2560
3.156	<i>green, blue-z, log-rs</i> / \emptyset	0.4477	0.2269
3.158	<i>green, blue-z, chr-rg</i> / \emptyset	0.4221	0.2450
3.160	<i>green, blue-z, chr-bg</i> / \emptyset	0.4407	0.2397
3.162	<i>green, blue-z, log-bg</i> / \emptyset	0.3778	0.2459

Table B.3: Results of the third step of the Stepwise Forward Selection for the selection of the best parameter sets. Pareto-optimal parameter sets are marked in green. Some close-to-optimal parameter sets, marked in yellow, are also retained for the next step.

Number	Input channels (R_1) / Input channels (R_2)	M	F
3.164	<i>green, blue-z, pix-lbp-var</i> / \emptyset	0.4696	0.2811
3.166	<i>green, blue-z, satint</i> / \emptyset	0.4001	0.2416
3.168	<i>green, blue-z, red-z</i> / \emptyset	0.3976	0.2528
3.170	<i>green, blue-z, green-z</i> / \emptyset	0.4568	0.2435
3.172	<i>ciex, ciey, red</i> / \emptyset	0.3271	0.2669
3.175	<i>ciex, ciey, green</i> / \emptyset	0.3218	0.2639
3.178	<i>ciex, ciey, blue</i> / \emptyset	0.3409	0.2570
3.180	<i>ciex, ciey, cos(hue), sin(hue)</i> / \emptyset	0.3727	0.2778
3.182	<i>ciex, ciey, cos(hue)</i> / \emptyset	0.3454	0.2886
3.184	<i>ciex, ciey, sin(hue)</i> / \emptyset	0.3626	0.2848
3.186	<i>ciex, ciey, ciex · cos(hue), ciex · sin(hue)</i> / \emptyset	0.3643	0.2948
3.188	<i>ciex, ciey, ciex · cos(hue)</i> / \emptyset	0.3588	0.2607
3.190	<i>ciex, ciey, ciex · sin(hue)</i> / \emptyset	0.3599	0.2770
3.192	<i>ciex, ciey, (1 - ciex) · cos(hue), (1 - ciex) · sin(hue)</i> / \emptyset	0.3677	0.2874
3.194	<i>ciex, ciey, (1 - ciex) · cos(hue)</i> / \emptyset	0.3625	0.2769
3.196	<i>ciex, ciey, (1 - ciex) · sin(hue)</i> / \emptyset	0.3633	0.2815
3.198	<i>ciex, ciey, ciey · cos(hue), ciey · sin(hue)</i> / \emptyset	0.3355	0.2909
3.200	<i>ciex, ciey, ciey · cos(hue)</i> / \emptyset	0.3414	0.2673
3.202	<i>ciex, ciey, ciey · sin(hue)</i> / \emptyset	0.3515	0.2780
3.204	<i>ciex, ciey, (1 - ciey) · cos(hue), (1 - ciey) · sin(hue)</i> / \emptyset	0.3334	0.2837
3.206	<i>ciex, ciey, (1 - ciey) · cos(hue)</i> / \emptyset	0.3409	0.2643
3.208	<i>ciex, ciey, (1 - ciey) · sin(hue)</i> / \emptyset	0.3522	0.2767
3.210	<i>ciex, ciey, sat</i> / \emptyset	0.3523	0.2664
3.212	<i>ciex, ciey, value</i> / \emptyset	0.3452	0.2589
3.214	<i>ciex, ciey, kl2</i> / \emptyset	0.3288	0.2803
3.216	<i>ciex, ciey, ciez</i> / \emptyset	0.3379	0.2526
3.218	<i>ciex, ciey, ciel</i> / \emptyset	0.3875	0.2642
3.220	<i>ciex, ciey, cieb</i> / \emptyset	0.3475	0.2867
3.222	<i>ciex, ciey, cieu</i> / \emptyset	0.3503	0.2552
3.224	<i>ciex, ciey, ciev</i> / \emptyset	0.3271	0.2911
3.226	<i>ciex, ciey, log-rghb</i> / \emptyset	0.3505	0.2751
3.228	<i>ciex, ciey, log-rs</i> / \emptyset	0.3367	0.2568
3.230	<i>ciex, ciey, chr-rg</i> / \emptyset	0.3371	0.2606
3.232	<i>ciex, ciey, chr-bg</i> / \emptyset	0.3413	0.2607
3.234	<i>ciex, ciey, log-bg</i> / \emptyset	0.3391	0.2574
3.236	<i>ciex, ciey, pix-lbp-var</i> / \emptyset	0.4105	0.2829
3.238	<i>ciex, ciey, satint</i> / \emptyset	0.3229	0.2620
3.240	<i>ciex, ciey, red-z</i> / \emptyset	0.3578	0.2542
3.242	<i>ciex, ciey, green-z</i> / \emptyset	0.3467	0.2612
3.244	<i>ciex, ciey, blue-z</i> / \emptyset	0.4198	0.2347
3.246	<i>ciez, log-rs, red</i> / \emptyset	0.3900	0.2529
3.249	<i>ciez, log-rs, blue</i> / \emptyset	0.3459	0.2393
3.251	<i>ciez, log-rs, cos(hue), sin(hue)</i> / \emptyset	0.3434	0.2783
3.253	<i>ciez, log-rs, cos(hue)</i> / \emptyset	0.3650	0.3008
3.255	<i>ciez, log-rs, sin(hue)</i> / \emptyset	0.3327	0.2814
3.257	<i>ciez, log-rs, ciez · cos(hue), ciez · sin(hue)</i> / \emptyset	0.3493	0.2781
3.259	<i>ciez, log-rs, ciez · cos(hue)</i> / \emptyset	0.3266	0.2762
3.261	<i>ciez, log-rs, ciez · sin(hue)</i> / \emptyset	0.3816	0.2948
3.263	<i>ciez, log-rs, (1 - ciez) · cos(hue), (1 - ciez) · sin(hue)</i> / \emptyset	0.3137	0.2733

Table B.3: Results of the third step of the Stepwise Forward Selection for the selection of the best parameter sets. Pareto-optimal parameter sets are marked in green. Some close-to-optimal parameter sets, marked in yellow, are also retained for the next step.

Number	Input channels (R_1) / Input channels (R_2)	M	F
3.265	<i>ciez, log-rs, (1 - ciezs) · cos(hue) / ∅</i>	0.3518	0.2665
3.267	<i>ciez, log-rs, (1 - ciezs) · sin(hue) / ∅</i>	0.3545	0.2904
3.269	<i>ciez, log-rs, log-rs · cos(hue), log-rs · sin(hue) / ∅</i>	0.4558	0.2427
3.271	<i>ciez, log-rs, log-rs · cos(hue) / ∅</i>	0.3528	0.2479
3.273	<i>ciez, log-rs, log-rs · sin(hue) / ∅</i>	0.3987	0.2646
3.275	<i>ciez, log-rs, (1 - log-rs) · cos(hue), (1 - log-rs) · sin(hue) / ∅</i>	0.4246	0.2885
3.277	<i>ciez, log-rs, (1 - log-rs) · cos(hue) / ∅</i>	0.4001	0.2769
3.279	<i>ciez, log-rs, (1 - log-rs) · sin(hue) / ∅</i>	0.3207	0.2652
3.281	<i>ciez, log-rs, sat / ∅</i>	0.3364	0.2630
3.283	<i>ciez, log-rs, value / ∅</i>	0.3389	0.2508
3.285	<i>ciez, log-rs, kl2 / ∅</i>	0.3292	0.2992
3.287	<i>ciez, log-rs, ciex / ∅</i>	0.4226	0.2299
3.289	<i>ciez, log-rs, ciey / ∅</i>	0.3524	0.2452
3.291	<i>ciez, log-rs, ciel / ∅</i>	0.4149	0.2400
3.293	<i>ciez, log-rs, ciebs / ∅</i>	0.3045	0.2861
3.295	<i>ciez, log-rs, cieus / ∅</i>	0.3695	0.2750
3.297	<i>ciez, log-rs, cievs / ∅</i>	0.3611	0.2992
3.299	<i>ciez, log-rs, log-rgbb / ∅</i>	0.3404	0.2862
3.301	<i>ciez, log-rs, chr-rg / ∅</i>	0.3641	0.2676
3.303	<i>ciez, log-rs, chr-bg / ∅</i>	0.3682	0.2470
3.305	<i>ciez, log-rs, log-bg / ∅</i>	0.3185	0.2564
3.307	<i>ciez, log-rs, pix-lbp-var / ∅</i>	0.3939	0.2568
3.309	<i>ciez, log-rs, satint / ∅</i>	0.3711	0.2395
3.311	<i>ciez, log-rs, red-z / ∅</i>	0.3641	0.2539
3.313	<i>ciez, log-rs, green-z / ∅</i>	0.3468	0.2446
3.315	<i>ciez, log-rs, blue-z / ∅</i>	0.4052	0.2375
3.317	<i>satint, chr-rg, red / ∅</i>	0.3567	0.2616
3.320	<i>satint, chr-rg, green / ∅</i>	0.3338	0.2489
3.322	<i>satint, chr-rg, blue / ∅</i>	0.3799	0.2448
3.324	<i>satint, chr-rg, cos(hue), sin(hue) / ∅</i>	0.3348	0.2804
3.326	<i>satint, chr-rg, cos(hue) / ∅</i>	0.3626	0.3100
3.328	<i>satint, chr-rg, sin(hue) / ∅</i>	0.3286	0.2938
3.330	<i>satint, chr-rg, satint · cos(hue), satint · sin(hue) / ∅</i>	0.3715	0.2880
3.332	<i>satint, chr-rg, satint · cos(hue) / ∅</i>	0.3550	0.2733
3.334	<i>satint, chr-rg, satint · sin(hue) / ∅</i>	0.3805	0.2860
3.336	<i>satint, chr-rg, (1 - satint) · cos(hue), (1 - satint) · sin(hue) / ∅</i>	0.3624	0.2807
3.338	<i>satint, chr-rg, (1 - satint) · cos(hue) / ∅</i>	0.3488	0.2753
3.340	<i>satint, chr-rg, (1 - satint) · sin(hue) / ∅</i>	0.3784	0.3015
3.342	<i>satint, chr-rg, chr-rg · cos(hue), chr-rg · sin(hue) / ∅</i>	0.4323	0.2398
3.344	<i>satint, chr-rg, chr-rg · cos(hue) / ∅</i>	0.3528	0.2476
3.346	<i>satint, chr-rg, chr-rg · sin(hue) / ∅</i>	0.4382	0.2579
3.348	<i>satint, chr-rg, (1 - chr-rg) · cos(hue), (1 - chr-rg) · sin(hue) / ∅</i>	0.3432	0.2766
3.350	<i>satint, chr-rg, (1 - chr-rg) · cos(hue) / ∅</i>	0.4052	0.2777
3.352	<i>satint, chr-rg, (1 - chr-rg) · sin(hue) / ∅</i>	0.3422	0.2662
3.354	<i>satint, chr-rg, sat / ∅</i>	0.3608	0.2492
3.356	<i>satint, chr-rg, value / ∅</i>	0.3540	0.2470
3.358	<i>satint, chr-rg, kl2 / ∅</i>	0.3398	0.3002
3.360	<i>satint, chr-rg, ciex / ∅</i>	0.4110	0.2401
3.362	<i>satint, chr-rg, ciey / ∅</i>	0.3510	0.2531

Table B.3: Results of the third step of the Stepwise Forward Selection for the selection of the best parameter sets. Pareto-optimal parameter sets are marked in green. Some close-to-optimal parameter sets, marked in yellow, are also retained for the next step.

Number	Input channels (R_1) / Input channels (R_2)	M	F
3.364	<i>satint, chr-rg, ciez</i> / \emptyset	0.3246	0.2542
3.366	<i>satint, chr-rg, ciel</i> / \emptyset	0.4183	0.2513
3.368	<i>satint, chr-rg, cieb</i> / \emptyset	0.3512	0.2972
3.370	<i>satint, chr-rg, cieu</i> / \emptyset	0.3888	0.2798
3.372	<i>satint, chr-rg, ciev</i> / \emptyset	0.3400	0.2949
3.374	<i>satint, chr-rg, log-rgbb</i> / \emptyset	0.3375	0.2685
3.376	<i>satint, chr-rg, log-rs</i> / \emptyset	0.3781	0.2784
3.378	<i>satint, chr-rg, chr-bg</i> / \emptyset	0.3338	0.2622
3.380	<i>satint, chr-rg, log-bg</i> / \emptyset	0.3466	0.2527
3.382	<i>satint, chr-rg, pix-lbp-var</i> / \emptyset	0.4432	0.2521
3.384	<i>satint, chr-rg, red-z</i> / \emptyset	0.3441	0.2591
3.386	<i>satint, chr-rg, green-z</i> / \emptyset	0.3335	0.2520
3.388	<i>satint, chr-rg, blue-z</i> / \emptyset	0.4590	0.2447
3.390	<i>green-z, blue, red</i> / \emptyset	0.3265	0.2497
3.393	<i>green-z, blue, green</i> / \emptyset	0.3255	0.2470
3.395	<i>green-z, blue, cos(hue), sin(hue)</i> / \emptyset	0.3575	0.2742
3.397	<i>green-z, blue, cos(hue)</i> / \emptyset	0.3848	0.3015
3.399	<i>green-z, blue, sin(hue)</i> / \emptyset	0.3631	0.2701
3.401	<i>green-z, blue, green-z · cos(hue), green-z · sin(hue)</i> / \emptyset	0.3111	0.2793
3.403	<i>green-z, blue, green-z · cos(hue)</i> / \emptyset	0.3551	0.2622
3.405	<i>green-z, blue, green-z · sin(hue)</i> / \emptyset	0.3235	0.2656
3.407	<i>green-z, blue, (1 - green-z) · cos(hue), (1 - green-z) · sin(hue)</i> / \emptyset	0.3604	0.2939
3.409	<i>green-z, blue, (1 - green-z) · cos(hue)</i> / \emptyset	0.3618	0.2625
3.411	<i>green-z, blue, (1 - green-z) · sin(hue)</i> / \emptyset	0.3634	0.2838
3.413	<i>green-z, blue, blue · cos(hue), blue · sin(hue)</i> / \emptyset	0.3322	0.2841
3.415	<i>green-z, blue, blue · cos(hue)</i> / \emptyset	0.3439	0.2580
3.417	<i>green-z, blue, blue · sin(hue)</i> / \emptyset	0.3585	0.2684
3.419	<i>green-z, blue, (1 - blue) · cos(hue), (1 - blue) · sin(hue)</i> / \emptyset	0.3642	0.2869
3.421	<i>green-z, blue, (1 - blue) · cos(hue)</i> / \emptyset	0.3399	0.2875
3.423	<i>green-z, blue, (1 - blue) · sin(hue)</i> / \emptyset	0.3650	0.2991
3.425	<i>green-z, blue, sat</i> / \emptyset	0.3061	0.2551
3.427	<i>green-z, blue, value</i> / \emptyset	0.3448	0.2518
3.429	<i>green-z, blue, kl2</i> / \emptyset	0.3301	0.2726
3.431	<i>green-z, blue, ciex</i> / \emptyset	0.3893	0.2416
3.433	<i>green-z, blue, ciej</i> / \emptyset	0.3466	0.2531
3.435	<i>green-z, blue, ciez</i> / \emptyset	0.3242	0.2540
3.437	<i>green-z, blue, ciel</i> / \emptyset	0.4073	0.2592
3.439	<i>green-z, blue, cieb</i> / \emptyset	0.3409	0.2818
3.441	<i>green-z, blue, cieu</i> / \emptyset	0.3207	0.2451
3.443	<i>green-z, blue, ciev</i> / \emptyset	0.3440	0.2842
3.445	<i>green-z, blue, log-rgbb</i> / \emptyset	0.3229	0.2601
3.447	<i>green-z, blue, log-rs</i> / \emptyset	0.3182	0.2383
3.449	<i>green-z, blue, chr-rg</i> / \emptyset	0.3740	0.2526
3.451	<i>green-z, blue, chr-bg</i> / \emptyset	0.3232	0.2527
3.453	<i>green-z, blue, log-bg</i> / \emptyset	0.3533	0.2503
3.455	<i>green-z, blue, pix-lbp-var</i> / \emptyset	0.4183	0.2902
3.457	<i>green-z, blue, satint</i> / \emptyset	0.3450	0.2551
3.459	<i>green-z, blue, red-z</i> / \emptyset	0.3377	0.2498

Table B.3: Results of the third step of the Stepwise Forward Selection for the selection of the best parameter sets. Pareto-optimal parameter sets are marked in green. Some close-to-optimal parameter sets, marked in yellow, are also retained for the next step.

Number	Input channels (R_1) / Input channels (R_2)	M	F
3.461	<i>green-z, blue, blue-z</i> / \emptyset	0.4017	0.2432
3.595.1	<i>ciex / satint, red</i>	0.4311	0.1887
3.595.2	<i>ciex / satint, green</i>	0.4497	0.1953
3.595.3	<i>ciex / satint, blue</i>	0.4046	0.1909
3.595.4	<i>ciex / satint, cos(hue), sin(hue)</i>	0.4627	0.1758
3.595.5	<i>ciex / satint, cos(hue)</i>	0.4086	0.1886
3.595.6	<i>ciex / satint, sin(hue)</i>	0.4303	0.1816
3.595.7	<i>ciex / satint, satint · cos(hue), satint · sin(hue)</i>	0.4122	0.1892
3.595.8	<i>ciex / satint, satint · cos(hue)</i>	0.4326	0.1993
3.595.9	<i>ciex / satint, satint · sin(hue)</i>	0.4166	0.1994
3.595.10	<i>ciex / satint, (1 - satint) · cos(hue), (1 - satint) · sin(hue)</i>	0.4049	0.1969
3.595.11	<i>ciex / satint, (1 - satint) · cos(hue)</i>	0.4517	0.2053
3.595.12	<i>ciex / satint, (1 - satint) · sin(hue)</i>	0.4703	0.2073
3.595.13	<i>ciex / satint, sat</i>	0.3912	0.1943
3.595.14	<i>ciex / satint, value</i>	0.3877	0.1980
3.595.15	<i>ciex / satint, kl2</i>	0.4082	0.1675
3.595.16	<i>ciex / satint, ciex</i>	0.4454	0.1887
3.595.17	<i>ciex / satint, ciez</i>	0.4260	0.1895
3.595.18	<i>ciex / satint, ciel</i>	0.5135	0.1889
3.595.19	<i>ciex / satint, cie</i>	0.4016	0.1807
3.595.20	<i>ciex / satint, ciex</i>	0.4375	0.1786
3.595.21	<i>ciex / satint, ciex</i>	0.4048	0.1823
3.595.22	<i>ciex / satint, log-rgbb</i>	0.4130	0.1843
3.595.23	<i>ciex / satint, log-rs</i>	0.4420	0.1850
3.595.24	<i>ciex / satint, chr-rg</i>	0.4329	0.1902
3.595.25	<i>ciex / satint, chr-bg</i>	0.4021	0.1854
3.595.26	<i>ciex / satint, log-bg</i>	0.4177	0.1836
3.595.27	<i>ciex / satint, pix-lbp-var</i>	0.4939	0.2721
3.595.28	<i>ciex / satint, reg-ent-c</i>	0.5386	0.3329
3.595.29	<i>ciex / satint, reg-soh-g</i>	0.5949	0.3941
3.595.30	<i>ciex / satint, red-z</i>	0.4271	0.1973
3.595.31	<i>ciex / satint, green-z</i>	0.3950	0.1980
3.595.32	<i>ciex / satint, blue-z</i>	0.4796	0.1812
3.595.33	<i>ciex / blue-z, red</i>	0.4477	0.1974
3.595.34	<i>ciex / blue-z, green</i>	0.3981	0.1928
3.595.35	<i>ciex / blue-z, blue</i>	0.4040	0.1915
3.595.36	<i>ciex / blue-z, cos(hue), sin(hue)</i>	0.4670	0.1879
3.595.37	<i>ciex / blue-z, cos(hue)</i>	0.4513	0.1821
3.595.38	<i>ciex / blue-z, sin(hue)</i>	0.4148	0.1838
3.595.39	<i>ciex / blue-z, blue-z · cos(hue), blue-z · sin(hue)</i>	0.5383	0.1823
3.595.40	<i>ciex / blue-z, blue-z · cos(hue)</i>	0.5077	0.1745
3.595.41	<i>ciex / blue-z, blue-z · sin(hue)</i>	0.4004	0.1903
3.595.42	<i>ciex / blue-z, (1 - blue-z) · cos(hue), (1 - blue-z) · sin(hue)</i>	0.4102	0.2010
3.595.43	<i>ciex / blue-z, (1 - blue-z) · cos(hue)</i>	0.4099	0.1814
3.595.44	<i>ciex / blue-z, (1 - blue-z) · sin(hue)</i>	0.4316	0.2018
3.595.45	<i>ciex / blue-z, sat</i>	0.4124	0.1924
3.595.46	<i>ciex / blue-z, value</i>	0.4277	0.1985
3.595.47	<i>ciex / blue-z, kl2</i>	0.4273	0.1778

Table B.3: Results of the third step of the Stepwise Forward Selection for the selection of the best parameter sets. Pareto-optimal parameter sets are marked in green. Some close-to-optimal parameter sets, marked in yellow, are also retained for the next step.

Number	Input channels (R_1) / Input channels (R_2)	M	F
3.595.48	<i>ciex / blue-z, ciey</i>	0.4437	0.1880
3.595.49	<i>ciex / blue-z, ciez</i>	0.4036	0.1870
3.595.50	<i>ciex / blue-z, ciel</i>	0.4490	0.1854
3.595.51	<i>ciex / blue-z, cieb</i>	0.4145	0.1767
3.595.52	<i>ciex / blue-z, cieu</i>	0.4420	0.1841
3.595.53	<i>ciex / blue-z, cieo</i>	0.4347	0.1748
3.595.54	<i>ciex / blue-z, log-rgbb</i>	0.4074	0.1832
3.595.55	<i>ciex / blue-z, log-rs</i>	0.4232	0.1760
3.595.56	<i>ciex / blue-z, chr-rg</i>	0.3798	0.1808
3.595.57	<i>ciex / blue-z, chr-bg</i>	0.4308	0.1784
3.595.58	<i>ciex / blue-z, log-bg</i>	0.4143	0.1862
3.595.59	<i>ciex / blue-z, pix-lbp-var</i>	0.5170	0.2851
3.595.60	<i>ciex / blue-z, reg-ent-c</i>	0.5496	0.3253
3.595.61	<i>ciex / blue-z, reg-soh-g</i>	0.6281	0.4017
3.595.62	<i>ciex / blue-z, red-z</i>	0.4287	0.1979
3.595.63	<i>ciex / blue-z, green-z</i>	0.3661	0.1930
3.597.1	<i>satint / ciez, red</i>	0.4571	0.1980
3.597.2	<i>satint / ciez, green</i>	0.4231	0.1909
3.597.3	<i>satint / ciez, blue</i>	0.4127	0.1956
3.597.4	<i>satint / ciez, cos(hue), sin(hue)</i>	0.4856	0.2058
3.597.5	<i>satint / ciez, cos(hue)</i>	0.4062	0.2005
3.597.6	<i>satint / ciez, sin(hue)</i>	0.4128	0.1836
3.597.7	<i>satint / ciez, ciez · cos(hue), ciez · sin(hue)</i>	0.4289	0.1903
3.597.8	<i>satint / ciez, ciez · cos(hue)</i>	0.4523	0.1999
3.597.9	<i>satint / ciez, ciez · sin(hue)</i>	0.3897	0.1978
3.597.10	<i>satint / ciez, (1 - ciez) · cos(hue), (1 - ciez) · sin(hue)</i>	0.4083	0.2095
3.597.11	<i>satint / ciez, (1 - ciez) · cos(hue)</i>	0.4245	0.2051
3.597.12	<i>satint / ciez, (1 - ciez) · sin(hue)</i>	0.4012	0.2063
3.597.13	<i>satint / ciez, sat</i>	0.4064	0.1969
3.597.14	<i>satint / ciez, value</i>	0.4452	0.1909
3.597.15	<i>satint / ciez, kl2</i>	0.4227	0.1728
3.597.16	<i>satint / ciez, ciex</i>	0.5280	0.1830
3.597.17	<i>satint / ciez, ciey</i>	0.4381	0.1910
3.597.18	<i>satint / ciez, ciel</i>	0.4865	0.1901
3.597.19	<i>satint / ciez, cieb</i>	0.4801	0.1752
3.597.20	<i>satint / ciez, cieu</i>	0.4055	0.1896
3.597.21	<i>satint / ciez, cieo</i>	0.4269	0.1843
3.597.22	<i>satint / ciez, log-rgbb</i>	0.4026	0.1884
3.597.23	<i>satint / ciez, log-rs</i>	0.4326	0.1861
3.597.24	<i>satint / ciez, chr-rg</i>	0.3998	0.1960
3.597.25	<i>satint / ciez, chr-bg</i>	0.4069	0.1921
3.597.26	<i>satint / ciez, log-bg</i>	0.3490	0.1848
3.597.27	<i>satint / ciez, pix-lbp-var</i>	0.4885	0.2632
3.597.28	<i>satint / ciez, reg-ent-c</i>	0.5283	0.3100
3.597.29	<i>satint / ciez, reg-soh-g</i>	0.5643	0.3881
3.597.30	<i>satint / ciez, red-z</i>	0.4229	0.1932
3.597.31	<i>satint / ciez, green-z</i>	0.4013	0.1880
3.597.32	<i>satint / ciez, blue-z</i>	0.5011	0.1821

Table B.3: Results of the third step of the Stepwise Forward Selection for the selection of the best parameter sets. Pareto-optimal parameter sets are marked in green. Some close-to-optimal parameter sets, marked in yellow, are also retained for the next step.

Number	Input channels (R_1) / Input channels (R_2)	M	F
3.599.1	<i>satint, chr-rg / red</i>	0.4589	0.2215
3.599.2	<i>satint, chr-rg / green</i>	0.4474	0.2116
3.599.3	<i>satint, chr-rg / blue</i>	0.4503	0.2030
3.599.4	<i>satint, chr-rg / cos(hue), sin(hue)</i>	0.4556	0.2277
3.599.5	<i>satint, chr-rg / cos(hue)</i>	0.4727	0.2590
3.599.6	<i>satint, chr-rg / sin(hue)</i>	0.4094	0.2463
3.599.7	<i>satint, chr-rg / sat</i>	0.4945	0.2794
3.599.8	<i>satint, chr-rg / value</i>	0.4230	0.2094
3.599.9	<i>satint, chr-rg / kl2</i>	0.4753	0.2485
3.599.10	<i>satint, chr-rg / ciex</i>	0.4546	0.2152
3.599.11	<i>satint, chr-rg / ciey</i>	0.4071	0.2138
3.599.12	<i>satint, chr-rg / ciez</i>	0.4467	0.1973
3.599.13	<i>satint, chr-rg / ciel</i>	0.4250	0.2308
3.599.14	<i>satint, chr-rg / cieb</i>	0.4596	0.2455
3.599.15	<i>satint, chr-rg / cieu</i>	0.5265	0.2998
3.599.16	<i>satint, chr-rg / ciev</i>	0.5097	0.2532
3.599.17	<i>satint, chr-rg / log-rgbb</i>	0.4541	0.2402
3.599.18	<i>satint, chr-rg / log-rs</i>	0.5033	0.2547
3.599.19	<i>satint, chr-rg / chr-bg</i>	0.4993	0.2513
3.599.20	<i>satint, chr-rg / log-bg</i>	0.5115	0.2463
3.599.21	<i>satint, chr-rg / pix-lbp-var</i>	0.5637	0.3763
3.599.22	<i>satint, chr-rg / reg-ent-c</i>	0.5986	0.4135
3.599.23	<i>satint, chr-rg / reg-soh-g</i>	0.7498	0.5377
3.599.24	<i>satint, chr-rg / red-z</i>	0.4947	0.2177
3.599.25	<i>satint, chr-rg / green-z</i>	0.4470	0.2115
3.599.26	<i>satint, chr-rg / blue-z</i>	0.4591	0.1991
3.601.1	<i>green-z, blue / red</i>	0.4263	0.2087
3.601.2	<i>green-z, blue / green</i>	0.4012	0.2041
3.601.3	<i>green-z, blue / cos(hue), sin(hue)</i>	0.4580	0.2188
3.601.4	<i>green-z, blue / cos(hue)</i>	0.4598	0.2713
3.601.5	<i>green-z, blue / sin(hue)</i>	0.3958	0.2447
3.601.6	<i>green-z, blue / sat</i>	0.4644	0.2612
3.601.7	<i>green-z, blue / value</i>	0.4205	0.2070
3.601.8	<i>green-z, blue / kl2</i>	0.4490	0.2558
3.601.9	<i>green-z, blue / ciex</i>	0.4238	0.2050
3.601.10	<i>green-z, blue / ciey</i>	0.3931	0.1982
3.601.11	<i>green-z, blue / ciez</i>	0.4803	0.1979
3.601.12	<i>green-z, blue / ciel</i>	0.4174	0.2049
3.601.13	<i>green-z, blue / cieb</i>	0.4329	0.2165
3.601.14	<i>green-z, blue / cieu</i>	0.5002	0.2724
3.601.15	<i>green-z, blue / ciev</i>	0.4709	0.2257
3.601.16	<i>green-z, blue / log-rgbb</i>	0.4622	0.2284
3.601.17	<i>green-z, blue / log-rs</i>	0.4656	0.2570
3.601.18	<i>green-z, blue / chr-rg</i>	0.5030	0.2806
3.601.19	<i>green-z, blue / chr-bg</i>	0.4646	0.2167
3.601.20	<i>green-z, blue / log-bg</i>	0.4561	0.2226
3.601.21	<i>green-z, blue / pix-lbp-var</i>	0.5193	0.3403
3.601.22	<i>green-z, blue / satint</i>	0.4308	0.2002
3.601.23	<i>green-z, blue / reg-ent-c</i>	0.5609	0.3934

Table B.3: Results of the third step of the Stepwise Forward Selection for the selection of the best parameter sets. Pareto-optimal parameter sets are marked in green. Some close-to-optimal parameter sets, marked in yellow, are also retained for the next step.

Number	Input channels (R_1) / Input channels (R_2)	M	F
3.601.24	<i>green-z, blue / reg-soh-g</i>	0.7303	0.5271
3.601.25	<i>green-z, blue / red-z</i>	0.4845	0.2200
3.601.26	<i>green-z, blue / blue-z</i>	0.4491	0.1983
3.603.1	<i>ciez, log-rs / red</i>	0.4591	0.2130
3.603.2	<i>ciez, log-rs / green</i>	0.4692	0.2077
3.603.3	<i>ciez, log-rs / blue</i>	0.4392	0.2035
3.603.4	<i>ciez, log-rs / cos(hue), sin(hue)</i>	0.4671	0.2435
3.603.5	<i>ciez, log-rs / cos(hue)</i>	0.5017	0.2667
3.603.6	<i>ciez, log-rs / sin(hue)</i>	0.4500	0.2576
3.603.7	<i>ciez, log-rs / sat</i>	0.4923	0.2719
3.603.8	<i>ciez, log-rs / value</i>	0.4543	0.2163
3.603.9	<i>ciez, log-rs / kl2</i>	0.5021	0.2624
3.603.10	<i>ciez, log-rs / ciex</i>	0.4582	0.2148
3.603.11	<i>ciez, log-rs / ciey</i>	0.4592	0.2112
3.603.12	<i>ciez, log-rs / ciel</i>	0.4356	0.2068
3.603.13	<i>ciez, log-rs / cieb</i>	0.4699	0.2386
3.603.14	<i>ciez, log-rs / cieu</i>	0.5093	0.2735
3.603.15	<i>ciez, log-rs / cie v</i>	0.4982	0.2459
3.603.16	<i>ciez, log-rs / log-rgbb</i>	0.5032	0.2412
3.603.17	<i>ciez, log-rs / chr-rg</i>	0.5264	0.2738
3.603.18	<i>ciez, log-rs / chr-bg</i>	0.5336	0.2414
3.603.19	<i>ciez, log-rs / log-bg</i>	0.4719	0.2364
3.603.20	<i>ciez, log-rs / pix-lbp-var</i>	0.5814	0.3773
3.603.21	<i>ciez, log-rs / satint</i>	0.4483	0.2080
3.603.22	<i>ciez, log-rs / reg-ent-c</i>	0.6110	0.4019
3.603.23	<i>ciez, log-rs / reg-soh-g</i>	0.7544	0.5724
3.603.24	<i>ciez, log-rs / red-z</i>	0.4520	0.2121
3.603.25	<i>ciez, log-rs / green-z</i>	0.4654	0.2153
3.603.26	<i>ciez, log-rs / blue-z</i>	0.4452	0.2051
3.605.1	<i>blue-z / green-z, red</i>	0.4375	0.1994
3.605.2	<i>blue-z / green-z, green</i>	0.3994	0.2026
3.605.3	<i>blue-z / green-z, blue</i>	0.4180	0.1917
3.605.4	<i>blue-z / green-z, cos(hue), sin(hue)</i>	0.4716	0.2031
3.605.5	<i>blue-z / green-z, cos(hue)</i>	0.4701	0.2039
3.605.6	<i>blue-z / green-z, sin(hue)</i>	0.4082	0.1844
3.605.7	<i>blue-z / green-z, green-z · cos(hue), green-z · sin(hue)</i>	0.4148	0.2008
3.605.8	<i>blue-z / green-z, green-z · cos(hue)</i>	0.4111	0.2086
3.605.9	<i>blue-z / green-z, green-z · sin(hue)</i>	0.3611	0.2139
3.605.10	<i>blue-z / green-z, (1 - green-z) · cos(hue), (1 - green-z) · sin(hue)</i>	0.4317	0.2110
3.605.11	<i>blue-z / green-z, (1 - green-z) · cos(hue)</i>	0.4483	0.2067
3.605.12	<i>blue-z / green-z, (1 - green-z) · sin(hue)</i>	0.3873	0.2160
3.605.13	<i>blue-z / green-z, sat</i>	0.4049	0.1990
3.605.14	<i>blue-z / green-z, value</i>	0.4166	0.2097
3.605.15	<i>blue-z / green-z, kl2</i>	0.3894	0.1843
3.605.16	<i>blue-z / green-z, ciex</i>	0.5503	0.1909
3.605.17	<i>blue-z / green-z, ciey</i>	0.4070	0.2083
3.605.18	<i>blue-z / green-z, cie z</i>	0.3965	0.1953
3.605.19	<i>blue-z / green-z, ciel</i>	0.4470	0.1904
3.605.20	<i>blue-z / green-z, cie b</i>	0.4360	0.1778

Table B.3: Results of the third step of the Stepwise Forward Selection for the selection of the best parameter sets. Pareto-optimal parameter sets are marked in green. Some close-to-optimal parameter sets, marked in yellow, are also retained for the next step.

Number	Input channels (R_1) / Input channels (R_2)	M	F
3.605.21	<i>blue-z / green-z, cie</i>	0.4284	0.1749
3.605.22	<i>blue-z / green-z, cie</i>	0.4137	0.1803
3.605.23	<i>blue-z / green-z, log-rgbb</i>	0.4198	0.1951
3.605.24	<i>blue-z / green-z, log-rs</i>	0.4056	0.1794
3.605.25	<i>blue-z / green-z, chr-rg</i>	0.4300	0.1885
3.605.26	<i>blue-z / green-z, chr-bg</i>	0.4181	0.1927
3.605.27	<i>blue-z / green-z, log-bg</i>	0.3766	0.1950
3.605.28	<i>blue-z / green-z, pix-lbp-var</i>	0.4869	0.2788
3.605.29	<i>blue-z / green-z, satint</i>	0.3948	0.1888
3.605.30	<i>blue-z / green-z, reg-ent-c</i>	0.5583	0.3507
3.605.31	<i>blue-z / green-z, reg-soh-g</i>	0.6104	0.4205
3.605.32	<i>blue-z / green-z, red-z</i>	0.4197	0.1970
3.607.1	<i>green, blue-z / red</i>	0.3958	0.2147
3.607.2	<i>green, blue-z / blue</i>	0.4188	0.1936
3.607.3	<i>green, blue-z / cos(hue), sin(hue)</i>	0.4769	0.2285
3.607.4	<i>green, blue-z / cos(hue)</i>	0.4557	0.2537
3.607.5	<i>green, blue-z / sin(hue)</i>	0.3980	0.2517
3.607.6	<i>green, blue-z / sat</i>	0.4586	0.2530
3.607.7	<i>green, blue-z / value</i>	0.4122	0.1913
3.607.8	<i>green, blue-z / kl2</i>	0.4356	0.2520
3.607.9	<i>green, blue-z / ciex</i>	0.3932	0.1980
3.607.10	<i>green, blue-z / ciey</i>	0.4235	0.2011
3.607.11	<i>green, blue-z / ciez</i>	0.4353	0.1956
3.607.12	<i>green, blue-z / ciel</i>	0.4013	0.2112
3.607.13	<i>green, blue-z / cie</i>	0.4883	0.2297
3.607.14	<i>green, blue-z / cie</i>	0.5046	0.2711
3.607.15	<i>green, blue-z / cie</i>	0.4559	0.2452
3.607.16	<i>green, blue-z / log-rgbb</i>	0.4220	0.2236
3.607.17	<i>green, blue-z / log-rs</i>	0.4699	0.2510
3.607.18	<i>green, blue-z / chr-rg</i>	0.5195	0.2717
3.607.19	<i>green, blue-z / chr-bg</i>	0.4854	0.2368
3.607.20	<i>green, blue-z / log-bg</i>	0.4970	0.2156
3.607.21	<i>green, blue-z / pix-lbp-var</i>	0.5106	0.3388
3.607.22	<i>green, blue-z / satint</i>	0.4187	0.1946
3.607.23	<i>green, blue-z / reg-ent-c</i>	0.5500	0.3696
3.607.24	<i>green, blue-z / reg-soh-g</i>	0.6994	0.5297
3.607.25	<i>green, blue-z / red-z</i>	0.4303	0.2085
3.607.26	<i>green, blue-z / green-z</i>	0.3690	0.2041
3.609.1	<i>green, ciez / red</i>	0.4279	0.2050
3.609.2	<i>green, ciez / blue</i>	0.4252	0.1864
3.609.3	<i>green, ciez / cos(hue), sin(hue)</i>	0.4372	0.2367
3.609.4	<i>green, ciez / cos(hue)</i>	0.4651	0.2621
3.609.5	<i>green, ciez / sin(hue)</i>	0.4474	0.2464
3.609.6	<i>green, ciez / sat</i>	0.5192	0.2512
3.609.7	<i>green, ciez / value</i>	0.4275	0.1962
3.609.8	<i>green, ciez / kl2</i>	0.4940	0.2464
3.609.9	<i>green, ciez / ciex</i>	0.4071	0.1893
3.609.10	<i>green, ciez / ciey</i>	0.4518	0.2017
3.609.11	<i>green, ciez / ciel</i>	0.4299	0.2216

Table B.3: Results of the third step of the Stepwise Forward Selection for the selection of the best parameter sets. Pareto-optimal parameter sets are marked in green. Some close-to-optimal parameter sets, marked in yellow, are also retained for the next step.

Number	Input channels (R_1) / Input channels (R_2)	M	F
3.609.12	<i>green, ciez / cieb</i>	0.4501	0.2225
3.609.13	<i>green, ciez / cieu</i>	0.4958	0.2709
3.609.14	<i>green, ciez / ciev</i>	0.5266	0.2290
3.609.15	<i>green, ciez / log-rgbb</i>	0.4424	0.2247
3.609.16	<i>green, ciez / log-rs</i>	0.4275	0.2548
3.609.17	<i>green, ciez / chr-rg</i>	0.4933	0.2704
3.609.18	<i>green, ciez / chr-bg</i>	0.4506	0.2191
3.609.19	<i>green, ciez / log-bg</i>	0.4673	0.2159
3.609.20	<i>green, ciez / pix-lbp-var</i>	0.5353	0.3509
3.609.21	<i>green, ciez / satint</i>	0.4552	0.1940
3.609.22	<i>green, ciez / reg-ent-c</i>	0.5669	0.3870
3.609.23	<i>green, ciez / reg-soh-g</i>	0.7116	0.5136
3.609.24	<i>green, ciez / red-z</i>	0.4132	0.2058
3.609.25	<i>green, ciez / green-z</i>	0.4120	0.1951
3.609.26	<i>green, ciez / blue-z</i>	0.4493	0.1822
3.611.1	<i>ciey / ciez, red</i>	0.4697	0.1932
3.611.2	<i>ciey / ciez, green</i>	0.4173	0.1934
3.611.3	<i>ciey / ciez, blue</i>	0.4337	0.2000
3.611.4	<i>ciey / ciez, $\cos(\text{hue})$, $\sin(\text{hue})$</i>	0.4144	0.1913
3.611.5	<i>ciey / ciez, $\cos(\text{hue})$</i>	0.4359	0.2067
3.611.6	<i>ciey / ciez, $\sin(\text{hue})$</i>	0.4062	0.1773
3.611.7	<i>ciey / ciez, $\text{ciez} \cdot \cos(\text{hue})$, $\text{ciez} \cdot \sin(\text{hue})$</i>	0.4716	0.1954
3.611.8	<i>ciey / ciez, $\text{ciez} \cdot \cos(\text{hue})$</i>	0.4387	0.1928
3.611.9	<i>ciey / ciez, $\text{ciez} \cdot \sin(\text{hue})$</i>	0.4282	0.2086
3.611.10	<i>ciey / ciez, $(1 - \text{ciez}) \cdot \cos(\text{hue})$, $(1 - \text{ciez}) \cdot \sin(\text{hue})$</i>	0.4070	0.2001
3.611.11	<i>ciey / ciez, $(1 - \text{ciez}) \cdot \cos(\text{hue})$</i>	0.4512	0.2037
3.611.12	<i>ciey / ciez, $(1 - \text{ciez}) \cdot \sin(\text{hue})$</i>	0.4159	0.1998
3.611.13	<i>ciey / ciez, sat</i>	0.4126	0.1914
3.611.14	<i>ciey / ciez, value</i>	0.4681	0.1954
3.611.15	<i>ciey / ciez, kl2</i>	0.4262	0.1800
3.611.16	<i>ciey / ciez, cix</i>	0.5290	0.1780
3.611.17	<i>ciey / ciez, ciel</i>	0.4718	0.2031
3.611.18	<i>ciey / ciez, cieb</i>	0.4160	0.1819
3.611.19	<i>ciey / ciez, cieu</i>	0.4156	0.1739
3.611.20	<i>ciey / ciez, ciev</i>	0.3951	0.1823
3.611.21	<i>ciey / ciez, log-rgbb</i>	0.4385	0.1865
3.611.22	<i>ciey / ciez, log-rs</i>	0.4510	0.1760
3.611.23	<i>ciey / ciez, chr-rg</i>	0.4142	0.1888
3.611.24	<i>ciey / ciez, chr-bg</i>	0.4791	0.1915
3.611.25	<i>ciey / ciez, log-bg</i>	0.4356	0.1885
3.611.26	<i>ciey / ciez, pix-lbp-var</i>	0.4623	0.2646
3.611.27	<i>ciey / ciez, satint</i>	0.4746	0.1881
3.611.28	<i>ciey / ciez, reg-ent-c</i>	0.5367	0.3200
3.611.29	<i>ciey / ciez, reg-soh-g</i>	0.5854	0.3527
3.611.30	<i>ciey / ciez, red-z</i>	0.4356	0.1931
3.611.31	<i>ciey / ciez, green-z</i>	0.4126	0.1967
3.611.32	<i>ciey / ciez, blue-z</i>	0.5628	0.1862
3.613.1	<i>ciex, ciey / red</i>	0.4515	0.2139
3.613.2	<i>ciex, ciey / green</i>	0.4086	0.2014

Table B.3: Results of the third step of the Stepwise Forward Selection for the selection of the best parameter sets. Pareto-optimal parameter sets are marked in green. Some close-to-optimal parameter sets, marked in yellow, are also retained for the next step.

Number	Input channels (R_1) / Input channels (R_2)	M	F
3.613.3	<i>ciex, ciey / blue</i>	0.4440	0.1986
3.613.4	<i>ciex, ciey / cos(hue), sin(hue)</i>	0.4642	0.2288
3.613.5	<i>ciex, ciey / cos(hue)</i>	0.4754	0.2588
3.613.6	<i>ciex, ciey / sin(hue)</i>	0.4207	0.2411
3.613.7	<i>ciex, ciey / sat</i>	0.4448	0.2554
3.613.8	<i>ciex, ciey / value</i>	0.4348	0.1957
3.613.9	<i>ciex, ciey / kl2</i>	0.4451	0.2603
3.613.10	<i>ciex, ciey / ciez</i>	0.4258	0.1952
3.613.11	<i>ciex, ciey / ciel</i>	0.4328	0.2017
3.613.12	<i>ciex, ciey / cieb</i>	0.4648	0.2345
3.613.13	<i>ciex, ciey / cieu</i>	0.5116	0.2734
3.613.14	<i>ciex, ciey / ciev</i>	0.4674	0.2316
3.613.15	<i>ciex, ciey / log-rgbb</i>	0.4115	0.2341
3.613.16	<i>ciex, ciey / log-rs</i>	0.4925	0.2543
3.613.17	<i>ciex, ciey / chr-rg</i>	0.5286	0.2863
3.613.18	<i>ciex, ciey / chr-bg</i>	0.4950	0.2362
3.613.19	<i>ciex, ciey / log-bg</i>	0.4962	0.2555
3.613.20	<i>ciex, ciey / pix-lbp-var</i>	0.5187	0.3478
3.613.21	<i>ciex, ciey / satint</i>	0.4393	0.2005
3.613.22	<i>ciex, ciey / reg-ent-c</i>	0.5510	0.3863
3.613.23	<i>ciex, ciey / reg-soh-g</i>	0.7180	0.5433
3.613.24	<i>ciex, ciey / red-z</i>	0.4362	0.2066
3.613.25	<i>ciex, ciey / green-z</i>	0.4627	0.2059
3.613.26	<i>ciex, ciey / blue-z</i>	0.4235	0.1941

Table B.3: Results of the third step of the Stepwise Forward Selection for the selection of the best parameter sets. Pareto-optimal parameter sets are marked in green. Some close-to-optimal parameter sets, marked in yellow, are also retained for the next step.

Number	Input channels (R_1) / Input channels (R_2)	M	F
4.1	<i>green-z, blue, sat, red / \emptyset</i>	0.3352	0.2486
4.5	<i>green-z, blue, sat, green / \emptyset</i>	0.3544	0.2533
4.8	<i>green-z, blue, sat, cos(hue), sin(hue) / \emptyset</i>	0.3382	0.2731
4.10	<i>green-z, blue, sat, cos(hue) / \emptyset</i>	0.3340	0.3016
4.12	<i>green-z, blue, sat, sin(hue) / \emptyset</i>	0.3729	0.2795
4.14	<i>green-z, blue, sat, green-z · cos(hue), green-z · sin(hue) / \emptyset</i>	0.3308	0.2866
4.16	<i>green-z, blue, sat, green-z · cos(hue) / \emptyset</i>	0.3281	0.2730
4.18	<i>green-z, blue, sat, green-z · sin(hue) / \emptyset</i>	0.3260	0.2815
4.20	<i>green-z, blue, sat, (1 - green-z) · cos(hue), (1 - green-z) · sin(hue) / \emptyset</i>	0.3482	0.2893
4.22	<i>green-z, blue, sat, (1 - green-z) · cos(hue) / \emptyset</i>	0.3468	0.2701
4.24	<i>green-z, blue, sat, (1 - green-z) · sin(hue) / \emptyset</i>	0.3660	0.2933
4.26	<i>green-z, blue, sat, blue · cos(hue), blue · sin(hue) / \emptyset</i>	0.3264	0.2922
4.28	<i>green-z, blue, sat, blue · cos(hue) / \emptyset</i>	0.3671	0.2840
4.30	<i>green-z, blue, sat, blue · sin(hue) / \emptyset</i>	0.3507	0.2962
4.32	<i>green-z, blue, sat, (1 - blue) · cos(hue), (1 - blue) · sin(hue) / \emptyset</i>	0.3490	0.2782
4.34	<i>green-z, blue, sat, (1 - blue) · cos(hue) / \emptyset</i>	0.3055	0.2949

Table B.4: Results of the fourth step of the Stepwise Forward Selection for the selection of the best parameter sets. Pareto-optimal parameter sets are marked in green.

Number	Input channels (R_1) / Input channels (R_2)	M	F
4.36	$green-z, blue, sat, (1 - blue) \cdot \sin(hue) / \emptyset$	0.3321	0.3077
4.38	$green-z, blue, sat, sat \cdot \cos(hue), sat \cdot \sin(hue) / \emptyset$	0.3368	0.2624
4.40	$green-z, blue, sat, sat \cdot \cos(hue) / \emptyset$	0.3574	0.2615
4.42	$green-z, blue, sat, sat \cdot \sin(hue) / \emptyset$	0.3066	0.2641
4.44	$green-z, blue, sat, (1 - sat) \cdot \cos(hue), (1 - sat) \cdot \sin(hue) / \emptyset$	0.3345	0.2550
4.46	$green-z, blue, sat, (1 - sat) \cdot \cos(hue) / \emptyset$	0.3576	0.2657
4.48	$green-z, blue, sat, (1 - sat) \cdot \sin(hue) / \emptyset$	0.3247	0.2792
4.53	$green-z, blue, sat, value / \emptyset$	0.3237	0.2480
4.56	$green-z, blue, sat, kl2 / \emptyset$	0.3432	0.2877
4.59	$green-z, blue, sat, ciex / \emptyset$	0.4224	0.2462
4.62	$green-z, blue, sat, ciey / \emptyset$	0.3314	0.2539
4.65	$green-z, blue, sat, ciez / \emptyset$	0.3301	0.2472
4.68	$green-z, blue, sat, ciel / \emptyset$	0.3671	0.2546
4.71	$green-z, blue, sat, cieb / \emptyset$	0.3415	0.2924
4.74	$green-z, blue, sat, cieu / \emptyset$	0.3691	0.2458
4.77	$green-z, blue, sat, ciev / \emptyset$	0.3432	0.2894
4.80	$green-z, blue, sat, log-rgbb / \emptyset$	0.3206	0.2776
4.83	$green-z, blue, sat, log-rs / \emptyset$	0.3600	0.2537
4.86	$green-z, blue, sat, chr-rg / \emptyset$	0.3173	0.2464
4.89	$green-z, blue, sat, chr-bg / \emptyset$	0.3458	0.2423
4.92	$green-z, blue, sat, log-bg / \emptyset$	0.3396	0.2484
4.95	$green-z, blue, sat, pix-lbp-var / \emptyset$	0.4063	0.2773
4.98	$green-z, blue, sat, satint / \emptyset$	0.3362	0.2527
4.103	$green-z, blue, sat, red-z / \emptyset$	0.3771	0.2527
4.106	$green-z, blue, sat, blue-z / \emptyset$	0.4082	0.2424
4.109	$green-z, blue, log-rs, red / \emptyset$	0.3512	0.2491
4.112	$green-z, blue, log-rs, green / \emptyset$	0.3424	0.2401
4.114	$green-z, blue, log-rs, \cos(hue), \sin(hue) / \emptyset$	0.3470	0.2759
4.116	$green-z, blue, log-rs, \cos(hue) / \emptyset$	0.3397	0.2951
4.118	$green-z, blue, log-rs, \sin(hue) / \emptyset$	0.3793	0.2729
4.120	$green-z, blue, log-rs, green-z \cdot \cos(hue), green-z \cdot \sin(hue) / \emptyset$	0.3297	0.2685
4.122	$green-z, blue, log-rs, green-z \cdot \cos(hue) / \emptyset$	0.3331	0.2425
4.124	$green-z, blue, log-rs, green-z \cdot \sin(hue) / \emptyset$	0.3460	0.2604
4.126	$green-z, blue, log-rs, (1 - green-z) \cdot \cos(hue), (1 - green-z) \cdot \sin(hue) / \emptyset$	0.3547	0.2773
4.128	$green-z, blue, log-rs, (1 - green-z) \cdot \cos(hue) / \emptyset$	0.3720	0.2529
4.130	$green-z, blue, log-rs, (1 - green-z) \cdot \sin(hue) / \emptyset$	0.3198	0.2612
4.132	$green-z, blue, log-rs, blue \cdot \cos(hue), blue \cdot \sin(hue) / \emptyset$	0.3339	0.2783
4.134	$green-z, blue, log-rs, blue \cdot \cos(hue) / \emptyset$	0.3566	0.2531
4.136	$green-z, blue, log-rs, blue \cdot \sin(hue) / \emptyset$	0.3508	0.2771
4.138	$green-z, blue, log-rs, (1 - blue) \cdot \cos(hue), (1 - blue) \cdot \sin(hue) / \emptyset$	0.3678	0.2699
4.140	$green-z, blue, log-rs, (1 - blue) \cdot \cos(hue) / \emptyset$	0.3312	0.2605
4.142	$green-z, blue, log-rs, (1 - blue) \cdot \sin(hue) / \emptyset$	0.3425	0.2831
4.144	$green-z, blue, log-rs, log-rs \cdot \cos(hue), log-rs \cdot \sin(hue) / \emptyset$	0.3676	0.2397
4.146	$green-z, blue, log-rs, log-rs \cdot \cos(hue) / \emptyset$	0.3170	0.2409
4.148	$green-z, blue, log-rs, log-rs \cdot \sin(hue) / \emptyset$	0.3331	0.2428
4.150	$green-z, blue, log-rs, (1 - log-rs) \cdot \cos(hue), (1 - log-rs) \cdot \sin(hue) / \emptyset$	0.3289	0.2777
4.152	$green-z, blue, log-rs, (1 - log-rs) \cdot \cos(hue) / \emptyset$	0.3320	0.2608
4.154	$green-z, blue, log-rs, (1 - log-rs) \cdot \sin(hue) / \emptyset$	0.3069	0.2531

Table B.4: Results of the fourth step of the Stepwise Forward Selection for the selection of the best parameter sets. Pareto-optimal parameter sets are marked in green.

Number	Input channels (R_1) / Input channels (R_2)	M	F
4.157	<i>green-z, blue, log-rs, value</i> / \emptyset	0.3466	0.2468
4.159	<i>green-z, blue, log-rs, kl2</i> / \emptyset	0.3544	0.2993
4.161	<i>green-z, blue, log-rs, ciex</i> / \emptyset	0.3885	0.2356
4.163	<i>green-z, blue, log-rs, ciey</i> / \emptyset	0.3333	0.2450
4.165	<i>green-z, blue, log-rs, ciez</i> / \emptyset	0.3387	0.2402
4.167	<i>green-z, blue, log-rs, ciel</i> / \emptyset	0.3862	0.2348
4.169	<i>green-z, blue, log-rs, cieb</i> / \emptyset	0.3051	0.2793
4.171	<i>green-z, blue, log-rs, cieu</i> / \emptyset	0.3869	0.2513
4.173	<i>green-z, blue, log-rs, ciev</i> / \emptyset	0.3112	0.2716
4.175	<i>green-z, blue, log-rs, log-rgbb</i> / \emptyset	0.3327	0.2674
4.177	<i>green-z, blue, log-rs, chr-rg</i> / \emptyset	0.3123	0.2520
4.179	<i>green-z, blue, log-rs, chr-bg</i> / \emptyset	0.3249	0.2427
4.181	<i>green-z, blue, log-rs, log-bg</i> / \emptyset	0.3335	0.2405
4.183	<i>green-z, blue, log-rs, pix-lbp-var</i> / \emptyset	0.3871	0.2648
4.185	<i>green-z, blue, log-rs, satint</i> / \emptyset	0.3485	0.2389
4.187	<i>green-z, blue, log-rs, red-z</i> / \emptyset	0.3482	0.2454
4.189	<i>green-z, blue, log-rs, blue-z</i> / \emptyset	0.4356	0.2210
4.460.1	<i>ciey</i> / <i>ciez, ciez · sin(hue), red</i>	0.4119	0.1940
4.460.2	<i>ciey</i> / <i>ciez, ciez · sin(hue), green</i>	0.4244	0.1967
4.460.3	<i>ciey</i> / <i>ciez, ciez · sin(hue), blue</i>	0.4364	0.1954
4.460.4	<i>ciey</i> / <i>ciez, ciez · sin(hue), sat</i>	0.4307	0.1946
4.460.5	<i>ciey</i> / <i>ciez, ciez · sin(hue), value</i>	0.3881	0.1953
4.460.6	<i>ciey</i> / <i>ciez, ciez · sin(hue), kl2</i>	0.4031	0.1710
4.460.7	<i>ciey</i> / <i>ciez, ciez · sin(hue), ciex</i>	0.5626	0.1853
4.460.8	<i>ciey</i> / <i>ciez, ciez · sin(hue), ciel</i>	0.5333	0.1907
4.460.9	<i>ciey</i> / <i>ciez, ciez · sin(hue), cieb</i>	0.4126	0.1895
4.460.10	<i>ciey</i> / <i>ciez, ciez · sin(hue), cieu</i>	0.4387	0.1814
4.460.11	<i>ciey</i> / <i>ciez, ciez · sin(hue), ciev</i>	0.4147	0.1813
4.460.12	<i>ciey</i> / <i>ciez, ciez · sin(hue), log-rgbb</i>	0.4174	0.1898
4.460.13	<i>ciey</i> / <i>ciez, ciez · sin(hue), log-rs</i>	0.3948	0.1913
4.460.14	<i>ciey</i> / <i>ciez, ciez · sin(hue), chr-rg</i>	0.4316	0.1962
4.460.15	<i>ciey</i> / <i>ciez, ciez · sin(hue), chr-bg</i>	0.4366	0.1967
4.460.16	<i>ciey</i> / <i>ciez, ciez · sin(hue), log-bg</i>	0.4477	0.2001
4.460.17	<i>ciey</i> / <i>ciez, ciez · sin(hue), pix-lbp-var</i>	0.4837	0.2486
4.460.18	<i>ciey</i> / <i>ciez, ciez · sin(hue), satint</i>	0.4237	0.1950
4.460.19	<i>ciey</i> / <i>ciez, ciez · sin(hue), reg-ent-c</i>	0.5115	0.2903
4.460.20	<i>ciey</i> / <i>ciez, ciez · sin(hue), reg-soh-g</i>	0.4839	0.2899
4.460.21	<i>ciey</i> / <i>ciez, ciez · sin(hue), red-z</i>	0.4265	0.1967
4.460.22	<i>ciey</i> / <i>ciez, ciez · sin(hue), green-z</i>	0.4226	0.2005
4.460.23	<i>ciey</i> / <i>ciez, ciez · sin(hue), blue-z</i>	0.4989	0.1865
4.462.1	<i>satint</i> / <i>ciez, log-bg, red</i>	0.4098	0.1923
4.462.2	<i>satint</i> / <i>ciez, log-bg, green</i>	0.4142	0.1931
4.462.3	<i>satint</i> / <i>ciez, log-bg, blue</i>	0.4348	0.2003
4.462.4	<i>satint</i> / <i>ciez, log-bg, cos(hue), sin(hue)</i>	0.3879	0.1906
4.462.5	<i>satint</i> / <i>ciez, log-bg, cos(hue)</i>	0.4476	0.1941
4.462.6	<i>satint</i> / <i>ciez, log-bg, sin(hue)</i>	0.3948	0.1884
4.462.7	<i>satint</i> / <i>ciez, log-bg, ciez · cos(hue), ciez · sin(hue)</i>	0.4085	0.1963
4.462.8	<i>satint</i> / <i>ciez, log-bg, ciez · cos(hue)</i>	0.4114	0.1931
4.462.9	<i>satint</i> / <i>ciez, log-bg, ciez · sin(hue)</i>	0.3889	0.1964

Table B.4: Results of the fourth step of the Stepwise Forward Selection for the selection of the best parameter sets. Pareto-optimal parameter sets are marked in green.

Number	Input channels (R_1) / Input channels (R_2)	M	F
4.462.10	<i>satint</i> / <i>ciez</i> , <i>log-bg</i> , $(1 - \text{ciez}) \cdot \cos(\text{hue})$, $(1 - \text{ciez}) \cdot \sin(\text{hue})$	0.4213	0.2004
4.462.11	<i>satint</i> / <i>ciez</i> , <i>log-bg</i> , $(1 - \text{ciez}) \cdot \cos(\text{hue})$	0.4010	0.1929
4.462.12	<i>satint</i> / <i>ciez</i> , <i>log-bg</i> , $(1 - \text{ciez}) \cdot \sin(\text{hue})$	0.4247	0.2094
4.462.13	<i>satint</i> / <i>ciez</i> , <i>log-bg</i> , <i>log-bg</i> · $\cos(\text{hue})$, <i>log-bg</i> · $\sin(\text{hue})$	0.5323	0.1754
4.462.14	<i>satint</i> / <i>ciez</i> , <i>log-bg</i> , <i>log-bg</i> · $\cos(\text{hue})$	0.4106	0.1845
4.462.15	<i>satint</i> / <i>ciez</i> , <i>log-bg</i> , <i>log-bg</i> · $\sin(\text{hue})$	0.5482	0.1796
4.462.16	<i>satint</i> / <i>ciez</i> , <i>log-bg</i> , $(1 - \text{log-bg}) \cdot \cos(\text{hue})$, $(1 - \text{log-bg}) \cdot \sin(\text{hue})$	0.4474	0.1811
4.462.17	<i>satint</i> / <i>ciez</i> , <i>log-bg</i> , $(1 - \text{log-bg}) \cdot \cos(\text{hue})$	0.6113	0.1879
4.462.18	<i>satint</i> / <i>ciez</i> , <i>log-bg</i> , $(1 - \text{log-bg}) \cdot \sin(\text{hue})$	0.3886	0.1965
4.462.19	<i>satint</i> / <i>ciez</i> , <i>log-bg</i> , <i>sat</i>	0.3707	0.1887
4.462.20	<i>satint</i> / <i>ciez</i> , <i>log-bg</i> , <i>value</i>	0.4035	0.1921
4.462.21	<i>satint</i> / <i>ciez</i> , <i>log-bg</i> , <i>kl2</i>	0.4270	0.1759
4.462.22	<i>satint</i> / <i>ciez</i> , <i>log-bg</i> , <i>ciex</i>	0.5004	0.1727
4.462.23	<i>satint</i> / <i>ciez</i> , <i>log-bg</i> , <i>ciey</i>	0.4022	0.1858
4.462.24	<i>satint</i> / <i>ciez</i> , <i>log-bg</i> , <i>ciel</i>	0.4939	0.1897
4.462.25	<i>satint</i> / <i>ciez</i> , <i>log-bg</i> , <i>cieb</i>	0.4133	0.1844
4.462.26	<i>satint</i> / <i>ciez</i> , <i>log-bg</i> , <i>cieu</i>	0.4260	0.1685
4.462.27	<i>satint</i> / <i>ciez</i> , <i>log-bg</i> , <i>ciev</i>	0.4195	0.1825
4.462.28	<i>satint</i> / <i>ciez</i> , <i>log-bg</i> , <i>log-rgbb</i>	0.4213	0.1995
4.462.29	<i>satint</i> / <i>ciez</i> , <i>log-bg</i> , <i>log-rs</i>	0.4116	0.1869
4.462.30	<i>satint</i> / <i>ciez</i> , <i>log-bg</i> , <i>chr-rg</i>	0.3950	0.1840
4.462.31	<i>satint</i> / <i>ciez</i> , <i>log-bg</i> , <i>chr-bg</i>	0.4078	0.1859
4.462.32	<i>satint</i> / <i>ciez</i> , <i>log-bg</i> , <i>pix-lbp-var</i>	0.4639	0.2571
4.462.33	<i>satint</i> / <i>ciez</i> , <i>log-bg</i> , <i>reg-ent-c</i>	0.4789	0.2965
4.462.34	<i>satint</i> / <i>ciez</i> , <i>log-bg</i> , <i>reg-soh-g</i>	0.5482	0.3703
4.462.35	<i>satint</i> / <i>ciez</i> , <i>log-bg</i> , <i>red-z</i>	0.4018	0.1896
4.462.36	<i>satint</i> / <i>ciez</i> , <i>log-bg</i> , <i>green-z</i>	0.4113	0.1930
4.462.37	<i>satint</i> / <i>ciez</i> , <i>log-bg</i> , <i>blue-z</i>	0.4838	0.1753
4.464.1	<i>ciex</i> / <i>satint</i> , <i>kl2</i> , <i>red</i>	0.4079	0.1723
4.464.2	<i>ciex</i> / <i>satint</i> , <i>kl2</i> , <i>green</i>	0.4382	0.1756
4.464.3	<i>ciex</i> / <i>satint</i> , <i>kl2</i> , <i>blue</i>	0.4609	0.1797
4.464.4	<i>ciex</i> / <i>satint</i> , <i>kl2</i> , $\cos(\text{hue})$, $\sin(\text{hue})$	0.3903	0.1818
4.464.5	<i>ciex</i> / <i>satint</i> , <i>kl2</i> , $\cos(\text{hue})$	0.4498	0.1879
4.464.6	<i>ciex</i> / <i>satint</i> , <i>kl2</i> , $\sin(\text{hue})$	0.4197	0.1777
4.464.7	<i>ciex</i> / <i>satint</i> , <i>kl2</i> , <i>satint</i> · $\cos(\text{hue})$, <i>satint</i> · $\sin(\text{hue})$	0.4027	0.1693
4.464.8	<i>ciex</i> / <i>satint</i> , <i>kl2</i> , <i>satint</i> · $\cos(\text{hue})$	0.3927	0.1776
4.464.9	<i>ciex</i> / <i>satint</i> , <i>kl2</i> , <i>satint</i> · $\sin(\text{hue})$	0.4224	0.1805
4.464.10	<i>ciex</i> / <i>satint</i> , <i>kl2</i> , $(1 - \text{satint}) \cdot \cos(\text{hue})$, $(1 - \text{satint}) \cdot \sin(\text{hue})$	0.4070	0.1781
4.464.11	<i>ciex</i> / <i>satint</i> , <i>kl2</i> , $(1 - \text{satint}) \cdot \cos(\text{hue})$	0.4332	0.1749
4.464.12	<i>ciex</i> / <i>satint</i> , <i>kl2</i> , $(1 - \text{satint}) \cdot \sin(\text{hue})$	0.4120	0.1696
4.464.13	<i>ciex</i> / <i>satint</i> , <i>kl2</i> , <i>kl2</i> · $\cos(\text{hue})$, <i>kl2</i> · $\sin(\text{hue})$	0.4602	0.1965
4.464.14	<i>ciex</i> / <i>satint</i> , <i>kl2</i> , <i>kl2</i> · $\cos(\text{hue})$	0.4286	0.1843
4.464.15	<i>ciex</i> / <i>satint</i> , <i>kl2</i> , <i>kl2</i> · $\sin(\text{hue})$	0.4171	0.1986
4.464.16	<i>ciex</i> / <i>satint</i> , <i>kl2</i> , $(1 - \text{kl2}) \cdot \cos(\text{hue})$, $(1 - \text{kl2}) \cdot \sin(\text{hue})$	0.4209	0.1902
4.464.17	<i>ciex</i> / <i>satint</i> , <i>kl2</i> , $(1 - \text{kl2}) \cdot \cos(\text{hue})$	0.4484	0.1674
4.464.18	<i>ciex</i> / <i>satint</i> , <i>kl2</i> , $(1 - \text{kl2}) \cdot \sin(\text{hue})$	0.4598	0.1909
4.464.19	<i>ciex</i> / <i>satint</i> , <i>kl2</i> , <i>sat</i>	0.4148	0.1753
4.464.20	<i>ciex</i> / <i>satint</i> , <i>kl2</i> , <i>value</i>	0.4275	0.1738
4.464.21	<i>ciex</i> / <i>satint</i> , <i>kl2</i> , <i>ciey</i>	0.4327	0.1755

Table B.4: Results of the fourth step of the Stepwise Forward Selection for the selection of the best parameter sets. Pareto-optimal parameter sets are marked in green.

Number	Input channels (R_1) / Input channels (R_2)	M	F
4.464.22	<i>ciex / satint, kl2, ciez</i>	0.3952	0.1787
4.464.23	<i>ciex / satint, kl2, ciel</i>	0.4679	0.1642
4.464.24	<i>ciex / satint, kl2, cieb</i>	0.4493	0.1623
4.464.25	<i>ciex / satint, kl2, cieu</i>	0.4366	0.1673
4.464.26	<i>ciex / satint, kl2, cieo</i>	0.4042	0.1615
4.464.27	<i>ciex / satint, kl2, log-rgbb</i>	0.4434	0.1670
4.464.28	<i>ciex / satint, kl2, log-rs</i>	0.4039	0.1848
4.464.29	<i>ciex / satint, kl2, chr-rg</i>	0.4413	0.1776
4.464.30	<i>ciex / satint, kl2, chr-bg</i>	0.4389	0.1725
4.464.31	<i>ciex / satint, kl2, log-bg</i>	0.4222	0.1702
4.464.32	<i>ciex / satint, kl2, pix-lbp-var</i>	0.5244	0.2264
4.464.33	<i>ciex / satint, kl2, reg-ent-c</i>	0.5068	0.2773
4.464.34	<i>ciex / satint, kl2, reg-soh-g</i>	0.4908	0.2711
4.464.35	<i>ciex / satint, kl2, red-z</i>	0.4119	0.1743
4.464.36	<i>ciex / satint, kl2, green-z</i>	0.3902	0.1689
4.464.37	<i>ciex / satint, kl2, blue-z</i>	0.5650	0.1677
4.464.38	<i>ciex / satint, cieo, red</i>	0.4203	0.1772
4.464.39	<i>ciex / satint, cieo, green</i>	0.4503	0.1763
4.464.40	<i>ciex / satint, cieo, blue</i>	0.3902	0.1751
4.464.41	<i>ciex / satint, cieo, cos(hue), sin(hue)</i>	0.4420	0.1745
4.464.42	<i>ciex / satint, cieo, cos(hue)</i>	0.4413	0.1887
4.464.43	<i>ciex / satint, cieo, sin(hue)</i>	0.4347	0.1783
4.464.44	<i>ciex / satint, cieo, satint · cos(hue), satint · sin(hue)</i>	0.4187	0.1810
4.464.45	<i>ciex / satint, cieo, satint · cos(hue)</i>	0.4203	0.1830
4.464.46	<i>ciex / satint, cieo, satint · sin(hue)</i>	0.4242	0.1849
4.464.47	<i>ciex / satint, cieo, (1 - satint) · cos(hue), (1 - satint) · sin(hue)</i>	0.4252	0.1852
4.464.48	<i>ciex / satint, cieo, (1 - satint) · cos(hue)</i>	0.4426	0.1817
4.464.49	<i>ciex / satint, cieo, (1 - satint) · sin(hue)</i>	0.4090	0.1902
4.464.50	<i>ciex / satint, cieo, cieo · cos(hue), cieo · sin(hue)</i>	0.3969	0.1939
4.464.51	<i>ciex / satint, cieo, cieo · cos(hue)</i>	0.4536	0.1963
4.464.52	<i>ciex / satint, cieo, cieo · sin(hue)</i>	0.4330	0.1810
4.464.53	<i>ciex / satint, cieo, (1 - cieo) · cos(hue), (1 - cieo) · sin(hue)</i>	0.4301	0.1808
4.464.54	<i>ciex / satint, cieo, (1 - cieo) · cos(hue)</i>	0.4276	0.1874
4.464.55	<i>ciex / satint, cieo, (1 - cieo) · sin(hue)</i>	0.4628	0.1722
4.464.56	<i>ciex / satint, cieo, sat</i>	0.3984	0.1792
4.464.57	<i>ciex / satint, cieo, value</i>	0.4530	0.1787
4.464.58	<i>ciex / satint, cieo, ciey</i>	0.4216	0.1759
4.464.59	<i>ciex / satint, cieo, ciez</i>	0.4051	0.1874
4.464.60	<i>ciex / satint, cieo, ciel</i>	0.4719	0.1774
4.464.61	<i>ciex / satint, cieo, cieu</i>	0.4412	0.1761
4.464.62	<i>ciex / satint, cieo, cieo</i>	0.4320	0.1803
4.464.63	<i>ciex / satint, cieo, log-rgbb</i>	0.4677	0.1754
4.464.64	<i>ciex / satint, cieo, log-rs</i>	0.4336	0.1717
4.464.65	<i>ciex / satint, cieo, chr-rg</i>	0.4845	0.1755
4.464.66	<i>ciex / satint, cieo, chr-bg</i>	0.4532	0.1848
4.464.67	<i>ciex / satint, cieo, log-bg</i>	0.4266	0.1784
4.464.68	<i>ciex / satint, cieo, pix-lbp-var</i>	0.4879	0.2304
4.464.69	<i>ciex / satint, cieo, reg-ent-c</i>	0.5101	0.3035

Table B.4: Results of the fourth step of the Stepwise Forward Selection for the selection of the best parameter sets. Pareto-optimal parameter sets are marked in green.

Number	Input channels (R_1) / Input channels (R_2)	M	F
4.464.70	<i>ciex / satint, cie, reg-soh-g</i>	0.5315	0.3208
4.464.71	<i>ciex / satint, cie, red-z</i>	0.4055	0.1818
4.464.72	<i>ciex / satint, cie, green-z</i>	0.3884	0.1722
4.464.73	<i>ciex / satint, cie, blue-z</i>	0.5142	0.1707
4.464.74	<i>ciex / blue-z, chr-rg, red</i>	0.4008	0.1850
4.464.75	<i>ciex / blue-z, chr-rg, green</i>	0.4010	0.1817
4.464.76	<i>ciex / blue-z, chr-rg, blue</i>	0.4022	0.1878
4.464.77	<i>ciex / blue-z, chr-rg, cos(hue), sin(hue)</i>	0.4422	0.1853
4.464.78	<i>ciex / blue-z, chr-rg, cos(hue)</i>	0.4423	0.1890
4.464.79	<i>ciex / blue-z, chr-rg, sin(hue)</i>	0.4025	0.1866
4.464.80	<i>ciex / blue-z, chr-rg, blue-z · cos(hue), blue-z · sin(hue)</i>	0.5153	0.1738
4.464.81	<i>ciex / blue-z, chr-rg, blue-z · cos(hue)</i>	0.5107	0.1669
4.464.82	<i>ciex / blue-z, chr-rg, blue-z · sin(hue)</i>	0.4344	0.1856
4.464.83	<i>ciex / blue-z, chr-rg, (1 - blue-z) · cos(hue), (1 - blue-z) · sin(hue)</i>	0.3868	0.1902
4.464.84	<i>ciex / blue-z, chr-rg, (1 - blue-z) · cos(hue)</i>	0.4299	0.1797
4.464.85	<i>ciex / blue-z, chr-rg, (1 - blue-z) · sin(hue)</i>	0.4247	0.1994
4.464.86	<i>ciex / blue-z, chr-rg, chr-rg · cos(hue), chr-rg · sin(hue)</i>	0.5633	0.1836
4.464.87	<i>ciex / blue-z, chr-rg, chr-rg · cos(hue)</i>	0.3971	0.1931
4.464.88	<i>ciex / blue-z, chr-rg, chr-rg · sin(hue)</i>	0.5416	0.1663
4.464.89	<i>ciex / blue-z, chr-rg, (1 - chr-rg) · cos(hue), (1 - chr-rg) · sin(hue)</i>	0.4814	0.1865
4.464.90	<i>ciex / blue-z, chr-rg, (1 - chr-rg) · cos(hue)</i>	0.5529	0.1817
4.464.91	<i>ciex / blue-z, chr-rg, (1 - chr-rg) · sin(hue)</i>	0.4447	0.1930
4.464.92	<i>ciex / blue-z, chr-rg, sat</i>	0.4354	0.1737
4.464.93	<i>ciex / blue-z, chr-rg, value</i>	0.4438	0.1824
4.464.94	<i>ciex / blue-z, chr-rg, kl2</i>	0.4100	0.1725
4.464.95	<i>ciex / blue-z, chr-rg, ciey</i>	0.4158	0.1862
4.464.96	<i>ciex / blue-z, chr-rg, ciez</i>	0.4046	0.1856
4.464.97	<i>ciex / blue-z, chr-rg, ciel</i>	0.4739	0.1816
4.464.98	<i>ciex / blue-z, chr-rg, cie</i>	0.4179	0.1796
4.464.99	<i>ciex / blue-z, chr-rg, cie</i>	0.4461	0.1841
4.464.100	<i>ciex / blue-z, chr-rg, cie</i>	0.4300	0.1812
4.464.101	<i>ciex / blue-z, chr-rg, log-rgbb</i>	0.4051	0.1730
4.464.102	<i>ciex / blue-z, chr-rg, log-rs</i>	0.4444	0.1815
4.464.103	<i>ciex / blue-z, chr-rg, chr-bg</i>	0.4141	0.1808
4.464.104	<i>ciex / blue-z, chr-rg, log-bg</i>	0.4073	0.1792
4.464.105	<i>ciex / blue-z, chr-rg, pix-lbp-var</i>	0.4796	0.2583
4.464.106	<i>ciex / blue-z, chr-rg, satint</i>	0.3764	0.1874
4.464.107	<i>ciex / blue-z, chr-rg, reg-ent-c</i>	0.5339	0.3220
4.464.108	<i>ciex / blue-z, chr-rg, reg-soh-g</i>	0.5666	0.3497
4.464.109	<i>ciex / blue-z, chr-rg, red-z</i>	0.4255	0.1839
4.464.110	<i>ciex / blue-z, chr-rg, green-z</i>	0.4258	0.1828
4.464.111	<i>ciex / blue-z, green-z, red</i>	0.4014	0.1888
4.464.112	<i>ciex / blue-z, green-z, green</i>	0.3900	0.1869
4.464.113	<i>ciex / blue-z, green-z, blue</i>	0.4009	0.1895
4.464.114	<i>ciex / blue-z, green-z, cos(hue), sin(hue)</i>	0.4409	0.1708
4.464.115	<i>ciex / blue-z, green-z, cos(hue)</i>	0.4168	0.1785
4.464.116	<i>ciex / blue-z, green-z, sin(hue)</i>	0.4076	0.1795
4.464.117	<i>ciex / blue-z, green-z, blue-z · cos(hue), blue-z · sin(hue)</i>	0.4769	0.1761
4.464.118	<i>ciex / blue-z, green-z, blue-z · cos(hue)</i>	0.5257	0.1840

Table B.4: Results of the fourth step of the Stepwise Forward Selection for the selection of the best parameter sets. Pareto-optimal parameter sets are marked in green.

Number	Input channels (R_1) / Input channels (R_2)	M	F
4.464.119	<i>ciex</i> / <i>blue-z, green-z, blue-z · sin(hue)</i>	0.4014	0.1923
4.464.120	<i>ciex</i> / <i>blue-z, green-z, (1 - blue-z) · cos(hue), (1 - blue-z) · sin(hue)</i>	0.4060	0.1933
4.464.121	<i>ciex</i> / <i>blue-z, green-z, (1 - blue-z) · cos(hue)</i>	0.4083	0.1977
4.464.122	<i>ciex</i> / <i>blue-z, green-z, (1 - blue-z) · sin(hue)</i>	0.3956	0.1996
4.464.123	<i>ciex</i> / <i>blue-z, green-z, green-z · cos(hue), green-z · sin(hue)</i>	0.3851	0.1844
4.464.124	<i>ciex</i> / <i>blue-z, green-z, green-z · cos(hue)</i>	0.4028	0.1842
4.464.125	<i>ciex</i> / <i>blue-z, green-z, green-z · sin(hue)</i>	0.4102	0.1950
4.464.126	<i>ciex</i> / <i>blue-z, green-z, (1 - green-z) · cos(hue), (1 - green-z) · sin(hue)</i>	0.4094	0.2092
4.464.127	<i>ciex</i> / <i>blue-z, green-z, (1 - green-z) · cos(hue)</i>	0.4417	0.2018
4.464.128	<i>ciex</i> / <i>blue-z, green-z, (1 - green-z) · sin(hue)</i>	0.3977	0.1968
4.464.129	<i>ciex</i> / <i>blue-z, green-z, sat</i>	0.3948	0.1852
4.464.130	<i>ciex</i> / <i>blue-z, green-z, value</i>	0.3902	0.1925
4.464.131	<i>ciex</i> / <i>blue-z, green-z, kl2</i>	0.3818	0.1745
4.464.132	<i>ciex</i> / <i>blue-z, green-z, ciej</i>	0.3850	0.1942
4.464.133	<i>ciex</i> / <i>blue-z, green-z, ciez</i>	0.4077	0.1884
4.464.134	<i>ciex</i> / <i>blue-z, green-z, ciel</i>	0.4538	0.1763
4.464.135	<i>ciex</i> / <i>blue-z, green-z, cieb</i>	0.4007	0.1838
4.464.136	<i>ciex</i> / <i>blue-z, green-z, cieu</i>	0.4380	0.1744
4.464.137	<i>ciex</i> / <i>blue-z, green-z, ciev</i>	0.3870	0.1777
4.464.138	<i>ciex</i> / <i>blue-z, green-z, log-rgbb</i>	0.3670	0.1878
4.464.139	<i>ciex</i> / <i>blue-z, green-z, log-rs</i>	0.3934	0.1777
4.464.140	<i>ciex</i> / <i>blue-z, green-z, chr-bg</i>	0.4125	0.1837
4.464.141	<i>ciex</i> / <i>blue-z, green-z, log-bg</i>	0.3882	0.1896
4.464.142	<i>ciex</i> / <i>blue-z, green-z, pix-lbp-var</i>	0.4347	0.2356
4.464.143	<i>ciex</i> / <i>blue-z, green-z, satint</i>	0.4036	0.1873
4.464.144	<i>ciex</i> / <i>blue-z, green-z, reg-ent-c</i>	0.5127	0.2970
4.464.145	<i>ciex</i> / <i>blue-z, green-z, reg-soh-g</i>	0.5007	0.3087
4.464.146	<i>ciex</i> / <i>blue-z, green-z, red-z</i>	0.4174	0.1844
4.466.1	<i>green-z, blue, log-rs / red</i>	0.4394	0.2091
4.466.2	<i>green-z, blue, log-rs / green</i>	0.4252	0.2066
4.466.3	<i>green-z, blue, log-rs / cos(hue), sin(hue)</i>	0.4697	0.2293
4.466.4	<i>green-z, blue, log-rs / cos(hue)</i>	0.4946	0.2604
4.466.5	<i>green-z, blue, log-rs / sin(hue)</i>	0.4666	0.2491
4.466.6	<i>green-z, blue, log-rs / sat</i>	0.5094	0.2540
4.466.7	<i>green-z, blue, log-rs / value</i>	0.4643	0.2067
4.466.8	<i>green-z, blue, log-rs / kl2</i>	0.4780	0.2526
4.466.9	<i>green-z, blue, log-rs / ciex</i>	0.4685	0.2092
4.466.10	<i>green-z, blue, log-rs / ciej</i>	0.4367	0.2011
4.466.11	<i>green-z, blue, log-rs / ciez</i>	0.4469	0.1870
4.466.12	<i>green-z, blue, log-rs / ciel</i>	0.4178	0.2163
4.466.13	<i>green-z, blue, log-rs / cieb</i>	0.4800	0.2293
4.466.14	<i>green-z, blue, log-rs / cieu</i>	0.4983	0.2596
4.466.15	<i>green-z, blue, log-rs / ciev</i>	0.4597	0.2327
4.466.16	<i>green-z, blue, log-rs / log-rgbb</i>	0.4522	0.2212
4.466.17	<i>green-z, blue, log-rs / chr-rg</i>	0.5489	0.2793
4.466.18	<i>green-z, blue, log-rs / chr-bg</i>	0.4658	0.2392
4.466.19	<i>green-z, blue, log-rs / log-bg</i>	0.4850	0.2221
4.466.20	<i>green-z, blue, log-rs / pix-lbp-var</i>	0.5438	0.3579
4.466.21	<i>green-z, blue, log-rs / satint</i>	0.4645	0.2063

Table B.4: Results of the fourth step of the Stepwise Forward Selection for the selection of the best parameter sets. Pareto-optimal parameter sets are marked in green.

Number	Input channels (R_1) / Input channels (R_2)	M	F
4.466.22	<i>green-z, blue, log-rs / reg-ent-c</i>	0.5665	0.3908
4.466.23	<i>green-z, blue, log-rs / reg-soh-g</i>	0.7096	0.5454
4.466.24	<i>green-z, blue, log-rs / red-z</i>	0.4662	0.2094
4.466.25	<i>green-z, blue, log-rs / blue-z</i>	0.4392	0.1870
4.468.1	<i>green-z, blue, sat / red</i>	0.4543	0.2225
4.468.2	<i>green-z, blue, sat / green</i>	0.3846	0.1978
4.468.3	<i>green-z, blue, sat / cos(hue), sin(hue)</i>	0.4936	0.2436
4.468.4	<i>green-z, blue, sat / cos(hue)</i>	0.4777	0.2569
4.468.5	<i>green-z, blue, sat / sin(hue)</i>	0.4289	0.2512
4.468.6	<i>green-z, blue, sat / value</i>	0.4018	0.2073
4.468.7	<i>green-z, blue, sat / kl2</i>	0.4825	0.2561
4.468.8	<i>green-z, blue, sat / ciex</i>	0.4324	0.2185
4.468.9	<i>green-z, blue, sat / ciey</i>	0.4388	0.2078
4.468.10	<i>green-z, blue, sat / ciez</i>	0.4409	0.2100
4.468.11	<i>green-z, blue, sat / ciel</i>	0.4197	0.2043
4.468.12	<i>green-z, blue, sat / cieb</i>	0.4888	0.2235
4.468.13	<i>green-z, blue, sat / cieu</i>	0.4731	0.2720
4.468.14	<i>green-z, blue, sat / cieo</i>	0.4925	0.2404
4.468.15	<i>green-z, blue, sat / log-rgbb</i>	0.4842	0.2301
4.468.16	<i>green-z, blue, sat / log-rs</i>	0.4828	0.2786
4.468.17	<i>green-z, blue, sat / chr-rg</i>	0.5009	0.2750
4.468.18	<i>green-z, blue, sat / chr-bg</i>	0.5072	0.2559
4.468.19	<i>green-z, blue, sat / log-bg</i>	0.4748	0.2397
4.468.20	<i>green-z, blue, sat / pix-lbp-var</i>	0.5506	0.3758
4.468.21	<i>green-z, blue, sat / satint</i>	0.4367	0.1934
4.468.22	<i>green-z, blue, sat / reg-ent-c</i>	0.5606	0.3831
4.468.23	<i>green-z, blue, sat / reg-soh-g</i>	0.7287	0.5371
4.468.24	<i>green-z, blue, sat / red-z</i>	0.4412	0.2134
4.468.25	<i>green-z, blue, sat / blue-z</i>	0.4372	0.1996
4.470.1	<i>blue-z / green-z, green-z · sin(hue), red</i>	0.3856	0.1969
4.470.2	<i>blue-z / green-z, green-z · sin(hue), green</i>	0.3958	0.2064
4.470.3	<i>blue-z / green-z, green-z · sin(hue), blue</i>	0.3792	0.1938
4.470.4	<i>blue-z / green-z, green-z · sin(hue), sat</i>	0.3985	0.2000
4.470.5	<i>blue-z / green-z, green-z · sin(hue), value</i>	0.4153	0.2063
4.470.6	<i>blue-z / green-z, green-z · sin(hue), kl2</i>	0.4042	0.1738
4.470.7	<i>blue-z / green-z, green-z · sin(hue), ciex</i>	0.5369	0.1946
4.470.8	<i>blue-z / green-z, green-z · sin(hue), ciey</i>	0.4230	0.1969
4.470.9	<i>blue-z / green-z, green-z · sin(hue), ciez</i>	0.4057	0.1954
4.470.10	<i>blue-z / green-z, green-z · sin(hue), ciel</i>	0.4621	0.2048
4.470.11	<i>blue-z / green-z, green-z · sin(hue), cieb</i>	0.4326	0.1918
4.470.12	<i>blue-z / green-z, green-z · sin(hue), cieu</i>	0.4289	0.1896
4.470.13	<i>blue-z / green-z, green-z · sin(hue), cieo</i>	0.4056	0.1919
4.470.14	<i>blue-z / green-z, green-z · sin(hue), log-rgbb</i>	0.4077	0.1863
4.470.15	<i>blue-z / green-z, green-z · sin(hue), log-rs</i>	0.4099	0.1914
4.470.16	<i>blue-z / green-z, green-z · sin(hue), chr-rg</i>	0.4275	0.2021
4.470.17	<i>blue-z / green-z, green-z · sin(hue), chr-bg</i>	0.3952	0.1956
4.470.18	<i>blue-z / green-z, green-z · sin(hue), log-bg</i>	0.3985	0.1933
4.470.19	<i>blue-z / green-z, green-z · sin(hue), pix-lbp-var</i>	0.4049	0.2322
4.470.20	<i>blue-z / green-z, green-z · sin(hue), satint</i>	0.3816	0.1899

Table B.4: Results of the fourth step of the Stepwise Forward Selection for the selection of the best parameter sets. Pareto-optimal parameter sets are marked in green.

Number	Input channels (R_1) / Input channels (R_2)	M	F
4.470.21	<i>blue-z / green-z, green-z · sin(hue), reg-ent-c</i>	0.5105	0.2963
4.470.22	<i>blue-z / green-z, green-z · sin(hue), reg-soh-g</i>	0.5162	0.3177
4.470.23	<i>blue-z / green-z, green-z · sin(hue), red-z</i>	0.4099	0.1998
4.470.24	<i>blue-z / green-z, log-rs, red</i>	0.4144	0.1807
4.470.25	<i>blue-z / green-z, log-rs, green</i>	0.4456	0.1851
4.470.26	<i>blue-z / green-z, log-rs, blue</i>	0.4140	0.1884
4.470.27	<i>blue-z / green-z, log-rs, cos(hue), sin(hue)</i>	0.4674	0.1937
4.470.28	<i>blue-z / green-z, log-rs, cos(hue)</i>	0.4569	0.2011
4.470.29	<i>blue-z / green-z, log-rs, sin(hue)</i>	0.4411	0.1851
4.470.30	<i>blue-z / green-z, log-rs, green-z · cos(hue), green-z · sin(hue)</i>	0.4189	0.1947
4.470.31	<i>blue-z / green-z, log-rs, green-z · cos(hue)</i>	0.4040	0.1910
4.470.32	<i>blue-z / green-z, log-rs, (1 - green-z) · cos(hue), (1 - green-z) · sin(hue)</i>	0.3990	0.1995
4.470.33	<i>blue-z / green-z, log-rs, (1 - green-z) · cos(hue)</i>	0.4420	0.1920
4.470.34	<i>blue-z / green-z, log-rs, (1 - green-z) · sin(hue)</i>	0.3658	0.1885
4.470.35	<i>blue-z / green-z, log-rs, log-rs · cos(hue), log-rs · sin(hue)</i>	0.5030	0.1798
4.470.36	<i>blue-z / green-z, log-rs, log-rs · cos(hue)</i>	0.4638	0.1773
4.470.37	<i>blue-z / green-z, log-rs, log-rs · sin(hue)</i>	0.5484	0.1785
4.470.38	<i>blue-z / green-z, log-rs, (1 - log-rs) · cos(hue), (1 - log-rs) · sin(hue)</i>	0.4988	0.1952
4.470.39	<i>blue-z / green-z, log-rs, (1 - log-rs) · cos(hue)</i>	0.5191	0.1999
4.470.40	<i>blue-z / green-z, log-rs, (1 - log-rs) · sin(hue)</i>	0.4119	0.1960
4.470.41	<i>blue-z / green-z, log-rs, sat</i>	0.4223	0.1882
4.470.42	<i>blue-z / green-z, log-rs, value</i>	0.4430	0.1828
4.470.43	<i>blue-z / green-z, log-rs, kl2</i>	0.4343	0.1866
4.470.44	<i>blue-z / green-z, log-rs, ciex</i>	0.5764	0.1691
4.470.45	<i>blue-z / green-z, log-rs, ciey</i>	0.4003	0.1917
4.470.46	<i>blue-z / green-z, log-rs, ciez</i>	0.3969	0.1774
4.470.47	<i>blue-z / green-z, log-rs, ciel</i>	0.5036	0.1813
4.470.48	<i>blue-z / green-z, log-rs, cieb</i>	0.4547	0.1760
4.470.49	<i>blue-z / green-z, log-rs, cieu</i>	0.4344	0.1913
4.470.50	<i>blue-z / green-z, log-rs, cieo</i>	0.4098	0.1719
4.470.51	<i>blue-z / green-z, log-rs, log-rgbb</i>	0.4571	0.1830
4.470.52	<i>blue-z / green-z, log-rs, chr-rg</i>	0.4280	0.1898
4.470.53	<i>blue-z / green-z, log-rs, chr-bg</i>	0.4392	0.1784
4.470.54	<i>blue-z / green-z, log-rs, log-bg</i>	0.4188	0.1875
4.470.55	<i>blue-z / green-z, log-rs, pix-lbp-var</i>	0.4653	0.2387
4.470.56	<i>blue-z / green-z, log-rs, satint</i>	0.4274	0.1923
4.470.57	<i>blue-z / green-z, log-rs, reg-ent-c</i>	0.5221	0.3045
4.470.58	<i>blue-z / green-z, log-rs, reg-soh-g</i>	0.5039	0.3341
4.470.59	<i>blue-z / green-z, log-rs, red-z</i>	0.4438	0.1827

Table B.4: Results of the fourth step of the Stepwise Forward Selection for the selection of the best parameter sets. Pareto-optimal parameter sets are marked in green.

$C(P)$	angles	m	M	F
L_g^W	n/a	0.001	0.5242	0.3010
L_g^W	n/a	0.002	0.4804	0.2900
L_g^W	n/a	0.005	0.4695	0.2746

Table B.8: Segmentation performance for different anisotropic energies, angles, and values of m . "N/a" indicates values which are not applicable or used in that particular case.

$C(P)$	angles	m	M	F
L_{σ}^W	n/a	0.01	0.4834	0.2778
L_{σ}^W	n/a	0.02	0.4906	0.2712
L_{σ}^W	n/a	0.05	0.4779	0.2652
L_{σ}^W	n/a	0.1	0.4792	0.2760
L_{σ}^W	n/a	0.2	0.4819	0.2708
L_{σ}^W	n/a	0.5	0.4869	0.2581
L_{σ}^W	n/a	1	0.4666	0.2519
L_{σ}^W	n/a	2	0.4798	0.2538
L_{σ}^W	n/a	5	0.5113	0.2670
L_{σ}^W	n/a	10	0.4614	0.2580
T_{σ}^W	n/a	0.001	0.4599	0.2874
T_{σ}^W	n/a	0.002	0.4475	0.2825
T_{σ}^W	n/a	0.005	0.4338	0.2681
T_{σ}^W	n/a	0.01	0.4324	0.2778
T_{σ}^W	n/a	0.02	0.4631	0.2690
T_{σ}^W	n/a	0.05	0.4694	0.2911
T_{σ}^W	n/a	0.1	0.4529	0.2782
T_{σ}^W	n/a	0.2	0.4538	0.2649
T_{σ}^W	n/a	0.5	0.4539	0.2609
T_{σ}^W	n/a	1	0.4567	0.2710
T_{σ}^W	n/a	2	0.4821	0.2795
T_{σ}^W	n/a	5	0.4514	0.2703
T_{σ}^W	n/a	10	0.4335	0.2621
P_{σ}^W	n/a	0.001	0.4422	0.2637
P_{σ}^W	n/a	0.002	0.4728	0.2672
P_{σ}^W	n/a	0.005	0.4644	0.2547
P_{σ}^W	n/a	0.01	0.4488	0.2667
P_{σ}^W	n/a	0.02	0.4667	0.2660
P_{σ}^W	n/a	0.05	0.4658	0.2744
P_{σ}^W	n/a	0.1	0.4600	0.2680
P_{σ}^W	n/a	0.2	0.4640	0.2715
P_{σ}^W	n/a	0.5	0.4811	0.2647
P_{σ}^W	n/a	1	0.4633	0.2656
P_{σ}^W	n/a	2	0.4434	0.2542
P_{σ}^W	n/a	5	0.4539	0.2497
P_{σ}^W	n/a	10	0.4402	0.2631
$P_{\sigma}^{W,N}$	isotropic	0.001	0.5593	0.3677
$P_{\sigma}^{W,N}$	isotropic	0.002	0.5444	0.3315
$P_{\sigma}^{W,N}$	isotropic	0.005	0.5187	0.3107
$P_{\sigma}^{W,N}$	isotropic	0.01	0.5048	0.2974
$P_{\sigma}^{W,N}$	isotropic	0.02	0.5096	0.2827
$P_{\sigma}^{W,N}$	isotropic	0.05	0.4985	0.2788
$P_{\sigma}^{W,N}$	isotropic	0.1	0.4937	0.2592
$P_{\sigma}^{W,N}$	isotropic	0.2	0.4853	0.2618
$P_{\sigma}^{W,N}$	isotropic	0.5	0.4804	0.2730
$P_{\sigma}^{W,N}$	isotropic	1	0.5008	0.2695
$P_{\sigma}^{W,N}$	isotropic	2	0.4602	0.2637

Table B.8: Segmentation performance for different anisotropic energies, angles, and values of m . "N/a" indicates values which are not applicable or used in that particular case.

$C(P)$	angles	m	M	F
$P_{\sigma}^{W,N}$	isotropic	5	0.4728	0.2743
$P_{\sigma}^{W,N}$	isotropic	10	0.4775	0.2668
$P_{\sigma}^{W,N}$	$\text{ang}(\int W, z)$	0.001	0.5210	0.3391
$P_{\sigma}^{W,N}$	$\text{ang}(\int W, z)$	0.002	0.5231	0.3185
$P_{\sigma}^{W,N}$	$\text{ang}(\int W, z)$	0.005	0.5380	0.3195
$P_{\sigma}^{W,N}$	$\text{ang}(\int W, z)$	0.01	0.5003	0.2894
$P_{\sigma}^{W,N}$	$\text{ang}(\int W, z)$	0.02	0.4981	0.2789
$P_{\sigma}^{W,N}$	$\text{ang}(\int W, z)$	0.05	0.4853	0.2747
$P_{\sigma}^{W,N}$	$\text{ang}(\int W, z)$	0.1	0.4791	0.2713
$P_{\sigma}^{W,N}$	$\text{ang}(\int W, z)$	0.2	0.4935	0.2801
$P_{\sigma}^{W,N}$	$\text{ang}(\int W, z)$	0.5	0.4921	0.2862
$P_{\sigma}^{W,N}$	$\text{ang}(\int W, z)$	1	0.4939	0.2775
$P_{\sigma}^{W,N}$	$\text{ang}(\int W, z)$	2	0.4898	0.2629
$P_{\sigma}^{W,N}$	$\text{ang}(\int W, z)$	5	0.4862	0.2915
$P_{\sigma}^{W,N}$	$\text{ang}(\int W, z)$	10	0.4720	0.2645
$P_{\sigma}^{W,N}$	$\text{ang}(W, z)$	0.001	0.5167	0.3453
$P_{\sigma}^{W,N}$	$\text{ang}(W, z)$	0.002	0.5532	0.3413
$P_{\sigma}^{W,N}$	$\text{ang}(W, z)$	0.005	0.5229	0.3255
$P_{\sigma}^{W,N}$	$\text{ang}(W, z)$	0.01	0.4934	0.2951
$P_{\sigma}^{W,N}$	$\text{ang}(W, z)$	0.02	0.5021	0.2884
$P_{\sigma}^{W,N}$	$\text{ang}(W, z)$	0.05	0.4817	0.2709
$P_{\sigma}^{W,N}$	$\text{ang}(W, z)$	0.1	0.5053	0.2658
$P_{\sigma}^{W,N}$	$\text{ang}(W, z)$	0.2	0.4983	0.2812
$P_{\sigma}^{W,N}$	$\text{ang}(W, z)$	0.5	0.4806	0.2764
$P_{\sigma}^{W,N}$	$\text{ang}(W, z)$	1	0.4827	0.2708
$P_{\sigma}^{W,N}$	$\text{ang}(W, z)$	2	0.4804	0.2633
$P_{\sigma}^{W,N}$	$\text{ang}(W, z)$	5	0.4785	0.2787
$P_{\sigma}^{W,N}$	$\text{ang}(W, z)$	10	0.4851	0.2665
C_{σ}^W	isotropic	n/a	0.7038	0.5194
C_{σ}^W	$\text{ang}(\int W, z)$	0.001	0.7133	0.5125
C_{σ}^W	$\text{ang}(\int W, z)$	0.002	0.6875	0.4723
C_{σ}^W	$\text{ang}(\int W, z)$	0.005	0.7139	0.5018
C_{σ}^W	$\text{ang}(\int W, z)$	0.01	0.6829	0.4984
C_{σ}^W	$\text{ang}(\int W, z)$	0.02	0.7116	0.5154
C_{σ}^W	$\text{ang}(\int W, z)$	0.05	0.6905	0.4829
C_{σ}^W	$\text{ang}(\int W, z)$	0.1	0.6514	0.4849
C_{σ}^W	$\text{ang}(\int W, z)$	0.2	0.6574	0.4900
C_{σ}^W	$\text{ang}(\int W, z)$	0.5	0.6726	0.4820
C_{σ}^W	$\text{ang}(\int W, z)$	1	0.6879	0.5019
C_{σ}^W	$\text{ang}(\int W, z)$	2	0.6981	0.4984
C_{σ}^W	$\text{ang}(\int W, z)$	5	0.6735	0.4949
C_{σ}^W	$\text{ang}(\int W, z)$	10	0.7008	0.5013
C_{σ}^W	$\text{ang}(W, z)$	0.001	0.7081	0.5282
C_{σ}^W	$\text{ang}(W, z)$	0.002	0.6851	0.5092

Table B.8: Segmentation performance for different anisotropic energies, angles, and values of m . "N/a" indicates values which are not applicable or used in that particular case.

$C(P)$	angles	m	M	F
C_{σ}^W	$\text{ang}(W, z)$	0.005	0.7251	0.5292
C_{σ}^W	$\text{ang}(W, z)$	0.01	0.7130	0.5166
C_{σ}^W	$\text{ang}(W, z)$	0.02	0.7010	0.5169
C_{σ}^W	$\text{ang}(W, z)$	0.05	0.6905	0.5114
C_{σ}^W	$\text{ang}(W, z)$	0.1	0.6994	0.5434
C_{σ}^W	$\text{ang}(W, z)$	0.2	0.6995	0.5393
C_{σ}^W	$\text{ang}(W, z)$	0.5	0.7148	0.5319
C_{σ}^W	$\text{ang}(W, z)$	1	0.7103	0.5189
C_{σ}^W	$\text{ang}(W, z)$	2	0.6974	0.5095
C_{σ}^W	$\text{ang}(W, z)$	5	0.7010	0.5298
C_{σ}^W	$\text{ang}(W, z)$	10	0.6981	0.5274

Table B.8: Segmentation performance for different anisotropic energies, angles, and values of m . "N/a" indicates values which are not applicable or used in that particular case.

t_{low}	t_{high}	M	F
10	10	0.3184	0.2465
10	15	0.3218	0.2456
10	20	0.3197	0.2463
10	30	0.3347	0.2477
20	20	0.3209	0.2447
20	30	0.3345	0.2483
20	40	0.3198	0.2445
20	60	0.3364	0.2474
50	50	0.3273	0.2498
50	75	0.3464	0.2456
50	100	0.3364	0.2427
50	150	0.3814	0.2352
70	70	0.3556	0.2393
70	105	0.3463	0.2349
70	140	0.3891	0.2259
70	210	0.3836	0.2181
100	100	0.4196	0.2347
100	150	0.3764	0.2313
100	200	0.4184	0.2229
100	300	0.3484	0.1930
200	200	0.4171	0.2246
200	300	0.3782	0.1977
200	400	0.3489	0.1847
200	600	0.4166	0.1592
300	300	0.4228	0.2046
300	450	0.3847	0.1915
300	600	0.3799	0.1756
300	900	0.3881	0.1549
400	400	0.4222	0.1979
400	600	0.3884	0.1856
400	800	0.4048	0.1699
400	1200	0.3982	0.1424
500	500	0.3808	0.1916
500	750	0.4192	0.1821
500	1000	0.4541	0.1854
500	1500	0.4268	0.1484
1000	1000	0.4488	0.1859
1000	1500	0.4566	0.1718
1000	2000	0.4487	0.1540
1000	3000	0.4653	0.1443

Table B.5: Segmentation performance for different values of t_{low} and t_{high} , thresholds controlling the double-phase segmentation method.

k	a	M	F
0.05	0.2	0.4925	0.3670
0.05	0.5	0.5142	0.3625
0.05	0.7	0.4930	0.3603
0.05	1.0	0.4936	0.3591
0.05	1.4	0.5029	0.3527
0.05	2.0	0.4942	0.3609
0.05	4.0	0.4836	0.3620
0.05	9.0	0.5056	0.3070
0.3	0.2	0.5066	0.3674
0.3	0.5	0.5307	0.3664
0.3	0.7	0.5253	0.3575
0.3	1.0	0.4997	0.3610
0.3	1.4	0.5134	0.3607
0.3	2.0	0.4868	0.3668
0.3	4.0	0.4752	0.3522
0.3	9.0	0.4665	0.2934
1.0	0.2	0.5023	0.3670
1.0	0.5	0.4919	0.3696
1.0	0.7	0.4964	0.3571
1.0	1.0	0.4735	0.3575
1.0	1.4	0.5131	0.3656
1.0	2.0	0.5003	0.3712
1.0	4.0	0.4765	0.3541
1.0	9.0	0.4523	0.2668
3.0	0.2	0.4867	0.3704
3.0	0.5	0.5126	0.3704
3.0	0.7	0.5078	0.3713
3.0	1.0	0.4922	0.3762
3.0	1.4	0.5081	0.3684
3.0	2.0	0.5124	0.3701
3.0	4.0	0.4801	0.3524
3.0	9.0	0.4672	0.2600
8.0	0.2	0.5198	0.3739
8.0	0.5	0.5147	0.3666
8.0	0.7	0.4902	0.3708
8.0	1.0	0.5124	0.3660
8.0	1.4	0.5119	0.3615
8.0	2.0	0.4925	0.3698
8.0	4.0	0.5123	0.3522
8.0	9.0	0.4363	0.2345

Table B.6: Segmentation performance for different values of k and a for a gradient-based data fit modification to the complexity energy of $f(G) = 1/(k + G^a)$ of equations (3.22) and (3.23).

k	a	M	F
0.3	0.2	0.5040	0.3756
0.3	0.5	0.5087	0.3749
0.3	0.7	0.5158	0.3686
0.3	1.0	0.5044	0.3685
0.3	1.4	0.5074	0.3680
0.3	2.0	0.5288	0.3654
0.3	4.0	0.4887	0.3644
0.3	9.0	0.4494	0.2540
0.7	0.2	0.5149	0.3726
0.7	0.5	0.5032	0.3636
0.7	0.7	0.4988	0.3603
0.7	1.0	0.4969	0.3653
0.7	1.4	0.4961	0.3655
0.7	2.0	0.5087	0.3646
0.7	4.0	0.4782	0.3516
0.7	9.0	0.4628	0.2572
1.0	0.2	0.5143	0.3682
1.0	0.5	0.4860	0.3568
1.0	0.7	0.4996	0.3631
1.0	1.0	0.5324	0.3648
1.0	1.4	0.5079	0.3700
1.0	2.0	0.5019	0.3671
1.0	4.0	0.4990	0.3610
1.0	9.0	0.4533	0.2697
2.0	0.2	0.4996	0.3645
2.0	0.5	0.4898	0.3567
2.0	0.7	0.5245	0.3598
2.0	1.0	0.5115	0.3646
2.0	1.4	0.5098	0.3663
2.0	2.0	0.5010	0.3696
2.0	4.0	0.5147	0.3655
2.0	9.0	0.4706	0.2811
5.0	0.2	0.4959	0.3597
5.0	0.5	0.5357	0.3652
5.0	0.7	0.5019	0.3630
5.0	1.0	0.5036	0.3535
5.0	1.4	0.5106	0.3579
5.0	2.0	0.4948	0.3623
5.0	4.0	0.4979	0.3647
5.0	9.0	0.4766	0.2996

Table B.7: Segmentation performance for different values of k and a for a gradient-based data fit modification to the complexity energy of $f(G) = \exp(-kG^a)$ of equations (3.22) and (3.24).

C | Publications

This appendix contains the list of all publications of which I am an author or coauthor, which are related to the topics of this thesis, and which were published or accepted for publication before the thesis defence.

- [GTSC⁺05] Laurent Guigues, Roger Trias-Sanz, Nesrine Chehata, Franck Taillandier, and Matthieu Deveau. Segmentation multi-échelles d'images: théorie et applications. *Bulletin d'Information Scientifique et Technique de l'Institut Géographique National*, 75:41–57, 2006. (In French).
- [TS03] Roger Trias-Sanz. Locating and classifying vegetation areas in high-resolution multi-spectral aerial images with very high reliability using supplementary information: A literature review. Technical Report 2003/1, SIP-CRIP5 (Université de Paris 5) and Institut Géographique National, 45 rue des Saints-Pères, Paris, France, November 2003.
- [TS04a] Roger Trias-Sanz. An edge-based method for registering a graph onto an image with application to cadastre registration. In ACIVS 2004 [ACI04], pages 333–340.
- [TS04b] Roger Trias-Sanz. An edge-based method for registering a graph onto an image with application to cadastre registration. Technical Report 2004/1, SIP-CRIP5 (Université de Paris 5) and Institut Géographique National, 45 rue des Saints-Pères, Paris, France, March 2004. (In French).
- [TS05a] Roger Trias-Sanz. A metric for evaluating and comparing hierarchical and multi-scale image segmentations. In *Proc. of the 2005 International Geosciences And Remote Sensing Symposium (IGARSS 2005)*, Seoul, South Korea, July 2005.
- [TS05b] Roger Trias-Sanz. A texture orientation estimator for discriminating between forests, orchards, vineyards, and tilled fields. In *Proc. of the 2005 International Geosciences And Remote Sensing Symposium (IGARSS 2005)*, Seoul, South Korea, July 2005.
- [TS06] Roger Trias-Sanz. Méthodes pour la classification semi-automatique du terrain en zone rurale. *Bulletin d'Information Scientifique et Technique de l'Institut Géographique National*, 2006? To appear. (In French).

- [TSB05] Roger Trias-Sanz and Didier Boldo. A high-reliability, high-resolution method for land cover classification into forest and non-forest. In Heikki Kälviäinen, Jussi Parkkinen, and Arto Kaarna, editors, *Proc. of the 14th Scandinavian Conference on Image Analysis (SCIA 2005)*, volume 3540 of *Lecture Notes in Computer Science*, pages 831–840, Joensuu, Finland, June 2005. Springer.
- [TSPD04] Roger Trias-Sanz and Marc Pierrot-Deseilligny. A region-based method for graph to image registration with application to cadastre data. In ICIP 2004 [ICI04].

Other publications of which I am the author or a coauthor, related to image analysis but not to the specific topics of this thesis, are:

- [LBTS03] N. Loménie, J. Barbeau, and R. Trias-Sanz. Integrating textural and geometric information for an automatic bridge detection system. In *Proc. of the 2003 International Geosciences And Remote Sensing Symposium (IGARSS 2003)*, Toulouse, France, July 2003.
- [TS02] R. Trias-Sanz. Automatically detecting geographical objects in high-resolution satellite images. Master's thesis, Laboratoire SIP, CRIP5, Université René Descartes-Paris 5, 45 rue des Saints-Pères; F-75006 Paris; France, September 2002.
- [TSL03] R. Trias-Sanz and N. Loménie. Automatic bridge detection in high-resolution satellite images. In J. L. Crowley et al., editors, *Proc. of the 3rd International Conference on Computer Vision Systems (ICVS 03)*, volume 2626 of *Lecture Notes in Computer Science*, pages 172–181, Graz, Austria, April 2003. Springer.
- [TSLB04] R. Trias-Sanz, N. Loménie, and J. Barbeau. Using textural and geometric information for an automatic bridge detection system. In *Proc. of the 2004 Advanced Concepts for Intelligent Vision Systems conference (ACIVS 2004)*, pages 325–332, Brussels, Belgium, September 2004.

D

Bibliography

- [AA04] Paul Aplin and Peter M. Atkinson. Predicting missing field boundaries to increase per-field classification accuracy. *Photogrammetric Engineering and Remote Sensing*, 70(1):141–149, January 2004.
- [AAD02] Ahmed Al-Ani and Mohamed Deriche. Feature selection using a mutual information based measure. In ICPR 2002 [ICP02].
- [ACI04] *Proc. of the 2004 Conf. on Advanced Concepts for Intelligent Vision Systems (ACIVS 2004)*, Brussels, Belgium, 2004.
- [AD02] Hakan Altınçay and Mübeccel Demirekler. Why does output normalization create problems in multiple classifier systems? In ICPR 2002 [ICP02].
- [AEMT03] Islam Abou El-Magd and T. W. Tanton. Improvements in land use mapping for irrigated agriculture from satellite sensor data using a multi-stage maximum likelihood classification. *International Journal of Remote Sensing*, 24(21):4197–4206, November 2003.
- [AK99] F. M. Alkoot and Josef Kittler. Experimental evaluation of expert fusion strategies. *Pattern Recognition Letters*, 20:1361–1369, 1999.
- [AKT+05] Selim Aksoy, Krzysztof Koperski, Carsten Tust, Giovanni Marchisio, and James C. Tilton. Learning Bayesian classifiers for scene classification with a visual grammar. *IEEE Transactions on Geoscience and Remote Sensing (TGRS)*, 43(3):581–589, March 2005.
- [Alt95] Mark L. G. Althouse. Image segmentation by local entropy methods. In *Proc. IEEE Intl. Conf. on Image Processing (ICIP 1995)*, Washington, D.C., USA, October 1995. IEEE.
- [Aqu97] Aquater. Evaluation studies on the identification of vines using aerial photography. Technical report, Aquater, Italy, December 1997. European Commission Joint Research Center, Space Applications Institute, Agricultural Information Systems Unit.
- [AS02] B. Aslan and G. Sech. Comparison of different goodness-of-fit tests. In *Conf. on Advanced Statistical Techniques in Particle Physics*, Grey College, Durham, UK, March 2002. University of Durham.

- [ASH03] Jes s Angulo, Jean Serra, and Allan Hanbury. Morphologie math matique et couleur. Talk at the GDR ISIS colloquia,  cole Nationale Sup rieure de T l communications, May 2003. (In French).
- [AZW00] Noureddine Abbadeni, Djemel Ziou, and Shengrui Wang. Autocovariance-based perceptual textural features corresponding to human visual perception. In ICPR 2000 [ICP00], pages 913–916.
- [BAF05] Charles M. Bachmann, Thomas L. Ainsworth, and Robert A. Fusina. Exploiting manifold geometry in hyperspectral imagery. *IEEE Transactions on Geoscience and Remote Sensing (TGRS)*, 43(3):441–454, March 2005.
- [Bai97] Caroline Baillard. *Analyse d'images a riennes st reo pour la restitution 3-D en milieu urbain*. PhD thesis,  cole Nationale Sup rieure des T l communications, Paris, France, October 1997.
- [Bar04] S. Bard. Quality assessment of cartographic generalisation. *Transactions in GIS*, 8:63–81, 2004.
- [BCA99] Nicolas Boichis, J.-P. Cocquerez, and S. Airault. Road intersection automatic extraction from aerial images: Application to the roundabout. In *Proc. of the 7th Intl. Conf. on Computer Vision (ICCV 1999)*, Kerkyra, Greece, September 1999. IEEE.
- [BCG98] Christophe Biernacki, Gilles Celeux, and G rard Govaert. Assessing a mixture model for clustering with the integrated classification likelihood. Technical Report RR-3521, INRIA, <http://www.inria.fr>, October 1998.
- [BDI+03] L. Bovino, G. Dimauro, S. Impedovo, M. G. Lucchese, R. Modugno, G. Pirlo, and A. Salzo. On the combination of abstract-level classifiers. *International Journal on Document Analysis and Recognition*, 6(1):42–54, August 2003.
- [Ber02] Pavel Berkhin. Survey of clustering data mining techniques. Technical report, Accrue Software, San Jose, California, USA, 2002.
- [Bes86] J. Besag. On the statistical analysis of dirty pictures. *Journal of the Royal Statistical Society, Series B*, 48:259–302, 1986.
- [Bez81] J. C. Bezdek. *Pattern Recognition with Fuzzy Objective Function Algorithms*. Plenum, New York, USA, 1981.
- [BF00] Jeff Berens and Graham D. Finlayson. Log-opponent chromaticity coding of colour space. In ICPR 2000 [ICP00], pages 206–211.
- [BG04] Peter Bajcsy and Peter Groves. Methodology for hyperspectral band selection. *Photogrammetric Engineering and Remote Sensing*, 70(7):793–802, July 2004.
- [BGYW02] Jovan G. Brankov, Nikolas P. Galatsanos, Yongyi Yang, and Miles N. Wernick. Similarity based clustering using the expectation maximization algorithm. In ICIP 2002 [ICI02].
- [BHS98] Alizera Bab-Hadiashar and David Suter. Robust range segmentation. In ICPR 1998 [ICP98], pages 969–971.
- [Bis04] Christopher M. Bishop. Bayesian conditional random fields. Plenary session at International Conference on Pattern Recognition, Cambridge, United Kingdom, August 2004.

-
- [BJZ03] Karen Brady, Ian Jermyn, and Josiane Zerubia. Adaptive wavelet packet models for texture description and segmentation. Presentation at GDR ISIS conference on texture, ENST, Paris, June 2003.
- [BKAA04] Faysal Boughorbel, Andreas Koschan, Besma Abidi, and Mongi Abidi. Gaussian energy functions for registration without correspondences. In ICPR 2004 [ICP04].
- [BMT04] Ester Bernadó Mansilla and Kam Ho Tin. On classifier domains of competence. In ICPR 2004 [ICP04].
- [Boe81] Barry Boehm. *Software Engineering Economics*. Advances in Computing Science & Technology. Prentice-Hall, 1981.
- [Boe00] Stefan Boettcher. Extremal optimization: Heuristics via coevolutionary avalanches. *Computing in Science & Engineering*, November/December 2000.
- [Boi00] Nicolas Boichis. *Extraction automatique des carrefours routiers dans les images aériennes guidée par une base de données cartographique*. PhD thesis, Université de Cergy-Pontoise, France, March 2000.
- [Bon95] A. Bonafonte. *Comprensión del habla en Tareas Semánticamente Restringidas*. PhD thesis, Escola Tècnica Superior d'Enginyeria de Telecomunicació de Barcelona, Universitat Politècnica de Catalunya, Jordi Girona, 1-3, 08034 Barcelona, Spain, 1995.
- [BP01a] Stefan Boettcher and Allon G. Percus. Extremal optimization for graph partitioning. *Physical Review E*, 64:026114, 2001.
- [BP01b] Stefan Boettcher and Allon G. Percus. Optimization with extremal dynamics. *Physical Review Letters*, 86(23):5211–5214, June 2001.
- [BPA03] Jon Atli Benediktsson, Martino Pesaresi, and Kolbeinn Arnason. Classification and feature extraction for remote sensing images from urban areas based on morphological transformations. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 41(9):1940–1949, September 2003.
- [BPPD04] O. Bentrach, N. Paparoditis, and M. Pierrot-Deseilligny. Stereopolis: an image-based urban environment modeling system. In *Proc. of the 4th International Symposium on Mobile Mapping Technology*, Kunming, China, 2004.
- [BPPDH04] O. Bentrach, N. Paparoditis, M. Pierrot-Deseilligny, and R. Horaud. Estimating sensor pose from images of a stereo rig. In *Proc. of the ISPRS conference*, Istanbul, Turkey, July 2004. IAPRS.
- [BPS05] Jón Atli Benediktsson, Jón Aegar Palmason, and Johannes R. Sveinsson. Classification of hyperspectral data from urban areas based on extended morphological profiles. *IEEE Transactions on Geoscience and Remote Sensing (TGRS)*, 43(3):480–491, March 2005.
- [BS98] Thomas Bülow and Gerald Sommer. Quaternionic gabor filters for local structure classification. In ICPR 1998 [ICP98], pages 808–810.
- [BS03] Jon Atli Benediktsson and Johannes R. Sveinsson. Multisource remote sensing data classification based on consensus and pruning. *IEEE Transactions on Geoscience and Remote Sensing (TGRS)*, 41(4):932–936, April 2003.

- [Bus] Franco Buseti. Genetic algorithms overview. <http://citeseer.ist.psu.edu/buseti01genetic.html>.
- [BYKC03] Lori Mann Bruce, Nick H. Younan, Roger L. King, and Anil Cheriyyadat. Spectral reduction image processing techniques. In IGARSS 2003 [IGA03].
- [Can97] F. Canters. Evaluating the uncertainty of area estimates derived from fuzzy land-cover classification. *Photogrammetric Engineering and Remote Sensing*, 63(4):403–414, 1997.
- [CB03] Anil Cheriyyadat and Lori Mann Bruce. Why principal component analysis is not an appropriate feature extraction method for hyperspectral data. In IGARSS 2003 [IGA03].
- [CBB05] Jocelyn Chanussot, Patrick Bas, and Lionel Brombrun. Airborne remote sensing of vineyards for the detection of dead vine trees. In *Proc. Intl. Geoscience and Remote Sensing Symposium (IGARSS 2005)*, Seoul, Korea, August 2005. IEEE GRSS.
- [CBD⁺03] Pascal Cachier, Eric Bardinet, Didier Dormont, Xavier Pennec, and Nicholas Ayache. Iconic feature based nonrigid registration: the PASHA algorithm. *Computer Vision and Image Understanding*, 89(2/3):272–298, February/March 2003.
- [CBP03] Jocelyn Chanussot, Jon Atli Benediktsson, and Martino Pesaresi. On the use of morphological alternated sequential filters for the classification of remote sensing images from urban areas. In IGARSS 2003 [IGA03].
- [CDH04] Jason J. Corso, Maneesh Dewan, and Gregory D. Hager. Image segmentation through energy minimization based subspace fusion. In ICPR 2004 [ICP04].
- [CEL⁺04] S. Chabrier, B. Emile, H. Laurent, C. Rosenberger, and P. Marché. Unsupervised evaluation of image segmentation; application to multi-spectral images. In ICPR 2004 [ICP04].
- [CF99] C. J Chung and A. G. Fabbri. Probabilistic prediction models for landslide hazard mapping. *Photogrammetric Engineering and Remote Sensing*, 65:1389–1399, 1999.
- [CFSV03] D. Conte, P. Foggia, C. Sansone, and M. Vento. Graph matching applications in pattern recognition and image processing. In ICIP 2003 [ICI03].
- [Chi03] Salim Chitroub. Optimal fusion-classification of multisource remote sensing imagery using global optimization and fuzzy logic. In IGARSS 2003 [IGA03].
- [CJSW01] H. D. Cheng, X. H. Jiang, Y. Sun, and Jingli Wang. Color image segmentation: advances and prospects. *Pattern Recognition*, 34:2259–2281, 2001.
- [CL94] Q. Chen and J. Y. S. Luh. Ambiguity reduction by relaxation labeling. *Pattern Recognition*, 27:165–180, 1994.
- [CL95] Q. Chen and J. Y. S. Luh. Relaxation labeling algorithm for information integration and its convergence. *Pattern Recognition*, 28:1705–1722, 1995.
- [CM03] M. E. Cablk and T. B. Minor. Detecting and discriminating impervious cover with high-resolution IKONOS data using principal component analysis and morphological operators. *International Journal of Remote Sensing*, 24(23):4627–4645, December 2003.

-
- [CMV02] Y. Caron, P. Makris, and N. Vincent. A method for detecting artificial objects in natural environments. In ICPR 2002 [ICP02].
- [CPDJS02] N. Chehata, M. Pierrot-Deseilligny, F. Jung, and G. Stamon. Extraction of 3D primitives from stereopairs of satellite images for automatic reconstruction of buildings. In *Proc. of the Conf. on Machine Vision and Applications (MVA 2002)*, Japan, December 2002.
- [CPMR04] Junqing Chen, Thrasyvoulos N. Pappas, Aleksandra Mojsilovic, and Bernice E. Rogowitz. Perceptually-tuned multiscale color-texture segmentation. In ICIP 2004 [ICI04].
- [CR03] Haili Chui and Anand Rangarajan. A new point matching algorithm for non-rigid registration. *Computer Vision and Image Understanding*, 89(2/3):114–141, February/March 2003.
- [Cra46] Harald Cramér. *Mathematical Methods of Statistics*. Princeton Mathematical Series. Princeton University Press, Princeton, USA, 1946.
- [CRH02] Marco Carcassoni, Eraldo Ribeiro, and Edwin R. Hancock. Texture recognition through modal analysis of spectral peak patterns. In ICPR 2002 [ICP02].
- [CS03] DongMei Chen and Douglas Stow. Strategies for integrating information from multiple spatial resolutions into land-use/land-cover classification routines. *Photogrammetric Engineering and Remote Sensing*, 69(11):1279–1287, November 2003.
- [CTL04] Chi Kin Chow, Hung Tat Tsui, and Tong Lee. Surface registration using a dynamic genetic algorithm. *Pattern Recognition*, 37(1):105–117, January 2004.
- [CVA03] Hua-Mei Chen, Pramod K. Varshney, and Manoj K. Arora. Performance of mutual information similarity measure for registration of multitemporal remote sensing images. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 41(11):2445–2454, 2003.
- [CW04] A. Carleer and E. Wolff. Exploitation of very high resolution satellite data for tree species identification. *Photogrammetric Engineering and Remote Sensing*, 70(1):135–140, January 2004.
- [CZ04] Songcan Chen and Yulian Zhu. Subpattern-based principle component analysis. *Pattern Recognition (rapid and brief communications)*, 37(5):1081–1083, May 2004.
- [DBFR⁺04] C. Di Bella, R. Faivre, F. Ruget, B. Seguin, M Guérif, B. Combal, M. Weiss, and C. Rebella. Remote sensing capabilities to estimate pasture production in France. *International Journal of Remote Sensing*, 25(23):5359–5372, December 2004.
- [DC03] C. Duchêne and C. Cambrier. Cartographic generalisation using cooperative agents. In *Proc. of the 2nd Conference on Autonomous Agents and Multi-Agent Systems (AAMAS 2003)*, 2003.
- [DC04] Huawu Deng and David A. Clausi. Unsupervised image segmentation using a simple MRF model with a new implementation scheme. In ICPR 2004 [ICP04].
- [DCGB02] Jean Pierre Da Costa, Christian Germain, and Pierre Baylou. Orientation difference statistics for texture description. In ICPR 2002 [ICP02].

- [dFCC00] Luciano da Fontoura Costa and Roberto Marcondes Cesar, Jr. *Shape Analysis and Classification: Theory and Practice*. Image Processing. CRC Press, December 2000.
- [DH73] R. Duda and P. Hart. *Pattern Classification and Scene Analysis*. Wiley, 1973.
- [Dij59] E.W. Dijkstra. A note on two problems in connexion with graphs. *Numerical Mathematics*, 1:269–271, 1959.
- [DK82] Pierre A. Devijver and Josef Kittler. *Pattern Recognition: A Statistical Approach*. Prentice Hall, 1982.
- [DMZB96] Xavier Descombes, Robin Morris, Josiane Zerubia, and Marc Berthod. Estimation of Markov random field prior parameters using Markov chain Monte Carlo maximum likelihood. Technical Report RR-3015, INRIA, <http://www.inria.fr>, October 1996.
- [DPDPC04] M. Deveau, M. Pierrot-Deseilligny, N. Paparoditis, and X. Chen. Relative laser scanner and image pose estimation using points and segments. In *Proc. of the ISPRS conference, Istanbul, Turkey, July 2004*. IAPRS.
- [DPT04] Robert P. W. Duin, E. Pełalska, and David M. J. Tax. The characterization of classification problems by classifier disagreements. In ICPR 2004 [ICP04].
- [DR96] Kim Dralle and Mats Rudemo. Stem number estimation by kernel smoothing of aerial photos. *Canadian Journal of Forest Research*, 26:1228–1236, 1996.
- [DR01] Pornphan Dulyakarn and Yuttapong Rangsanseri. Fuzzy *c*-means clustering using spatial information with application to remote sensing. In *Proc. of the 22nd Asian Conference on Remote Sensing*, Singapore, November 2001. Submitted. (<http://www.crisp.nus.edu.sg/acrs2001/pdf/113RANGS.PDF>).
- [DSDCM02] C. De Stefano, A. Della Cioppa, and A. Marcelli. An adaptive weighted majority vote rule for combining multiple classifiers. In ICPR 2002 [ICP02].
- [Duc01] C. Duchêne. Road generalisation using agents. In *Proc. of 9th Annual Conference on GIS Research*, Glamorgan, United Kingdom, 2001.
- [Duc03] C. Duchêne. Automated map generalisation using communicating agents. In *Proc. of the 21st International Conference on Cartography (ICA 2003)*, pages 160–169, Durban, South Africa, 2003.
- [Dui02] Robert P. W. Duin. The combining classifier: to train or not to train? In ICPR 2002 [ICP02].
- [DWC04] A. J. W. De Wit and J. G. P. W. Clevers. Efficiency and accuracy of per-field classification for operational crop mapping. *International Journal of Remote Sensing*, 25(20):4091–4112, October 2004.
- [DZ01a] Benoit Dubuc and Steven W. Zucker. Complexity, confusion and perceptual grouping. Part I: The curve-like representation. *International Journal of Computer Vision*, 42(1/2):55–82, April/May 2001.
- [DZ01b] Benoit Dubuc and Steven W. Zucker. Complexity, confusion and perceptual grouping. Part II: Mapping complexity. *International Journal of Computer Vision*, 42(1/2):83–115, April/May 2001.

-
- [EM03a] Francisco Eugenio and Ferran Marqués. Automatic satellite image georeferencing using a contour-marching approach. *IEEE Transactions on Geoscience and Remote Sensing (TGRS)*, 41(12):2869–2880, December 2003.
- [EM03b] Francisco Eugenio and Ferran Marqués. Automatic satellite image georeferencing using a contour-matching approach. *IEEE Transactions on Geoscience and Remote Sensing (TGRS)*, 41(12):2869–2880, December 2003.
- [Epa69] V. A. Epanechnikov. Nonparametric estimates of a multivariate probability density. *Theory of Probability and its Applications*, 14:153–158, 1969.
- [Fau79] O.D. Faugeras. Digital color image processing within the framework of a human visual model. *ASSP*, 27:380–393, 1979.
- [FB81] O. D. Faugeras and M. Berthod. Improving consistency and reducing ambiguity in stochastic labeling: An optimization approach. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 3:412–423, April 1981.
- [FBB03] Samuel Foucher, Jean-Marc Boucher, and Goze B. Béné. Multiscale classification and filtering of SAR images using Dempster-Shafer theory. In *IGARSS 2003 [IGA03]*.
- [FC04] Jeff Fortuna and David Capson. Improved support vector classification using PCA and ICA feature space modification. *Pattern Recognition*, 37(6):1116–1129, June 2004.
- [FCB05] Mathieu Fauvel, Jocelyn Chanussot, and Jon Atli Benediktsson. Fusion of methods for the classification of remote sensing images from urban areas. In *Proc. Intl. Geoscience and Remote Sensing Symposium (IGARSS 2005)*, Seoul, Korea, August 2005. IEEE GRSS.
- [FCTW92] G. M. Foody, N. A. Campbell, N. M. Trodd, and T. F. Wood. Derivation and applications of probabilistic measures of class membership from maximum-likelihood classification. *Photogrammetric Engineering and Remote Sensing*, 58(9):1335–1341, 1992.
- [FH85] Pascal Fua and Andrew J. Hanson. Locating cultural regions in aerial imagery using geometric cues. In *Proc. of the Image Understanding Workshop*, pages 271–278. DARPA, 1985.
- [FH87] Pascal Fua and Andrew J. Hanson. Resegmentation using generic shape: Locating general cultural objects. *Pattern Recognition Letters*, 5:243–252, March 1987.
- [Fin00] Graham D. Finlayson. Computational colour constancy. In *ICPR 2000 [ICP00]*, pages 191–196.
- [FL03] Ana L. N. Fred and José M. N. Leitão. A new cluster isolation criterion based on dissimilarity increments. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 25(8):944–958, August 2003.
- [Fla95] Julien Flack. *Interpretation of Remotely Sensed Data Using Guided Techniques*. PhD thesis, School of Computer Science, Curtin University of Technology, Australia, November 1995.
- [FMML04] Jordi Freixenet, Xavier Muñoz, Joan Martí, and Xavier Lladó. Colour texture segmentation by region-boundary cooperation. In *Proc. of the 8th European Conf. on Computer Vision (ECCV 2004)*, Prague, Czech Republic, May 2004.

- [Fon03] I. L. Fonseca. The impact of data normalisation on unsupervised continuous classification of landforms. In IGARSS 2003 [IGA03].
- [FP03] David A. Forsyth and Jean Ponce. *Computer Vision, a modern approach*. Prentice Hall, 2003.
- [FS02] Guoliang Fan and Xiaomu Song. A study of contextual modeling and texture characterization for multiscale Bayesian segmentation. In ICIP 2002 [ICI02].
- [FSS⁺03] F. del Frate, G. Schiavon, D. Solimini, M. Borgeaud, D. H. Hoekman, and M. A. M. Vissers. Crop classification using multiconfiguration C-band SAR data. *IEEE Transactions on Geoscience and Remote Sensing (TGRS)*, 41(7):1611–1619, July 2003.
- [FW96] Reda E. Fayek and Andrew K. C. Wong. Extracting buildings from aerial topographic maps. In *Proc. IEEE Intl. Conf. on Image Processing (ICIP 1996)*, volume 2, pages 401–404, Lausanne, Switzerland, September 1996. IEEE.
- [FY97] Alan M. N. Fu and Hong Yan. A new probabilistic relaxation method based on probability space partition. *Pattern Recognition*, 30(11):1905–1917, 1997.
- [Gar02] Aurélie Garnier. Modélisation de textures dans les images aériennes. Master’s thesis, École Supérieure en Sciences Informatiques, 930, rue des Colles; 06903 Sophia-Antipolis, France, September 2002. (In French) (<http://www.essi.fr>).
- [Gaw70] Louis C. Gawthrop, editor. *The Administrative Process and Democratic Theory*. Houghton Mifflin Co, 1970.
- [GB03] Sidharta Gautama and Alexander Borghgraef. Using graph matching to compare VHR satellite images with GIS data. In IGARSS 2003 [IGA03].
- [GdGES03] S. Giada, T. de Groeve, D. Ehrlich, and P. Soille. Information extraction from very high resolution satellite imagery over Lukole refugee camp, Tanzania. *International Journal of Remote Sensing*, 24(22):4251–4266, November 2003.
- [GEYR04] K. P. Gallo, C. D. Elvidge, L. Yang, and B. C. Reed. Trends in night-time city lights and vegetation indices associated with urbanization within the conterminous USA. *International Journal of Remote Sensing (letters)*, 25(10):2003–2007, May 2004.
- [GG84] Stewart Geman and D. Geman. Stochastic relaxation, Gibbs distributions and the Bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 6(6):721–741, 1984.
- [Gho00] Sugata Ghosal. On algebraic smoothing: Theory and results. In ICPR 2000 [ICP00], pages 21–24.
- [Gif04] Sigrid Giffon. Classification grossière par radiométrie et texture en zones agricoles. Master’s thesis, Université Jean Monnet, Saint Étienne, France, July 2004. (In French).
- [GLL99] J. Gustafson, N. Lindberg, and M. Lundeberg. The August spoken language system. In *6th European Conference on Speech Communication and Technology (Eurospeech’99)*, pages 1155–1158, Budapest, Hungary, 1999.
- [GLMC] L. Guigues, H. Le Men, and J.-P. Cocquerez. Codage optimal d’observations de \mathbb{R}^n selon un modèle gaussien. exemple d’application à la segmentation d’images. Article draft; <mailto:laurent.guigues@ign.fr>.

-
- [GLMC03a] L. Guigues, H. Le Men, and J.-P. Cocquerez. Analyse et représentation ensembles-échelle d'une image. In *Proc. 19th GRETSI Symposium on Signal and Image Processing*, Paris, France, September 2003.
- [GLMC03b] L. Guigues, H. Le Men, and J.-P. Cocquerez. Scale-sets image analysis. In *ICIP 2003 [ICI03]*.
- [GN02] M. Girvan and M. E. J. Newman. Community structure in social and biological networks. *Proceedings of the National Academy of Sciences (PNAS)*, 99(12):7821–7826, June 2002.
- [Gos88] Ardeshir Goshtasby. Registration of images with geometric distortions. *IEEE Transactions on Geoscience and Remote Sensing (TGRS)*, 26(1):60–64, jan 1988.
- [GP03] S. E. Grigorescu and N. Petkov. Texture analysis using Rényi's generalized entropies. In *ICIP 2003 [ICI03]*.
- [GPK00] S. E. Grigorescu, N. Petkov, and P. Kruizinga. A comparative study of filter based texture operators using mahalanobis distance. In *ICPR 2000 [ICP00]*, pages 897–900.
- [GRH⁺03] Darrel R. Greenhill, Lennart T. Ripke, Adrian P. Hitchman, Graeme A. Jones, and Graeme G. Wilkinson. Characterization of suburban areas for land use planning using landscape ecological indicators derived from Ikonos-2 multi-spectral imagery. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 41(9):2015–2021, September 2003.
- [GS03] Th. Gevers and H. M. G. Stokman. Robust photometric invariant region detection in multispectral images. *International Journal of Computer Vision*, 53(2):135–151, July 2003.
- [GSST03] Ardeshir Goshtasby, Lawrence Staib, Colin Studholme, and Demetri Terzopoulos. Nonrigid image registration: guest editors' introduction. *Computer Vision and Image Understanding*, 89(2/3):109–113, February/March 2003.
- [GTSC⁺06] Laurent Guigues, Roger Trias-Sanz, Nesrine Chehata, Franck Taillandier, and Matthieu Deveau. Segmentation multi-échelles d'images: théorie et applications. *Bulletin d'Information Scientifique et Technique de l'Institut Géographique National*, 75:41–57, 2006. (In French).
- [Gui00] Laurent Guigues. Un algorithme pour la détection de zones de végétation sur images aériennes. Oral communication to GT6.2 "Complex systems for image analysis" of GdR ISIS (CNRS/MENRT), ENST, Paris, France, October 2000. (<http://www-isis.enst.fr/Kiosque/journees/Annonces/annonce-reunion-119.html>).
- [Gui03] Laurent Guigues. Causal minimal cuts in hierarchies of regions and multi-scale image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 2003. Submitted to Special Issue on Energy Minimization Methods in Computer Vision and Pattern Recognition.
- [Gui04] Laurent Guigues. *Modèles Multi-Échelles pour la Segmentation d'Images*. PhD thesis, École Doctorale Sciences et Ingénierie de l'Université de Cergy-Pointoise, France, December 2004.

- [GZD04] Bert Guindon, Ying Zhang, and Craig Dillabaugh. Landsat urban mapping based on a combined spectral-spatial methodology. *Remote Sensing of Environment*, 92(2):218–232, August 2004.
- [HCR04] Kyung-Soo Han, Jean-Louis Champeaux, and Jean-Louis Roujean. A land cover classification product over France at 1 km resolution using SPOT4/VEGETATION data. *Remote Sensing of Environment*, 92(1):52–66, July 2004.
- [HCSZ04] A. Hafiane, S. Chaudhuri, G. Seetharaman, and B. Zavidovique. Image retrieval using \ast -trees for GIS applications. In *Proc. of the 8th World Conf. on Systemics, Cybernetics and Informatics (SCI)*, Orlando, USA, July 2004.
- [HCW93] Tao-I. Hsu, A. D. Calway, and R. Wilson. Texture analysis using the multiresolution fourier transform. In *Proc. of the 8th Scandinavian Conf. on Image Analysis (SCIA 1993)*, Tromsø, Norway, May 1993.
- [HD98] Christine Hivernat and Xavier Descombes. Mise en correspondance et recalage de graphes: application aux réseaux routiers extraits d'un couple carte/image. Technical Report RR-3529, INRIA, <http://www.inria.fr>, October 1998.
- [HEBF04] Reija Haapanen, Alan R. Ek, Marvin E. Bauer, and Andrew O. Finley. Delineation of forest/nonforest land use classes using nearest neighbor methods. *Remote Sensing of Environment*, 89(3):265–271, February 2004.
- [HEMK98] Kostas Haris, Serafim N. Efstratiadis, Nicos Maglaveras, and Aggelos K. Kat-saggelos. Hybrid image segmentation using watersheds and fast region merging. *IEEE Transactions on Image Processing*, 7(12):1684–1699, December 1998.
- [HH02] Michal Haindl and Vojtěch Havlíček. A multiscale colour texture model. In ICPR 2002 [ICP02].
- [HH03] Petter Holme and Mikael Huss. Discovery and analysis of biochemical subnetwork hierarchies. Preprint arXiv Quantitative Biology - Molecular Networks 0309011 v1, 23 Sep 2003, September 2003.
- [HSD73] R. M. Haralick, K. Shanmugam, and I. Dinstein. Textural features for image classification. *IEEE Transactions on Systems, Man, and Cybernetics (SMC)*, 3:610–621, 1973.
- [HSIW96] A. Hilton, A. J. Stoddart, J. Illingworth, and T. Windeatt. Marching triangles: Range image fusion for complex object modelling. In *Proc. IEEE Intl. Conf. on Image Processing (ICIP 1996)*, volume 2, pages 381–384, Lausanne, Switzerland, September 1996. IEEE.
- [HW89] Christopher E. Heil and David F. Walnut. Continuous and discrete wavelet transforms. *SIAM Review*, 31(4):628–666, December 1989.
- [Hye03] Anaïs Hyenne. Comptage automatique d'oliviers par traitement d'images satellites haute résolution. Master's thesis, École Nationale des Sciences Géographiques and École Nationale Supérieure des Télécommunications, Université de Marne la Vallée, France, June 2003.
- [HZ83] R. A. Hummel and S. W. Zucker. On the foundations of relaxation labeling process. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 5(3):267–286, May 1983.

-
- [HŽ98] Michal Haindl and Pavel Žid. Fast segmentation of planar surfaces in range images. In ICPR 1998 [ICP98], pages 985–987.
- [ICI02] *Proc. IEEE Intl. Conf. on Image Processing (ICIP 2002)*, Rochester, New York, USA, September 2002. IEEE.
- [ICI03] *Proc. IEEE Intl. Conf. on Image Processing (ICIP 2003)*, Barcelona, Spain, September 2003. IEEE.
- [ICI04] *Proc. IEEE Intl. Conf. on Image Processing (ICIP 2004)*, Singapore, October 2004. IEEE.
- [ICP98] *Proc. 14th Intl. Conf. on Pattern Recognition (ICPR 1998)*, Brisbane, Australia, August 1998. IEEE Computer Society.
- [ICP00] *Proc. 15th Intl. Conf. on Pattern Recognition (ICPR 2000)*, Barcelona, Spain, September 2000. IAPR.
- [ICP02] *Proc. 16th Intl. Conf. on Pattern Recognition (ICPR 2002)*, Québec City, Québec, Canada, August 2002. IAPR.
- [ICP04] *Proc. 17th Intl. Conf. on Pattern Recognition (ICPR 2004)*, Cambridge, UK, August 2004. IAPR.
- [IGA03] *Proc. Intl. Geoscience and Remote Sensing Symposium (IGARSS 2003)*, Toulouse, France, July 2003. IEEE GRSS.
- [Ing93] L. Ingber. Simulated annealing: Practice versus theory. *Mathematical Computer Modelling*, 18(11):29–57, 1993.
- [Jib02] Hassan Jibrini. *Reconstruction de modèles de bâtiments à partir de données cadastrales vectorisées et d'un couple d'images aériennes à haute résolution*. PhD thesis, École Nationale Supérieure de Télécommunications, Paris, France, April 2002.
- [JK98] Xiaoyi Jiang and Peter Kühni. Search-based contour closure in range images. In ICPR 1998 [ICP98], pages 16–18.
- [JM92] Jean-Michel Jolion and Annick Montanvert. The adaptive pyramid, a framework for 2D image analysis. *CVGIP: Image Understanding*, 55(3):339–348, May 1992.
- [Jou02] A. G. Journel. Combining knowledge from diverse sources: an alternative to traditional data independence hypothesis. *Mathematical Geology*, 34:573–596, 2002.
- [JS02] A. M. Jakomulska and M. N. Stawiecka. Integrating spectral and textural information for land cover mapping. In Gerard Bégni, editor, *Observing our environment from Space: New solutions for a new millennium. Proc. of the 21st EARSeL Symposium*, pages 347–354. Balkema, 2002.
- [JTLB04] Anil K. Jain, Alexander Topchy, Martin H. C. Law, and Joachim M. Buhmann. Landscape of clustering algorithms. In ICPR 2004 [ICP04].
- [Jun99] Dieter Jungnickel. *Graphs, Networks and Algorithms*, volume 5 of *Algorithms and Computation in Mathematics*. Springer, Berlin, 1999.
- [KC89] A. Khotanzad and J. Y. Chen. Unsupervised segmentation of textured images by edge detection in multidimensional features. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 11(4):414–421, 1989.

- [KDN⁺95] Tapas Kanungo, Byron Dom, Wayne Niblack, David Steele, and Jacob Sheinvald. MDL-based multi-band image segmentation using a fast region merging scheme. Technical Report RJ 9960 (87919) Computer Science, IBM Research Division, Almaden Research Center, San Jose, CA 95120-6099, USA, May 1995.
- [KGV83] S. Kirkpatrick, C. Gelatt, and M. Vecchi. Optimization by simulated annealing. *Science*, 220:671–680, 1983.
- [KH03] Alireza Khotanzad and Orlando J. Hernandez. Color image retrieval using multispectral random field texture model and color content features. *Pattern Recognition*, 36(8):1679–1694, August 2003.
- [KKC05] Alexey Kostin, Josef Kittler, and William Christmas. Object recognition by symmetrised graph matching using relaxation labelling with an inhibitory mechanism. *Pattern Recognition Letters*, 26:381–393, 2005.
- [KLH03] Sanjiv Kumar, Alexander C. Loui, and Martial Hebert. An observation-constrained generative approach for probabilistic classification of image regions. *Image and Vision Computing*, 21:87–97, 2003.
- [KLM94] G. Koepfler, C. Lopez, and J.M. Morel. A multiscale algorithm for image segmentation by variational methods. *SIAM Journal on Numerical Analysis*, 31(1):282–299, February 1994.
- [Koh01] T. Kohonen. *Self-Organizing Maps*, volume 30 of *Springer Series in Information Sciences*. Springer, third edition, 2001.
- [KP99] Peter Kruizinga and Nikolay Petkov. Nonlinear operator for oriented texture. *IEEE Transactions on Image Processing*, 8(10):1395–1407, oct 1999.
- [KP00] P. Kruizinga and N. Petkov. A nonlinear texture operator specialised in the analysis of dot-patterns. In ICPR 2000 [ICP00], pages 197–201.
- [KPQ02] Zoltan Kato, Ting-Chuen Pong, and Song Guo Qiang. Multicue MRF image segmentation: Combining texture and color features. In ICPR 2002 [ICP02].
- [KSF05] F. P. Kressler, K. Steinnocher, and M. Franzen. Object-oriented classification of orthophotos to support update of spatial databases. In *Proc. Intl. Geoscience and Remote Sensing Symposium (IGARSS 2005)*, Seoul, Korea, August 2005. IEEE GRSS.
- [LBMN00] Florence Le Ber, Ludmila Mangelinck, and Amedeo Napoli. Un système de reconnaissance d’organisations spatiales agricoles sur images satellitaires. In *Proc. 12ème Congrès de Reconnaissance de Formes et Intelligence Artificielle (RFIA 2000)*, volume 1, pages 119–128, Paris, France, February 2000. AFRIF-AFIA.
- [LDZ98] A. Lorette, X. Descombes, and J. Zerubia. Extraction des zones urbaines fondée sur une analyse de la texture par modélisation markovienne. Technical Report RR-3423, INRIA, <http://www.inria.fr>, May 1998.
- [LE02] Huitao Luo and Alexandros Eleftheriadis. Rubberband: An improved graph search algorithm for interactive object segmentation. In ICIP 2002 [ICI02].
- [Lec89] Y.G. Leclerc. Constructing simple stable descriptions for image partitioning. *International Journal of Computer Vision*, 3(1):73–102, 1989.

-
- [Lei03] J. K. Lein. Applying evidential reasoning methods to agricultural land cover classification. *International Journal of Remote Sensing*, 24(21):4161–4180, November 2003.
- [LGW04] Weiguo Liu, Sucharita Gopal, and Curtis E. Woodcock. Uncertainty and confidence in land cover classification using a hybrid classifier approach. *Photogrammetric Engineering and Remote Sensing*, 70(8):963–971, August 2004.
- [LH03] Bin Luo and E. R. Hancock. A unified framework for alignment and correspondence. *Computer Vision and Image Understanding*, 92(1):26–55, October 2003.
- [LIH⁺04] V. Lacroix, M. Idrissa, A. Hincq, H. Bruynseels, and O. Swartenbroekx. SPOT5 images for urbanization detection. In ACIVS 2004 [ACI04].
- [LMH04] Baihua Li, Qinggang Meng, and Horst Holstein. New method for sparse point-sets matching with underlying non-rigidity. In ICPR 2004 [ICP04].
- [Lon98] Sven Lončarić. A survey of shape analysis techniques. *Pattern Recognition*, 31(8):983–1001, August 1998.
- [LS04] A. Leemans and J. Sijbers. Multiresolutional rigid-body registration of space curves. In ACIVS 2004 [ACI04].
- [LSLH02] Fei Liu, Xiaodan Song, Yupin Luo, and Dongcheng Hu. Unsupervised Mumford-Shah energy based hybrid of texture and nontexture image segmentation. In ICIP 2002 [ICI02].
- [LY94] J. Liu and Y.-H. Yang. Multiresolution color image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 16(7):689–700, 1994.
- [Mac91] Bruce MacLennan. Gabor representations of spatiotemporal visual images. Technical Report CS-91-144, Computer Science Department, University of Tennessee, Knoxville, TN 37996, YSA, September 1991. Revised October 21, 1994. (<http://citeseer.nj.nec.com/macLennan94gabor.html>).
- [Mal89] S. Mallat. A theory for multiresolution signal decomposition: the wavelet representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 11:674–693, July 1989.
- [Mal97] S. Mallat. *A wavelet tour of signal processing*. Academic Press, 1997.
- [Man04] Roberto Manduchi. Learning outdoor color classification from just one training image. In *Proc. of the 8th European Conf. on Computer Vision (ECCV 2004)*, Prague, Czech Republic, May 2004.
- [Mar82] D. Marr. *Vision*. Freeman and Co., 1982.
- [Mar87] Jr. Marple, S. L. *Digital Spectral Analysis with Applications*. Prentice-Hall, 1987.
- [MAR98] MARS Project. Vinident Study final report: Methodological test for the identification and mapping of vineyards using aerial photography. Technical report, Geosys and Tragsatec, 1998? Contract n. 126169702 F1ED ISP, European Commission Joint Research Center, Space Applications Institute, Agricultural Information Systems Unit.

- [MASB03] Jose L. Marroquin, Edgar Arce Santana, and Salvador Botello. Hidden Markov measure field models for image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 25(11):1380–1387, November 2003.
- [MASB04] Cicero Mota, Til Aach, Ingo Stuke, and Erhardt Barth. Estimation of multiple orientations in multi-dimensional signals. In *ICIP 2004 [ICI04]*.
- [MBB02] Abhinav Mathur, Lori Mann Bruce, and John Byrd. Discrimination of subtly different vegetative species via hyperspectral data. In *Proc. Intl. Geoscience and Remote Sensing Symposium (IGARSS 2002)*, Toronto, Canada, 2002. IEEE GRSS.
- [MBM03] F. Murtagh, D. Barreto, and J. Marcello. Decision boundaries using Bayes factors: The case of cloud masks. *IEEE Transactions on Geoscience and Remote Sensing (TGRS)*, 41(12):2952–2958, December 2003.
- [mCVLhL04] Hua mei Chen, Pramod K. Varshney, Jiancong Luo, and Ting hung Lin. A global optimization scheme for mutual information based remote sensing image registration. In *ACIVS 2004 [ACI04]*.
- [MDK04] Vadim Mottl, Sergey Dvoekno, and Andrey Kopylov. Pattern recognition in interrelated data: The problem, fundamental assumptions, recognition algorithms. In *ICPR 2004 [ICP04]*.
- [MDZ96] Robin Morris, Xavier Descombes, and Josiane Zerubia. An analysis of some models used in image segmentation. Technical Report RR-3016, INRIA, <http://www.inria.fr>, October 1996.
- [MFM04] David R. Martin, Charless C. Fowlkes, and Jitendra Malik. Learning to detect natural image boundaries using local brightness, color, and texture cues. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 26(5):530–549, May 2004.
- [MGBdC04] F. Michelet, C. Germain, P. Baylou, and J. P. da Costa. Local multiple orientation estimation: Isotropic and recursive oriented network. In *ICPR 2004 [ICP04]*.
- [Mic95] Z. Michalewicz. *Genetic Algorithms + Data Structures = Evolution Programs*. Springer, New York, USA, third edition, 1995.
- [MK97] K. Messer and Josef Kittler. A comparison of colour texture attributes selected by statistical feature selection and neural network methods. *Pattern Recognition*, 18:1241–1246, 1997.
- [MKIZ98] Makoto Maeda, Kousuke Kumamaru, Katsuhiko Inoue, and Hongbin Zha. Surface recovery by using regularization theory and its application to multi-resolution analysis. In *ICPR 1998 [ICP98]*, pages 19–23.
- [MLK04] Takaharu Miyoshi, Weiqing Li, and Kazufumi Kaneda. Automatic extraction of buildings utilizing geometric features of a scanned topographic map. In *ICPR 2004 [ICP04]*.
- [MLT04] Soe Win Myint, Nina S.-N. Lam, and John M. Tyler. Wavelets for urban spatial feature discrimination: Comparisons with fractal, spatial autocorrelation, and spatial co-occurrence approaches. *Photogrammetric Engineering and Remote Sensing*, 70(7):803–812, July 2004.
- [MM96] B. S. Manjunath and W. Y. Ma. Texture features for browsing and retrieval of image data. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 18(8):837–842, August 1996.

-
- [MMP00] A. M. Massone, F. Masulli, and A. Petrosino. Fuzzy clustering algorithms on LANDSAT images for detection of waste areas: A comparison. In *Advances in Fuzzy Systems and Intelligent Technologies*, pages 165–175, Maastricht, The Netherlands, 2000. Shaker.
- [MNL03] Adilson E. Motter, Takashi Nishikawa, and Ying-Cheng Lai. Large-scale structural organization of social networks. *Physical Review E*, 68:036105, 2003.
- [MP00] Majid Mirmehdi and Maria Petrou. Segmentation of color textures. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 22(2):142–159, feb 2000.
- [MP04] Topi Mäenpää and Matti Pietikäinen. Classification with color and texture: jointly or separately? *Pattern Recognition*, 37(8):1629–1640, August 2004.
- [MPV02] Topi Mäenpää, Matti Pietikäinen, and Jaako Viertola. Separating color and pattern information for color texture discrimination. In ICPR 2002 [ICP02].
- [MRH04] Alexei D. Miasnikov, Jayson E. Rome, and Robert M. Haralick. A hierarchical projection pursuit clustering algorithm. In ICPR 2004 [ICP04].
- [MS89] David Mumford and Jiambo Shah. Optimal approximations by piecewise smooth functions and associated variational problems. *Communications on Pure and Applied Mathematics*, 17(4):577–685, 1989.
- [MSC04] Jonathan Milgram, Robert Sabourin, and Mohamed Cheriet. Two-stage classification system combining model-based and discriminative approaches. In ICPR 2004 [ICP04].
- [MSK04] Marina Mueller, Karl Segl, and Hermann Kaufmann. Edge- and region-based segmentation technique for the extraction of large, man-made objects in high-resolution satellite imagery. *Pattern Recognition*, 37(8):1619–1628, August 2004.
- [MYGM+04] V. Meas-Yedid, E. Glory, E. Morelon, Ch. Pinset, G. Stamon, and J-Ch. Olivo-Marin. Automatic color space selection for biological image segmentation. In ICPR 2004 [ICP04].
- [New03a] M. E. J. Newman. Fast algorithm for detecting community structure in networks. Preprint arXiv Condensed Matter 0309508 v1, 22 Sep 2003, September 2003.
- [New03b] M. E. J. Newman. The structure and function of complex networks. *SIAM Review*, 45(2):167–256, 2003.
- [New04] G. N. Newsam. Edge and line detection as exercises in hypothesis testing. In ICIP 2004 [ICI04].
- [NG02] M. E. J. Newman and M. Girvan. Mixing patterns and community structure in networks. Preprint arXiv Condensed Matter 0210146 v1, 7 Oct 2002, October 2002.
- [NG03] M. E. J. Newman and M. Girvan. Finding and evaluating community structure in networks. Preprint arXiv Condensed Matter 03082187 v2, 11 Aug 2003, August 2003.

- [NHHA04] Vincent Noblet, Christian Heinrich, Fabrice Heitz, and Jean-Paul Armspach. A topology preserving non-rigid registration method using a symmetric similarity function-application to 3-D brain images. In *Proc. of the 8th European Conf. on Computer Vision (ECCV 2004)*, Prague, Czech Republic, May 2004.
- [Nin03] Yoshiki Ninomiya. A stabilized vegetation index and several mineralogic indices defined for ASTER VNIR and SWIR data. In IGARSS 2003 [IGA03].
- [NN01] S. Noronha and R. Nevatia. Detection and modeling of buildings from multiple aerial images. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 23(5):501–518, May 2001.
- [NRC+00] R. Neelamani, J. Romberg, H. Choi, R. Riedi, and R. Baraniuk. Multiscale image segmentation using joint texture and shape analysis. In *Proc. of the SPIE Conf. on Wavelet Applications in Signal and Image Processing*, San Diego, California, USA, August 2000.
- [NSK02] Hideki Noda, Mahdad N. Shirazi, and Eiji Kawaguchi. Mrf-based texture segmentation using wavelet decomposed images. *Pattern Recognition*, 35:771–782, 2002.
- [NYC00] Jose Nicolás, María J. Yzuel, and Juan Campos. Colour information as a third dimension in fourier transform and correlation. In ICPR 2000 [ICP00], pages 515–518.
- [OB94] Jonathan J. Oliver and Rohan A. Baxter. MML and Bayesianism: Similarities and differences. Technical Report 206, Department of Computer Science, Monash University, Clayton, Victoria 3168, Australia, December 1994. Amended August 15th, 1995. (<http://citeseer.nj.nec.com/oliver94mml.html>).
- [OBFW98] J. M. Orwell, J. F. Boyce, Haddonm J. F., and G. H. Watson. Detecting periodic structure. In ICPR 1998 [ICP98], pages 714–716.
- [OH94] Honathan H. Oliver and David Hand. Introduction to minimum encoding inference. Technical Report 4-94, Department of Statistics, Open University, Walton Hall, MK7 6AA, UK, July 1994. Amended November 14th 1994. Also Monash Univ. CS Dept. technical report 205. (<http://citeseer.nj.nec.com/oliver94introduction.html>).
- [OPM02] Timo Olaja, Matti Pietikäinen, and Topi Mäenpää. Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 24(7):971–987, July 2002.
- [Ots79] N. Otsu. A threshold selection method from grey-level histograms. *IEEE Transactions on Systems, Man, and Cybernetics (SMC)*, 9(1):62–66, 1979.
- [Pal04] Christoph Palm. Color texture classification by integrative co-occurrence matrices. *Pattern Recognition*, 37(5):965–976, May 2004.
- [Par06] Vilfredo Pareto. *Manuale d'economia politica*. Società Editrice Libreria, Milano, Italy, 1906.
- [Par62] E. Parzen. On estimation of a probability density and mode. *Annals of Mathematical Statistics*, 35:1065–1076, 1962.

-
- [Pav78] Theodosios Pavlidis. A review of algorithms for shape analysis. *Computer Graphics and Image Processing*, 7:243–258, 1978.
- [PB03] Pedro Pina and Teresa Barata. Classification by mathematical morphology. In IGARSS 2003 [IGA03].
- [PBA03] Jon Aevor Palmason, Jon Atli Benediktsson, and Kolbeinn Arnason. Morphological transformations and feature extraction for urban data with high spectral and spatial resolution. In IGARSS 2003 [IGA03].
- [PC03] N. W. Park and K. H. Chi. A probabilistic approach to predictive spatial data fusion for geological hazard assessment. In IGARSS 2003 [IGA03].
- [PDZ00] Olivier Pony, Xavier Descombes, and Josiane Zerubia. Classification d'images satellitaires hyperspectrales en zone rurale et périurbaine. Technical Report RR-4008, INRIA, <http://www.inria.fr>, September 2000.
- [Pha02] Dzung L. Pham. Fuzzy clustering with spatial constraints. In ICIIP 2002 [ICI02].
- [PI97] Markus Peura and Jukka Iivarinen. Efficiency of simple shape descriptors. In *Proc. of the 3rd International Workshop on Visual Form*, Capri, Italy, May 1997.
- [PKBP02] D. A. Pouliot, D. J. King, F. W. Bell, and D. G. Pitt. Automated tree crown detection and delineation in high-resolution camera imagery of coniferous forest regeneration. *Remote Sensing of Environment*, 82(2-3):322–334, October 2002.
- [PMPP04] Antonio Plaza, Pablo Martinez, Rosa Perez, and Javier Plaza. A new approach to mixed pixel classification of hyperspectral imagery based on extended morphological profiles. *Pattern Recognition*, 37(6):1097–1116, June 2004.
- [PMR03] Gintautas Palubinskas, Rupert Müller, and Peter Reinartz. Radiometric normalization of optical remote sensing imagery. In IGARSS 2003 [IGA03].
- [PNHA84] S. Peleg, J. Naor, R. Hartley, and D. Avnir. Multiple resolution texture analysis and classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 6:518–523, 1984.
- [PP93] Nikhil R. Pal and Sankar K. Pal. A review on image segmentation techniques. *Pattern Recognition*, 26(9):1277–1294, 1993.
- [PSZ04] G. Poggi, G. Scarpa, and J. Zerubia. Segmentation of remote-sensing images by supervised TS-MRF. In ICIIP 2004 [ICI04].
- [PTJ00] N. Paparoditis, C. Thom, and H. Jibrini. Surface reconstruction in urban areas from multiple views of aerial digital frames. In *IAPRS*, volume 33, Amsterdam, The Netherlands, 2000.
- [Pun03] Chi-Man Pun. Rotation-invariant texture feature for image retrieval. *Computer Vision and Image Understanding*, 89(1):24–43, January 2003.
- [Qui93] J. R. Quinlan. *C4.5: Programs for machine learning*. Morgan Kaufman, San Mateo, California, US, 1993.
- [Rab89] Lawrence R. Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, February 1989.

- [RANB98] Thierry Ranchin, Michel Albuissou, Bernard Naert, and Gilbert Boyer. The VINIDENT study: Evaluation studies on the identification of vines using aerial photography in France. Technical Report TM/97/R/19, École des Mines de Paris, Paris, France, April 1998. VINIDENT Studies – Contract n. 12606-97-02 F1ED ISP F - Armines; Contract Armines n. 110.0049.
- [RCC+03] Filippo Radicchi, Claudio Castellano, Federico Coconci, Vittorio Loreto, and Domenico Parisi. Defining and identifying communities in networks. Preprint arXiv Condensed Matter 0309488 v1, 21 Sep 2003, September 2003.
- [RDG+03] Dar A. Roberts, Philip E. Dennison, Margaret E. Gardner, Yasha Hetzel, Susan L. Ustin, and Christopher T. Lee. Evaluation of the potential of Hyperion for fire danger assessment by comparison to the Airborne Visible/Infrared Imaging Spectrometer. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 41(6):1297–1310, June 2003.
- [RHZ76] A. Rosenfeld, R. Hummel, and S. Zucker. Scene labeling by relaxation operations. *IEEE Transactions on Systems, Man, and Cybernetics (SMC)*, 6:320–433, June 1976.
- [Ric05] John A. Richards. Analysis of remotely sensed data: The formative decades and the future. *IEEE Transactions on Geoscience and Remote Sensing (TGRS)*, 43(3):422–432, March 2005.
- [RM97] A. Ruas and W. Mackaness. Strategies for urban map generalisation. In *Proc. of the 18th International Conference on Cartography (ICA 1997)*, pages 1387–1394, Stockholm, Sweden, 1997.
- [RMWAB01] Jean-Marc Robbez-Masson, Tom Wassenaar, Patrick Andrieux, and Frédéric Baret. Reconnaissance par télédétection rapprochée des vignes et analyse de leur structure spatiale à l’aide d’une analyse fréquentielle intra-parcellaire. *Ingénieries*, 27:59–67, set 2001.
- [RNA+99] Thierry Ranchin, Bernard Naert, Michel Albuissou, Gilbert Boyer, and Pär Åstrand. An automatic method for vine detection in airborne imagery using wavelet transform and multiresolution analysis. *Photogrammetric Engineering and Remote Sensing*, dec 1999.
- [Ros56] M. Rosenblatt. Remarks on some nonparametric estimates of a density functions. *Annals of Mathematical Statistics*, 27:642–669, 1956.
- [RP96] A. Ruas and C. Plazanet. Strategies for automated map generalisation. In *Proc. of the 7th international Symposium on Spatial Data Handling (SDH 1996)*, pages 319–336, Delft, The Netherlands, 1996.
- [Rua01] A. Ruas. Automating the process of generalisation, the age of maturity? In *Proc. of the 20th International Conference on Cartography (ICA 2001)*, pages 1943–1953, Beijing, China, 2001.
- [Rua02] A. Ruas. Spatial analysis and agent principles to automate generalization process. In *International workshop on semantic processing of spatial data (GeoPro 2002)*, Mexico, 2002.
- [SBS04] Carlos Santamaria, Mirosław Bober, and Wiesław Szajnowski. Texture analysis using level-crossing statistics. In *ICPR 2004 [ICP04]*.

-
- [Sch78] G. Schwarz. Estimating the dimension of a model. *The Annals of Statistics*, 6:461–464, 1978.
- [Sch02] Bernt Schiele. How many classifiers do i need? In ICPR 2002 [ICP02].
- [SD03] Aaron K. Shackelford and Curt H. Davis. A hierarchical fuzzy classification approach for high-resolution multispectral data over urban areas. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 41(9):1920–1931, September 2003.
- [SDS99] P. C. Smits, S. G. Dellepiane, and R. A. Schowengerdt. Quality assessment of image classification algorithms for land-cover mapping: a review and a proposal for a cost-based approach. *International Journal of Remote Sensing*, 20(8):1461–1486, 1999.
- [SF03] Vincent Simonneaux and Pierre François. Identifying main crop classes in an irrigated area using high resolution image time series. In IGARSS 2003 [IGA03].
- [SFW03] Annemarie Schneider, Mark A. Friedl, and Curtis E. Woodcock. Mapping urban areas by fusing multiple sources of coarse resolution remotely sensed data. In IGARSS 2003 [IGA03].
- [SG00] Philippe Salembier and Luis Garrido. Binary partition tree as an efficient representation for image processing, segmentation, and information retrieval. *IEEE Transactions on Image Processing*, 9(4):561–576, April 2000.
- [SH81] Linda G. Shapiro and Robert M. Haralick. Structural descriptions and inexact matching. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 3(5):504–519, September 1981.
- [Sha76] G. Shafer. *A Mathematical Theory of Evidence*. Princeton University Press, Princeton, New Jersey, USA, 1976.
- [Shi04] Peg Shippert. Why use hyperspectral imagery? *Photogrammetric Engineering and Remote Sensing*, 70(4):377–380, April 2004.
- [Sil86] B. W. Silverman. *Density Estimation for Statistics and Data Analysis*. Chapman and Hall, New York, USA, 1986.
- [SKK03] K. Steinnocher, F. P. Kressler, and M. Köstl. Modelling population pressure in sub-urban and rural regions based on remote sensing and statistical data. In IGARSS 2003 [IGA03].
- [SKS95] T. Shimbashi, Y. Kokubo, and N. Shiota. Region segmentation using edge based circle growing. In *Proc. IEEE Intl. Conf. on Image Processing (ICIP 1995)*, Washington, D.C., USA, October 1995. IEEE.
- [SL00] Nicu Sebe and Michael S. Lew. Wavelet based texture classification. In ICPR 2000 [ICP00], pages 959–962.
- [SMS02] Maneesha Singh, Markos Markou, and Sameer Singh. Colour image texture analysis: Dependence on colour spaces. In ICPR 2002 [ICP02].
- [SPA03] Miguel Segui Prieto and Alastair R. Allen. A similarity metric for edge images. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 25(10):1265–1272, October 2003.

- [SPZ03] Giuseppe Scarpa, Giovanni Poggo, and Josiane Zerubia. A binary tree-structured MRF model for multispectral satellite image segmentation. Technical Report RR-5062, INRIA, <http://www.inria.fr>, December 2003.
- [SQL04] Stephen South, Jiaguo Qi, and David P. Lusch. Optimal classification methods for mapping agricultural tillage practice. *Remote Sensing of Environment*, 91(1):90–97, May 2004.
- [SS02] Maneesha Singh and Sameer Singh. Spatial texture analysis: A comparative study. In ICPR 2002 [ICP02].
- [SSAO00] Akira Suzuki, Akio Shio, Hiroyuki Arai, and Sakuichi Ohtsuka. Dynamic shadow compensation of aerial images based on color and spatial analysis. In ICPR 2000 [ICP00], pages 317–210.
- [SSK⁺04] K. S. Schmidt, A. K. Skidmore, E. H. Kloosterman, H. van Oosten, L. Kumar, and J. A. M. Janssen. Mapping coastal vegetation using an expert system and hyperspectral imagery. *Photogrammetric Engineering and Remote Sensing*, 6(70):703–715, June 2004.
- [SSP03] Joachin Schöll and Elisabeth Schöll-Paschinger. Classification by restricted random walks. *Pattern Recognition*, 36:1279–1290, jun 2003.
- [Ste97] Klaus Steinnocher. Texturanalyse zur Detektion von Siedlungsgebieten in hochauflösenden panchromatischen Satellitelbilddaten. In F. Dollinger and J. Strobl, editors, *AGIT IX - Salzburger Geographische Materialien*, volume 26, pages 143–152, Salzburg, Austria, 1997. Instituts für Geographie der Universität Salzburg.
- [Str03] Bernd-M. Straub. Automatic extraction of trees from aerial images and surface models. In Baumgartner et al., editors, *Proceedings of ISPRS Conference on Photogrammetric Image Analysis (PIA)*, München, Germany, September 2003. ISPRS.
- [SW03] Conghe Song and Curtis E. Woodcock. Monitoring forest succession with multitemporal Landsat images: Factors of uncertainty. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 41(11):2557–2567, 2003.
- [TA95] Dipankar Talukdar and Raj Acharya. Estimation of fractal dimension using alternating sequential filters. In *Proc. IEEE Intl. Conf. on Image Processing (ICIP 1995)*, Washington, D.C., USA, October 1995. IEEE.
- [TD02] F. Taillandier and R. Deriche. Reconstruction de primitives linéaires 3-D en multi-vues pour la modélisation de scènes urbaines. In *Proc. 13ème Congrès de Reconnaissance de Formes et Intelligence Artificielle (RFIA 2002)*, Angers, France, January 2002. AFRIF-AFIA.
- [THC03] Nancy Thomas, Chad Hendrix, and Russel G. Congalton. A comparison of urban mapping methods using high-resolution digital imagery. *Photogrammetric Engineering and Remote Sensing*, 69(9):963–972, 2003.
- [THK⁺00] J. R. G. Townshend, C. Huang, S. N. V. Kalluri, R. S. Defries, S. Liang, and K. Yang. Beware of per-pixel characterization of land cover. *International Journal of Remote Sensing*, 21(4):839–843, 2000.
- [TK02] Seishi Takamura and Naoki Kobayashi. Practical extension to CIELuv color space to improve uniformity. In ICIP 2002 [ICI02].

-
- [TLT00] Yo Tao, Ernest C. M. Lam, and Yuan Y. Tang. Extraction of fractal feature for pattern recognition. In ICPR 2000 [ICP00], pages 527–530.
- [TMBJP04] Alexander Topchy, Behrouz Minaei-Bidgoli, Anil K. Jain, and William F. Punch. Adaptive clustering ensembles. In ICPR 2004 [ICP04].
- [TP05] Sakari Tuominen and Anssi Pekkarinen. Performance of different spectral and textural aerial photograph features in multi-source forest inventory. *Remote Sensing of Environment*, 94(2):256–268, January 2005.
- [TS04a] Roger Trias-Sanz. An edge-based method for registering a graph onto an image with application to cadastre registration. In ACIVS 2004 [ACI04], pages 333–340.
- [TS04b] Roger Trias-Sanz. An edge-based method for registering a graph onto an image with application to cadastre registration. Technical Report 2004/1, SIP-CRIP5 (Université de Paris 5) and Institut Géographique National, 45 rue des Saints-Pères, Paris, France, March 2004. (In French).
- [TS04c] M. Turker and B. T. San. Detection of collapsed buildings caused by the 1999 Izmit, Turkey earthquake through digital analysis of post-event aerial photographs. *International Journal of Remote Sensing*, 25(21):4701–4714, November 2004.
- [TS05a] Roger Trias-Sanz. A metric for evaluating and comparing hierarchical and multi-scale image segmentations. In *Proc. of the 2005 International Geosciences And Remote Sensing Symposium (IGARSS 2005)*, Seoul, South Korea, July 2005.
- [TS05b] Roger Trias-Sanz. A texture orientation estimator for discriminating between forests, orchards, vineyards, and tilled fields. In *Proc. of the 2005 International Geosciences And Remote Sensing Symposium (IGARSS 2005)*, Seoul, South Korea, July 2005.
- [Tsa03] Victor J. D. Tsai. Automatic shadow detection and radiometric restoration on digital aerial images. In IGARSS 2003 [IGA03].
- [TSB05] Roger Trias-Sanz and Didier Boldo. A high-reliability, high-resolution method for land cover classification into forest and non-forest. In Heikki Kälviäinen, Jussi Parkkinen, and Arto Kaarna, editors, *Proc. of the 14th Scandinavian Conference on Image Analysis (SCIA 2005)*, volume 3540 of *Lecture Notes in Computer Science*, pages 831–840, Joensuu, Finland, June 2005. Springer.
- [TSH03] Sotaro Tanaka, Toshiro Sugimura, and Hideki Hashiba. An estimation of seasonal true color of vegetation cover for satellite image mosaic using color transfer cube (CTC). In IGARSS 2003 [IGA03].
- [TSL03] Roger Trias-Sanz and Nicolas Loménie. Automatic bridge detection in high-resolution satellite images. In J. L. Crowley et al., editors, *Proc. of the 3rd International Conference on Computer Vision Systems (ICVS 03)*, volume 2626 of *Lecture Notes in Computer Science*, pages 172–181, Graz, Austria, April 2003. Springer.
- [TSPD04] Roger Trias-Sanz and Marc Pierrot-Deseilligny. A region-based method for graph to image registration with application to cadastre data. In ICIP 2004 [ICI04].

- [TvBDK00] David M. J. Tax, Martijn van Breukelen, Robert P. W. Duin, and Josef Kittler. Combining multiple classifiers by averaging or by multiplying? *Pattern Recognition*, 33:1475–1485, 2000.
- [TZS⁺03] Shihao Tang, Qijiang Zhu, Yanmin Shuai, Donghui Xie, and Gongle Zhou. A new vegetation index and its principle and application. In IGARSS 2003 [IGA03].
- [UG04] Devrim Unay and Bernard Gosselin. An approach for recognizing stem-end/calix regions in apple quality sorting. In ACIVS 2004 [ACI04].
- [UGR04] Kanuzori Umeda, Goy Godin, and Marc Rioux. Registration of range and color images using gradient constraints and range intensity images. In ICPR 2004 [ICP04].
- [VB00] Constantin Vertan and Nozha Boujemaa. Color texture classification by normalized color space representation. In ICPR 2000 [ICP00], pages 584–587.
- [VD00] C. Vestri and F. Devernay. Using robust methods for automatic extraction of buildings. In *Proc. of the 2000 Conf. on Computer Vision and Pattern Recognition (CVPR 2000)*, Hawaii, USA, 2000. IEEE Computer Society.
- [VdWSLVD99] G. Van de Wouwer, P. Scheunders, S. Livens, and D. Van Dyck. Wavelet correlation signatures for color texture characterization. *Pattern Recognition*, 32(3):443–451, March 1999.
- [VdWSVD98] G. Van de Wouwer, P. Scheunders, and D. Van Dyck. Rotation-invariant texture characterization using isotropic wavelet frames. In ICPR 1998 [ICP98], pages 814–816.
- [VF04] Olli Virtajoki and Pasi Fränti. Divide-and-conquer algorithm for creating neighborhood graph for clustering. In ICPR 2004 [ICP04].
- [VG02] Jean-Marc Viglino and Laurent Guigues. Géoréférencement automatique de feuilles cadastrales. In *Proc. 13ème Congrès de Reconnaissance de Formes et Intelligence Artificielle (RFIA 2002)*, pages 135–143, Angers, France, January 2002. AFRIF-AFIA.
- [VJ01a] Paul Viola and Michael Jones. Rapid object detection using a boosted cascade of simple features. In *Proc. of the 2001 Conf. on Computer Vision and Pattern Recognition (CVPR 2001)*. IEEE Computer Society, 2001.
- [VJ01b] Paul Viola and Michael J. Jones. Robust real-time object detection. Technical Report 2001/01, Compaq Cambridge Research Laboratory, Cambridge, Massachusetts, USA, February 2001.
- [VKR03] Y. V. Venkatesh and S. Kumar Raja. On the classification of multispectral satellite images using the multilayer perceptron. *Pattern Recognition*, 36(9):2161–2175, sep 2003.
- [VKS03] Iris Vanhamel, A. Katartzis, and Hichem Sahli. Hierarchical segmentation via a diffusion scheme in color-texture feature space. In ICIP 2003 [ICI03].
- [VMP03] Nicolas Vandenbroucke, Ludovic Macaire, and Jack-Gérard Postaire. Color image segmentation by pixel classification in an adapted hybrid color space. application to soccer image analysis. *Computer Vision and Image Understanding*, 90(2):190–216, May 2003.

-
- [VPS03] Iris Vanhamel, Ioannis Pratihakis, and Hichem Sahli. Multiscale gradient watersheds of color images. *IEEE Transactions on Image Processing*, 12(6):617–626, June 2003.
- [VSGP04] Ewout Vansteenkiste, Abram Schoutteet, Sidharta Gautama, and Wilfried Philips. Comparing color and textural information in very high resolution satellite image classification. In ICIP 2004 [ICI04].
- [Wal98] Volker Walter. Automatic classification of remote sensing data for GIS database revision. In *International Archives of Photogrammetry and Remote Sensing (IAPRS)*, volume 32, pages 641–648, Stuttgart, Germany, 1998.
- [Wan98] Jia-Ping Wang. Stochastic relaxation on partitions with connected components and its application to image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 20(6):619–636, June 1998.
- [Was01] Tom Wassenaar. *Reconnaissance des états de surface du sol en milieu viticole méditerranéen par télédétection à très haute résolution spatiale*. PhD thesis, École Nationale Supérieure Agronomique de Montpellier, Montpellier, France, March 2001.
- [Wea99] David L. Weakliem. A critique of the Bayesian Information Criterion for model selection. *Sociological Methods & Research*, 27(3):359–397, February 1999.
- [WF02] Slawo Wesolkowski and Paul Fieguth. Adaptive color image segmentation using Markov random fields. In ICIP 2002 [ICI02].
- [WG83] C. S. Wallace and M. P. Georgeff. A general objective for inductive inference. Technical Report 32, Department of Computer Science, Monash University, Clayton, Victoria 3168, Australia, March 1983. (<http://www.csse.monash.edu.au/lloyd/tildeMML/Structured/TR32/>).
- [Wil05] Graeme G. Wilkinson. Results and implications of a study of fifteen years of satellite image classification experiments. *IEEE Transactions on Geoscience and Remote Sensing (TGRS)*, 43(3):433–440, March 2005.
- [WLL04] Jia Wang, Hanqing Lu, and Qingshan Liu. Tensor voting toward feature space analysis. In ICPR 2004 [ICP04].
- [WPPM04] Gang Wu, Chunhong Pan, Veronique Prinnet, and Songde Ma. A land use classification method based on region and edge information fusion. In ICIP 2004 [ICI04].
- [WS05] Timothy A. Warner and Karen Steinmaus. Spatial classification of orchards and vineyards with high spatial resolution panchromatic imagery. *Photogrammetric Engineering and Remote Sensing*, 71(2):179–187, February 2005.
- [WSK03] Adam Winstanley, Bashir Salaik, and Laura Keyes. Statistical language models for topographic data recognition. In IGARSS 2003 [IGA03].
- [XUM04] Q. Xiao, S. L. Ustin, and E. G. McPherson. Using AVIRIS data and multiple-masking techniques to map urban forest species. *International Journal of Remote Sensing*, 25(24):5637–5654, December 2004.
- [YBG97] Shan Yu, Marc Berthod, and Gérard Giraudon. Towards robust analysis of satellite images using map information — application to urban area detection. Technical Report RR-3293, INRIA, <http://www.inria.fr>, November 1997.

- [YC00] A. L. Yuille and James M. Coughlan. Fundamental limits of Bayesian inference: Order parameters and phase transitions for road tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 22(2):160–173, February 2000.
- [YCWZ01] A. L. Yuille, James M. Coughlan, Yingnian Wu, and Song Chun Zhu. Order parameters for detecting target curves in images: When does high level knowledge help? *International Journal of Computer Vision*, 41(1/2):9–34, January/February 2001.
- [YPP04] Zhanwu Yu, Veronique Prinet, and Chunhong Pan. A novel two-steps strategy for automatic GIS-image registration. In ICIP 2004 [ICI04].
- [YS04] Stella X. Yu and Jianbo Shi. Segmentation given partial grouping constraints. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 26(2):173–183, February 2004.
- [Zad65] L. A. Zadeh. Fuzzy sets. *Information and Control*, 8:338–353, 1965.
- [ZFW04] Chengqian Zhang, Steven E. Franklin, and Michael A. Wulder. Geostatistical and texture analysis of airborne-acquired images used in forest classification. *International Journal of Remote Sensing*, 25(4):859–865, February 2004.
- [ZKH78] S. W. Zucker, E. V. Krishnamurthy, and R. L. Haar. Relaxation processes for scene labeling: convergence, speed and stability. *IEEE Transactions on Systems, Man, and Cybernetics (SMC)*, 8:41–48, 1978.
- [ZL04] Dengsheng Zhang and Guojun Lu. Review of shape representation and description techniques. *Pattern Recognition*, 37(1):1–19, January 2004.
- [ZLY95] Song Chun Zhu, Tai Sing Lee, and Alan L. Yuille. Region competition: Unifying snakes, region growing, energy/bayes/MDL for multi-band image segmentation. In *Proc. of the 5th Intl. Conf. on Computer Vision (ICCV 1995)*, Boston, Massachusetts, USA, June 1995. IEEE.
- [ZPM03] Zhaohui Zhang, Veronique Prinet, and Songde Ma. Water body extraction from multi-source satellite images. In IGARSS 2003 [IGA03].
- [ZT02] Jianguo Zhang and Tieniu Tan. Brief review of invariant texture analysis methods. *Pattern Recognition*, 35:735–747, 2002.
- [ZT03] Jianguo Zhang and Tieniu Tan. Affine invariant classification and retrieval of texture images. *Pattern Recognition*, 36:657–664, 2003.
- [ZTM02] Jianguo Zhang, Tieniu Tan, and Li Ma. Invariant texture segmentation via circular gabor filters. In ICPR 2002 [ICP02].
- [ZWGS03] Q. Zhang, J. Wang, P. Gong, and P. Shi. Study of urban spatial patterns from SPOT panchromatic imagery using textural analysis. *International Journal of Remote Sensing*, 24(21):4137–4160, November 2003.
- [ZWM97] Song Chun Zhu, Ying Nian Wu, and David Mumford. Minimax entropy principle and its application to texture modeling. *Neural Computation*, 9(9):1627–1660, 1997.
- [ZWM98] Song Chun Zhu, Yingnian Wu, and David Mumford. Filters, random fields and maximum entropy (FRAME): Towards a unified theory for texture modeling. *International Journal of Computer Vision*, 27(2):107–126, 1998.

-
- [ZXZ03] Jie Zhou, Leping Xin, and David Zhang. Scale-orientation histogram for texture image retrieval. *Pattern Recognition*, 36:1061–1062, 2003.
- [ZY95] Song Chun Zhu and A. L. Yuille. Region competition and its analysis: A unified theory for image segmentation. Technical Report 95-07, Harvard Robotics Laboratory, Division of Applied Sciences, Harvard University, Cambridge, Massachusetts, USA, 1995.
- [ZY96] Song Chun Zhu and Alan L. Yuille. Region competition: Unifying snake/balloon, region growing and bayes/MDL/energy for multi-band image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 18(9), September 1996.

E | **What it is all about: executive summary for non-scientists**

As is to be expected of a doctoral thesis, this document is heavy with mathematics, technical descriptions, algorithms, methods, and detailed quantitative evaluations. All this information helps researchers understand and duplicate the results obtained; developers and implementers looking to use these methods for their own applications also need the level of detail given in the previous chapters. However, this thesis, being eminently practical in nature, may be of interest to decision makers in the remote sensing, cartography, and image analysis industries, who, although familiar with the general concepts of image analysis, may not have a scientific or engineering background, or are simply not interested in the technical details.

Furthermore, because this thesis represents three years of work by the author, it has also proved an important learning experience. Although it is typical, in research texts, to treat the author as an abstract entity, the fact remains that I have gained important insights, experience, and knowledge from it. Perhaps, this thesis has shaped me as much as I have shaped it. I will describe what I personally have gained from this thesis, for it may be of interest to decision makers to know the formative value of a Ph.D.

Following a recommendation by the Bernard Gregory association (<http://www.abg.asso.fr>), this chapter is an executive summary of this thesis, as well as an analysis, made from a practical and a personal point of view, of the main benefits of this work, both to me and to the IGN.

Framework

The industrial production of maps, currently, requires that experts examine photographs taken from aeroplanes or satellites and manually add into a database elements representing roads, towns, forests and other objects. This is costly and time consuming. In order to accelerate and make this task less expensive, researchers try to find methods so that computers can analyse these images, automatically interpret them, and modify the database accordingly.

In this thesis, I have developed methods that partially automate this process for vegetation geographic objects (forests, fields, vineyards, orchards, . . .), for rural areas, so that a computer can process all but the most difficult parts, and manual treatment is required for only a small portion of the original data.

Industrial and research context

Other research laboratories Other research teams, both at the IGN image analysis lab (MATIS) and elsewhere, are working on similar automatic image interpretation problems, for other kinds of objects, such as roads, buildings, cities, or rivers. There is ongoing research by Layla Gordon at the British Ordnance Survey on the automatic determination of a tree's species, and Corina Iovan has recently started a thesis at IGN on the automatic classification of vegetation in urban areas. Because private map-making companies are usually not interested in detailed maps, but rather concentrate on low-detail road maps, or world-wide or continent-wide maps, this kind of research is mostly done by public or semi-public mapping agencies. Other stakeholders are forestry companies, which need a way to easily count and evaluate their assets, tax agencies, which need to know the real extent and type of crops, urban planners, and the military.

Enabling developments This kind of research has been made possible in recent times by two developments: First, digital photography and digital image processing, which makes it easy to use computers to automatically process images. Second, the commercial availability of very detailed aerial and satellite images. Previously, only low resolution satellite images were available, so research was done on scales of kilometres rather than the metres used in this thesis. Low resolution remote sensing is still very much in use today because it has important applications on global climate modelling, ecosystem assessment, fire monitoring, and many others.

Industrial context and competition The French National Mapping Agency, IGN, is a company with a unique status. Although it is a public company, its budget and management is separate from that of the State. Instead, the French government pays IGN for some services, in particular, the production of cartographic data for areas with low population density, with little sales potential, and IGN must complement this revenue with other sources. In that respect, IGN acts as a private company of which the French government happens to be an important customer. The main product sold to individual customers are detailed maps on paper, mainly for trekking or biking. To companies or agencies, IGN sells geographic databases and processed aerial images known as orthoimages. IGN has important competition in sales of orthoimages, notably from Istar, and in sales of road maps, but little competition in sales of geographic databases.

Because of its semi-public status, research at IGN is public and available to all. There is no attempt to patent it or keep it secret. Therefore, while I could not protect my developments in these ways, I did not have any problem to present my results in research conferences or publish them in scientific journals. The only intellectual property issues to take care of relate to proper dating and precedence in scientific publications—where the first to publish is deemed to have been the discoverer or inventor—and proper handling of others' intellectual property, for example, in programming.

My reasons for writing this thesis

Committing to a thesis work, which lasts typically three to four years, is an important decision. I chose to invest this time in further learning because it would improve my professional prospects, and it would let me increase my knowledge not only of image analysis techniques but also of project management—because a thesis is a large project for a single person to undertake. An important criterion was that, since this would be applied research carried in an

industrial context, it would make my transition from the academic world to industry smoother, while at the same time increasing my value as a researcher.

Project management

Risk management

There are several risk factors that can cause the failure of such a project, as well as success factors that are required for its success. The three most important risk factors are scientific, burn-out, and financial.

Science Because this research requires the development of new methods and techniques which did not exist before, there is the risk that no such new methods will be found. Scientific research is an unmanaged activity, and whether a solution to a problem is found or not can depend a lot on chance. However, there are ways to maximise the probability of success: First, working in an environment conducive to creation—this includes not being overly distracted by small, everyday tasks, and by unnecessary pressure in the form of unreasonable deadlines. Second, being in a scientifically rich environment—where colleagues with different fields of expertise can complement and help guide each other. Third, having the adequate means, mainly in the form of computing equipment and support departments. Finally, being a part of wider research networks, participating in scientific conferences, collaborating with other research labs, and presenting my results to peers in order to receive their comments. My thesis advisors, my supervisor at IGN, and myself, took care that these factors were present, except for collaborating with external labs, which unfortunately we could not manage to do, but was more than compensated by within-lab team work and by participation in international scientific conferences.

Burn-out The psychological risk is harder to handle. It is a well known fact that many doctoral students experience a depression period in the middle of their thesis, usually caused by looming deadlines, the perceived lack of interesting results, and a general fatigue with the research topic. I managed to avoid this problem, and although it is harder to analyse the reasons, I believe a main contribution is the fact that, through very careful planning which took into account the uncertainties and probable setbacks found in research work, I managed to avoid overpressure and at the same time be aware of constant progress as the research advanced step by step.

Finance The financial risk was apparently limited since IGN has the resources for a three-year project, and I received a salary/scholarship connected with this work that assured my financial stability. However, this scholarship was limited to three years, which is too short for a project of this scope. It was possible, in the end, to obtain a university scholarship extension for some months, but otherwise the project would have had to be severely cut—for it is in the final months that most of the benefits are realised.

Scheduling and supervision

From the beginning of the thesis, I decided that I would plan and track my advancements as well as possible. A thesis is a three-year project, and the only real milestone is the thesis defence at the end so, if no plan is set, at the beginning the deadline is so far away that there

is little pressure, and there is a big risk of wandering around researching unimportant topics. This initial plan had a reasonable amount of slack to allow for setbacks and uncertainties, and I adjusted it from time to time as I moved on. By checking my real situation against the planned one, I could not only determine whether I needed to go faster or could relax a bit, but I could also check whether I was making any progress—something more difficult than it sounds, since day-to-day achievements are tiny compared to the grand goal to be reached at the end of the thesis. I believe that having planned the thesis in such a way greatly helped me avoid the kind of mid-thesis burn-out problems that I described before.

In addition, right from the beginning I took the habit of scheduling periodic meetings with my supervisors. This had an interesting effect: because I was, of course, eager to show some progress at each meeting, I could steer my research myself to avoid useless areas, managing not to spend too much time looking for a solution to an obscure problem or getting lost in the details, when a reformulation of the problem actually led to an easier solution.

Cost and financials

Although it is difficult to estimate the total cost to IGN of this thesis, I will attempt to at least give a rough estimate:

Expenses

Personnel Myself, full-time for 3 years as an IGN researcher, and full-time for five months as a university researcher (because the thesis lasted for almost 3.5 years, but IGN statutes make it impossible to employ me for more than 3 years). One M.Sc. student, as intern, full-time for six months.

Assisting personnel Supervisors, colleagues working for me for a short time, and other IGN departments, an estimated one person-month full-time equivalent.

Computing equipment Office personal computer, for 3 years, amortization time 3 years: 2000€. Shared equipment: two printers, two multi-purpose servers, two workstations, one computation server, shared by 20 people: 600€. Internet access and LAN equipment. One multi-computer cluster, effectively free, since it uses computers belonging to other lab members—and accounted for in their budget—during their idle time.

Input data Research-grade aerial image acquisition mission: cost cannot be disclosed.

Consumables Paper and toner for 15000 printed pages and 15000 photocopies: 500€.

Travel and conferences For six conferences: 8000€.

Infrastructure Building, office space, utilities: 22000€.

Income

Government scholarship As part of a programme to promote applied thesis, the French government pays a grant of 45000€ to *the company* in order to help it employ me.

Calculating the long-term income arising from the results of this thesis is beyond the scope of this section. We can, however, estimate the value of the computer program that embodies the algorithms developed in the thesis:

Using Barry Boehms' Constructive Cost Model (COCOMO) [Boe81], in its basic version, we can estimate the Total Cost to Develop of a computer program.. David A. Wheeler's SLOccount is a tool that scans a set of source code files, determines their programming language and the number of physical source lines of code, and applies the COCOMO model to estimate the monetary cost to develop the program. Although the COCOMO II model would give more accurate estimates, it requires a count of *logical* lines of code, instead of the physical count that SLOccount gives. I used the basic COCOMO model in "organic" mode, which is the development mode that most closely resembles the one used for my programs. Given the following program size

Programming Language	Lines of code	%
C++	52886	97.27%
TCL	935	1.72%
Shell (bash)	519	0.95%
Others	29	0.05%

the estimated Total Cost to Develop is 1 793 744 US\$.

People are often surprised at the high values estimated by the COCOMO model, but it must be noted that they include not just the cost of programming time, but also design and specification, documentation, debugging, and testing, and all overhead such as facilities, equipment, and corporate services such as accounting.

Acquired know-how, professional, and soft skills

Throughout this thesis, I developed many important scientific, professional, technical and soft skills, which I will try to detail in this section.

Scientific and professional skills On the scientific and professional side, I enhanced my capacity to solve complex problems in rational terms. I conducted this research with scientific rigour, that is, with a critical attitude that tends to disregard beliefs and preconceptions and take as true only what has been mathematically or empirically proven. Although full mathematical proofs are unnecessary here, and beyond the scope of this research, extensive evaluation and tests show the actual behaviour of the methods I propose, so informed decisions can be taken on whether or not to use them for a particular application. I also learnt to search through the literature to find relevant information and similar work done previously by other researchers, in order to avoid reinventing the wheel and repeating past mistakes. Finally, this assignment gave me the capacity to easily adapt to new fields —since I was new to remote sensing and image analysis at the beginning— and quickly become specialised in advanced scientific and technical areas.

Soft skills I had already acquired these scientific skills, although to a lesser degree, in previous research work in my undergraduate studies. It is in soft skills that I gained the most from these thesis: Because a thesis is a large long-term project (at least for a single person), I gained organizational skills such as the capacity to manage long-term projects, the capacity to effectively organize and use my ideas and those of others, and the endurance required to go through it.

Because this research was set in a collaborative context, I learned to effectively team-work, both in order to benefit from the knowledge and experience of others, and, as I became more

experienced myself, to help others. I needed to communicate my ideas and results to my supervisors, my co-workers, and my peers at international scientific conferences, both in oral and in writing, in order not only to let them know of the status of my research, but also to gain insights from their comments. However, because a thesis is basically a one-person project, I learned to work independently when necessary, to take long and short-term scientific and engineering decisions on my own—which, of course, includes learning about, and analysing, the different factors involved—and to adapt to different situations and fields of knowledge. Both because of this required independence, and because I was researching in areas which, basically, had not been explored before, I needed to demonstrate creative thinking.

I also gained first-hand experience of an international, competitive, and evolving professional environment, in which academic and research needs—necessary for the long-term development of new products—must coexist with many real-world constraints such as financial resources, schedules and deadlines, inter-site collaboration agreements, confidentiality, and market value. However, my attempts to develop a specific collaboration with an external research lab failed; nonetheless, this helped me understand the importance of the human factor in this kind of collaborations. Related to this, I followed two training programs, one on inter-personal conflict management, and one on communicating effectively and convincingly.

Finally, midway through this thesis I supervised a M.Sc. student who was doing her master's thesis at the laboratory. From this experience I gained some human management skills and some further planning skills. Furthermore, I taught several courses—on C++, Java, and image analysis, three times as the only teacher for the course, and once as a teaching assistant—from which I learnt not only how to communicate my knowledge, but also how to guide and coach students to maximise their learning, and to adapt my explanations to their specific requirements. Since for the C++ course I was also responsible for writing the course slides and preparing the exercises, I gained experience on producing coherent and complete learning materials.

Technical know-how and scientific knowledge Although this chapter concentrates on the human side, a thesis is mainly a scientific research work, so the technical know-how that I gained through it must not be underestimated. These include image analysis techniques such as segmentation, registration, and classification; image concepts such as colour spaces and texture, and methods to use them; mathematical tools such as graphs, probability theory, fuzzy theory, and statistics; and knowledge of domain-specific techniques such as radiometry correction, aerial triangulation, 3D vision, orthoimages, and advanced camera technologies. Finally, I learned how to build and use computer clusters, which I needed to obtain numerical results in reasonable time.

Results, benefits, and impact

The main deliverables of this work are the thesis document, some scientific articles, and computer programs.

These deliverables contain different forms of the main result of this thesis: algorithms and methods for automatically interpreting an aerial image and finding—with high precision—the fields, forests, orchards, vineyards, and other vegetation areas therein. These methods can be used to partially automate the interpretation of images which must be done in order to populate a geographical database, and which must be repeated from time to time to keep the database up to date. These databases are necessary for the production of maps and other cartography products.

Thesis document The thesis document, the document that you are reading, gives all mathematical and algorithmic details of the methods that I have developed, as well as the results of the quantitative evaluations of these methods. This will enable other researchers to build upon this work, in order to improve it, or to use it for other applications, and will also allow them to check its correctness and validity.

Scientific articles During the thesis, I submitted and published eight articles to international conferences. In addition, at the time of writing I have submitted three articles to international scientific journals, which are undergoing review prior to an eventual publication. These articles describe, in summarized form, the intermediate results that I obtained during the thesis. Publishing them before the end of the thesis allows other scientists to benefit from them earlier, and gives me the opportunity to obtain assessments of the quality of my work from field experts. Having an article accepted at a conference or journal is also an important piece of feedback, that indicates that my research is of good quality, of interest to the community, and is going the right way.

Computer programs Finally, because I implemented all these methods in computer programs, these programs will allow IGN to easily use the proposed methods in an industrial production chain.

The value of these programs can be estimated. This is included in the previous “cost and financials” section.

Benefits The benefits to the company, IGN, are three-fold. In the short term, quality scientific research is, on its own, one of the goals of the company: the French government evaluates IGN’s performance based on four indices; one of them is the number of peer-reviewed articles published in journals and major scientific conferences. By pursuing this research and publishing its results, I have therefore contributed to a major source of revenue for the company. My colleagues at IGN’s MATIS lab will also be able to reuse some of my methods for their own research.

Furthermore, my developments are already being used by others at IGN. Emilie Collet used the analysis of texture and colour channels as part of a semi-automatic method for detecting forest boundaries. She is currently working on porting my programs to other platforms as a step towards full industrialization. Arnaud Le Bris is using and extending my classification methods to locate glaciers, snow, rocks, and forests in aerial images of high mountain areas. Initial results show a classification accuracy over 80%, which is remarkable since my system was not designed with that goal in mind. Corina Iovan has recently started a thesis on the automatic classification of vegetation in urban areas, and she will use my work as a starting point. Last but not least, during my work I learned to build and use a multi-computer cluster, so that I could run long computations in the lab’s computers during the night, effectively speeding them 20-fold; without it, evaluations would have been, necessarily, much less comprehensive. Encouraged by that experience, the lab has recently purchased a rack-mounted multi-processor cluster.

Finally, the long-term continuity of IGN depends on its ability to innovate, to develop and offer new cartographic products, and to reduce production costs for existing ones. In that respect, my research, after a short period of industrialization—which I estimate to be between half and one year—will allow IGN to reduce the costs associated with manual interpretation of aerial photographs. Finally, and although it was not an original goal of this thesis, the results are much more detailed than what is actually needed for current maps, because the system uses, as input, images of very high resolution. This means that, if necessary, the system could help

to produce maps of much higher resolution, if such a product was deemed to have commercial interest.

Although typically omitted from such analyses, it is also necessary to study the effects of such work in society as a whole, even if the scope of this work makes such effects limited.

On one hand, there is an effect on the workforce. Automating any task, including this task of aerial image analysis, is likely to make some jobs redundant. There is however an important counterargument to this: partially automated systems, such as the one presented here, also allow the same number of people work faster and more effectively. The available funding at IGN for image interpretation is such that the average map contains data that is 10 years old. There is a clear need to improve this, and it is not feasible, economically, to do it by hiring more photo-interpreters. This is the reason why this thesis and similar studies were commissioned.

On the other hand, society at large may benefit from improvements in remote sensing techniques, such as the ones brought forward in this thesis. Although knowing the exact position of a vineyard may have little effect on most people's health, the general increase of the use of remote sensing that we have witnessed in recent years has a definite impact on areas as diverse as the assessment of earthquake damage, the development of better climate models, the detection of suspicious activity in military nuclear facilities, and the tracking of ongoing tsunamis, forest fires, or oil spills,

Finally, this thesis has also benefited me in several ways. I have already detailed the know-how, scientific and professional knowledge, and soft skills that I have gained through it. In addition, because it was not only a research work but also a full-time job as an industrial researcher, it will help smooth my transition to the workforce, while keeping my scientific level high.

Classification semi-automatique du terrain en zone rurale par télédétection à haute résolution

Cette thèse présente une chaîne d'analyse d'image qui, à partir d'images numériques à haute résolution et à trois ou quatre canaux (50 cm, couleur et, dans certains cas, proche infrarouge), mais aussi en s'appuyant sur le parcellaire cadastral, rend une segmentation des images en parcelles agraires (champs, forêts, vignes, . . .), et une classification de celles-ci, avec une très haute fiabilité, et attribue à chaque segment classifié une mesure qui indique la confiance que le système a en cette classification.

Elle inclut une étude sur l'intérêt de la texture et les espaces de couleur pour la segmentation et la classification, deux méthodes de recalage de graphes sur une image, un modèle de probabilité novateur et ses algorithmes de classification par régions associés, et un estimateur très précis de la période et l'orientation.

Semi-automatic rural land cover classification from high-resolution remote sensing images

This thesis presents a complete image analysis system which, from high-resolution 3 or 4-channel digital images (50 cm, colour and optionally near infrared), and using the cadastre database, segments the images into agriculturally-homogeneous regions (fields, forests, vines, and so on), and classifies these regions, tagging each classified region with a confidence measure which indicates the system's confidence in each classification.

It includes a study of the value of texture features and transformed colour spaces for segmentation and classification, two methods for registering a graph onto an image, a novel probability model and associated per-region classification algorithms, and a high-precision period and orientation estimator.

Mots clés: classification; analyse d'images; télédétection; occupation du sol; végétation; classification de cultures; recalage du cadastre.

Keywords: classification; image analysis; remote sensing; land cover; vegetation; crop classification; cadastre registration.

Discipline: Sciences de la Vie et de la Matière

Spécialité: Mathématiques-Informatique

Université Paris 5 — René Descartes: 12 rue de l'École de Médecine, Paris, France

Institut Géographique National: 2-4 avenue Pasteur, Saint-Mandé, France

UFR de mathématiques et informatique: 45 rue des Saints Pères, Paris, France

École doctorale "cognition, comportement, conduites humaines": Inst. de Psychologie, 71 avenue E. Vaillant, Boulogne-Billancourt, France

Laboratoire CRIP5: Université René Descartes Paris 5, 45 rue des Saints Pères, Paris, France