J. Martín-Herrero – TECHNIQUES DE BAS NIVEAU EN TRAITEMENT D'IMAGES POUR LA TÉLÉDETECTION DES MILIEUX NON HOMOGÈNES – 2006

Résumé - La télédétection vise à acquérir l'information sur des cibles en étudiant leur réponse aux ondes électromagnétiques. Et partout nous rencontrons des milieux non homogènes et des composites. Connaître comment ces milieux non homogènes répondent a la sonde de télédétection est de la plus grande importance pour la praticabilité même de la télédétection. Le comportement macroscopique d'un composite peut s'exprimer en fonction des caractéristiques macroscopiques de ses constituants, mais d'une manière complexe incluant la géométrie de leur arrangement. Si nous pouvons obtenir le tenseur diélectrique efficace d'un composé, nous pouvons modéliser sa réponse au champ électromagnétique, et donc sa réponse comme cible de télédétection. La nécessité pour inclure la géométrie détaillée du système d'une façon efficace dans des méthodes numériques, ainsi qu'une équivalence entre les images numériques et les modèles de treillis des composites, suggère le recours aux techniques de bas niveau de traitement d'images numériques.
Le cadre de cette thèse est le traitement numérique d'un problème général de télédétection fondée sur le problème électromagnétique d'homogénéisation dans des microstructures. Dans ce contexte, deux techniques de traitement d'images de bas niveau sont présentées, à savoir, une nouvelle méthode pour l'étiquetage des composantes connexes, présentant des améliorations significatives par rapport aux méthodes existantes, et une méthode de codage des configurations locales avec plusieurs caractéristiques la rendant appropriée pour des applications variées. Leurs avantages sont discutés, et des exemples d'application sont fournis au-delà du domaine spécifique étant à leur origine, comme la vision artificielle, le codage d'image, ou encore la synthèse d'image.

Mots clés - traitement d'images, étiquetage des composantes connexes, codage des configurations locales, télédétection, vision artificielle, codage d'image.

Abstract – The aim of remote sensing is obtaining information about targets by studying their response to electromagnetic waves. And everywhere we found non homogeneous media. Knowing how these non homogeneous media respond to the remote sensing probe is of great importance for the very feasibility of remote sensing. The macroscopic behaviour of a composite can be expressed as a function of the macroscopic characteristics of its constituents, but usually in a complex way which includes the geometry of their arrangement. If we are able to obtain the effective permittivity tensor of any given composite, we can model its macroscale response to the electromagnetic field, and therefore its response as a remote sensing target. The necessity of including the detailed geometry of the system in an efficient way in the numerical methods, together with an equivalence between grid models and digital images, suggest the recourse to low level image processing techniques.
The framework of this thesis is the numerical treatment of a general problem in remote sensing based on the electromagnetic problem of homogenization of microstructures. In this context, two low level image processing techniques are presented, a new method for the labelling of connected components, with significant advantages over the classical methods, and a local configuration encoding scheme with characteristics which render it useful for different applications. Their advantages and applicability are discussed, together with some examples of application in fields out of the scope of the specific problem which originated them, namely computer vision, image coding, and image synthesis.

Keywords - image processing, connected component labelling, local configuration encoding, remote sensing, computer vision, image coding.

N° d'ordre : 3166

THÈSE

PRÉSENTÉE A

L'UNIVERSITÉ DE BORDEAUX I

ÉCOLE DOCTORALE DES SCIENCES PHYSIQUES
ET DE L'INGÉNIEUR

Par Julio MARTÍN-HERRERO

POUR OBTENIR LE GRADE DE

DOCTEUR

SPÉCIALITÉ : AUTOMATIQUE, PRODUCTIQUE, SIGNAL ET IMAGE

TECHNIQUES DE BAS NIVEAU EN TRAITEMENT D'IMAGES POUR LA TÉLÉDETECTION DES MILIEUX NON HOMOGÈNES

Soutenue le 2 juin 2006

Après avis de :

MM. J.L. ALBA-CASTRO	Professeur – ETSET Vigo	
P. PINA	Professeur – IST Lisbonne	Rapporteurs

Devant la commission d'examen formée de :

MM. N. CHRISTOV	Professeur – LAGIS Lille	Président
C. GERMAIN	Maître de Conférences – ENITA Bordeaux	Rapporteur
J.L. ALBA-CASTRO	Professeur – ETSET Vigo	Examinateurs
P. BAYLOU	Professeur – ENSEIRB Bordeaux	
M. NAJIM	Professeur – ENSEIRB Bordeaux	
P. PINA	Professeur – IST Lisbonne	

– 2006 –

Low Level Image Processing for Nonhomogeneous Media in Remote Sensing

Julio Martín-Herrero

ESI-LAPS

When I look back, it seems to me
as if this almost miraculous
change of inclination and will
was the immediate suggestion
of the guardian angel of my life,
the last effort made
by the spirit of preservation
to avert the storm
that was even then
hanging in the stars
and ready to envelop me.

People are crazy and times are strange,
I'm locked in tight, I'm out of range.
I used to care, but things have changed.

Preface

Ils ne se rendent pas compte.

The text that follows is to be defended as a PhD Thesis at the École Doctorale des Sciences Physiques et de l'Ingénieur of the Université Bordeaux I, within the speciality Signal et Image.

That of life is the strangest of paths, and we often encounter people and traverse landscapes which we would hardly have ever expected. Specially at the first stages of our journey. For when we have worn out several pairs of boots, we have already trod many sharp rocks and have met many different wills; enough at least as to not be surprised anymore by the unexpected. After a time, you learn a fundamental truth: Even the hardest of stones will erode in the wind; if you want to stand, you should bend as the proverbial bamboo cane.

Behold, this is one of my deepest bows yet. After having obtained a PhD in Physics in 2002 with a thesis on spatially explicit stochastic models on remotely sensed imagery, with a good load of pattern recognition, image processing, and computer graphics, I would hardly have ever expected that only four years later I should have to demonstrate again my capacity for research in any of the aforementioned fields. Moreover, according to my obviously erroneous perception of the scientific world, the unusual multidisciplinarity of my previous research would have hinted, endorsed by a degree in electrical engineering, if anything, a much looked-for ability for multidisciplinary problem solving, so precious in engineering. But I was wrong. Above any other consideration, in the eyes of my colleagues, I was guilty of

one of the worst faults an initiated can commit: I had deserted the ranks of the hyperspecialized, I had shaken off the *orejeras*, I had dared to join the physicists lines. Well, I'm back.

However, at the corners of life you do not only meet hostile beings, but also merry companions and welcoming hosts. I have to gratefully acknowledge the people of ESI-LAPS for their understanding, support, sympathy, and disposition, specially Prof. Mohammed Najim, Christian Germain, and Jean-Pierre da Costa, but also Prof. Pierre Baylou, Prof. Gilbert Grenier, Olivier Lavialle, and Saied Homayouni. Working with them has been a pleasure, which I hope I will keep enjoying in the future. I cannot refer to the LAPS without mentioning Prof. Anselmo Seoane, who knocked at the door on my behalf. My recognition is extended to many people involved in PIMHAI, that project which, more than anything, has established the links for fruitful collaboration with a number of researchers in Britain, France, and Portugal. Even if it were just for this, it was worth the effort.

I have to acknowledge also the School of Computer Science and Software Engineering of The University of Western Australia, and specially my host Prof. Mohammed Bennamoun, for having me and providing me with the nicest of environments, the best of libraries, and the necessary peace of mind to write this report.

I couldn't forget Prof. Jaime Peón, infidel among believers, awake in an age of idiocy; eroded, yes, but still standing in the heights of reason. A physicist!

Perth, Western Australia
Summer 2005/06

Contents

Preface **v**

Résumé **xiii**

1 Introduction **1**
1.1 A world of composites 1
1.2 Effective medium 6
1.2.1 The effective permittivity 8
1.2.2 The numerical method 11
1.3 The role of image processing 15
1.4 Original contributions 19

2 Hybrid Labelling **23**
2.1 Introduction 23
2.1.1 Connectivity in digital images 24
2.1.2 The classical recursive approach 26
2.1.3 The classical iterative approach 27
2.1.4 Recent developments 29
2.2 The algorithm 32
2.3 Benchmark results 40
2.3.1 Algorithmic performance 40
2.3.2 Sample image sets 44
2.3.3 Stack use 52
2.3.4 Speed 59
2.3.5 Performance detecting spanning objects 69
2.3.6 One dimensional arrays for image storage 73
2.3.7 Hybridizing Wolff and Swendsen-Wang 75
2.4 Real world applications 82
2.4.1 A real-time 2D computer vision application 82
2.4.2 A real world 3D computer vision application 89
2.5 Concluding remarks 107

3 Stöhr Edge Encoding **115**
3.1 Introduction . . . 115
3.2 The algorithm . . . 119
3.3 Chain codes . . . 123
3.3.1 Robust code . . . 124
3.3.2 Compact code . . . 130
3.3.3 Template matching with SCC . . . 133
3.3.4 A 3D chain code . . . 143
3.4 Fast machine vision . . . 153
3.4.1 Some tests . . . 155
3.5 Texture analysis . . . 165
3.6 SEE in the numerical method . . . 173
3.6.1 Interpolation and random walks . . . 173
3.6.2 Numerical integration . . . 175

A Computation of the effective response **179**

B Monte Carlo solution of elliptic PDE **185**

C Exodus Monte Carlo **187**

D SCC explicit algorithms **189**

E Numerical results **193**
E.1 2D models . . . 193
E.2 3D models . . . 199

Acknowledgments **205**

Résumé

Les composites sont partout. Tout milieu qui ne se compose pas d'un constituant simple est un composé, ou, dans le jargon de l'électromagnétisme, un milieu non homogène. Par composites (ou milieu microinhomogène, ou milieu avec microstructures) nous nous référons à une structure assemblée à partir d'un nombre très grand de particules de matériaux spécifiques disposés dans une manière spécifique, où l'on assume que chaque particule est considérablement plus petite que la taille du domaine, mais assez grande comparée aux domaines d'application des équations de macro-échelle, c'est à dire de taille plus grande qu'atomique ou moléculaire.

La télédétection vise à acquérir l'information sur des cibles en étudiant leur réponse aux stimuli électromagnétiques. La télédétection nous permet d'obtenir des informations approfondies à grande échelle sur l'environnement, et d'étudier également les galaxies éloignées et les confins de l'univers. Et partout où nous regardons, nous rencontrons des milieux non homogènes et des composites. Les milieux non homogènes parfois ne sont pas les cibles elles-mêmes, mais sont interposés entre l'observateur et la cible. Connaître comment ces milieux non homogènes affectent le passage de la sonde de télédétection (la propagation des ondes électromagnétiques) est de la plus grande importance pour la praticabilité même de la télédétection.

Cependant, l'importance de la caractérisation de la réponse électromagnétique des composites n'est pas limitée à la télédétection. Les composites (céramiques, fibre de carbone, polymères en plastique) jouent un rôle central dans beaucoup de progrès technologiques. Caractériser des composites et des milieux non homogènes est une nécessité de nos jours dans de nom-

breux domaines technologiques, que ce soit dans la production de nouveaux matériaux, pour comprendre le comportement des matériaux existants, ou encore pour la connaissance scientifique pure. En cela le traitement d'image a beaucoup à apporter.

Le comportement macroscopique d'un composite peut s'exprimer en fonction des caractéristiques macroscopiques de ses constituants, mais habituellement d'une manière complexe qui inclut l'information sur la géométrie de l'arrangement des constituants de ce composite. La théorie du milieu efficace (TME) est un outil puissant pour décrire les propriétés macroscopiques des milieux hétérogènes complexes. Dans le cas spécifique de l'électromagnétisme, la TME laisse identifier le champ moyen se propageant à l'intérieur d'un milieu hétérogène avec un champ se propageant à l'intérieur d'un milieu homogène caractérisé par une constante diélectrique efficace. Ainsi, les milieux hétérogènes, à l'équilibre macroscopique et sous l'approximation du milieu efficace (AME), se comportent comme des milieux homogènes avec des caractéristiques mesoscopiques spécifiques, appelées efficaces.

Ces paramètres efficaces peuvent être trouvés par des règles analytiques, dans les systèmes les plus simples, ou par des méthodes empiriques. Si nous pouvons obtenir le tenseur "constante diélectrique efficace", généralement complexe, d'un composé (un minéral, la glace polaire, un type de sol, etc.), nous pouvons modeliser sa réponse au champ électromagnétique, et donc sa réponse comme cible de télédétection. On a proposé dans la littérature plusieurs différentes règles (dites de mélange) pour le calcul de la constante diélectrique efficace d'une matière composite, en fonction de l'arrangement spatial et des caractéristiques diélectriques de ses constituants. Cependant, tous sont conçus pour des géométries spécifiques, et malgré le fait qu'elles peuvent être appliquées à différentes configurations avec succès, aucune règle universelle n'existe.

Une approche commune pour traiter la variété de règles de mélange consiste à trouver les limites supérieures et inférieures en fonction des constantes diélectriques et des volumes partiels des constituants. Ceci permet d'approcher la valeur de la constante diélectrique efficace avec une marge

raisonnable. Cependant, le fait est que les configurations complexes montrant des inclusions arbitrairement formées et qui sortent des limites ne sont pas rares. Les anomalies entre les limites théoriques et les mesures se produisent parce que les limites théoriques sont fondées sur des configurations spécifiques, et ne tiennent pas compte de la configuration détaillée de la microstructure. Les modèles numériques tenant compte de la géométrie spécifique du problème offrent une alternative aux règles de mélange et aux limites théoriques. La nécessité pour inclure la géométrie détaillée du système d'une façon efficace dans la méthode numérique, ainsi qu'une équivalence facilement établie entre les images numériques et les modèles de treillis des composites, suggère le recours aux techniques de bas niveau de traitement d'images numériques pour faciliter l'exécution efficace des méthodes de résolution des modèles numériques.

Les techniques de traitement d'images sont intensivement employées pour l'analyse des échantillons de matières composites, en utilisant des systèmes d'acquisition variés. Cependant, il existe également un autre champ en science des composites qui peut profiter des méthodes et du savoir-faire issu du traitement des images: c'est celui des modèles numériques. En effet, les techniques de traitement d'images peuvent être employées avec succès dans la résolution des méthodes numériques pour l'analyse des systèmes composés, quand ceux-ci sont modélisés avec les grilles discrètes régulières, qui, d'un point de vue informatique, ne diffèrent pas des images numériques.

Les méthodes numériques fonctionnant sur de tels modèles exigent habituellement l'identification des différentes agrégations des cellules d'inclusion, c'est à dire, l'identification de chaque granule ou fibre élémentaire du matériau, et leur caractérisation. Dans ce but, deux travaux fondamentaux se dégagent. Le premier porte sur l'identification et la caractérisation des agrégats des cellules appartenant aux mêmes espèces. Ceci est connu en traitement d'images comme l'étiquetage d'objets ou l'étiquetage des composantes connexes. L'autre traite de la caractérisation de la structure locale de chaque noeud dans le treillis, c'est à dire, un descripteur local de la configuration des voisinages des cellules élémentaires (pixels dans la représentation équivalente d'image).

Le cadre de cette thèse est celui d'un problème général de télédétection ou de caractérisation de matériau composite traité avec une méthode numérique fondée sur une formulation théorique pour le problème électromagnétique d'homogénéisation dans les microstructures. Dans ce contexte, le travail original présenté dans ce mémoire porte sur les deux techniques de traitement d'image de bas niveau mentionnées ci-dessus, à savoir, une nouvelle méthode pour l'étiquetage des composantes connexes, présentant des améliorations significatives par rapport aux méthodes existantes, et une méthode de codage des configurations locales qui offre plusieurs caractéristiques la rendant appropriée pour des applications variées. Ce mémoire introduit donc l'algorithme de composantes connexes dans le chapitre 2 et la méthode de codage de configurations locales dans le chapitre 3. Leurs avantages sont discutés, et des exemples d'application sont fournis au-delà du domaine spécifique étant à leur origine, comme la vision artificielle, la synthèse d'image, ou encore le codage d'image. Quant à lui, le chapitre 1 traite de la pertinence de l'approche "analyse d'image" pour résoudre le problème électromagnétique sous-jacent afin que la complexité de ce problème puisse être appréciée dans toute son ampleur.

Chapter 1

Introduction

'Before I come on board your vessel,' said he,
'will you have the kindness to inform me whither you are bound?'

1.1 A world of composites

Composites are everywhere. Anything which is not made up of a single constituent is a composite, or, in the electromagnetism lingo, a nonhomogeneous medium. The landscape is a composite. The soil is a composite. Polar ice, and also that of glaciers, is a composite. It is made of water molecules and air bubbles, pollutants discounted. The atmosphere is a composite, a *layered* one, and each layer is itself a composite. Rocks are composites, and also modern *composites*, which have opened the doors to many technological advances, due to their never-seen-before properties regarding lightweight, endurance, strength, plasticity, and versatility. Why all these fabulous properties were not seen before? Because composites present features different from that of their constituents. Mix two or more substances that were not mixed ever before, and you have got a new material. That is what makes composites so interesting: Put two pure constituents in the same pot, mix them in the appropriate proportions, and you will get something new, which behaves differently from any of the pure substances. And that is also what makes them so *complex*. The resultant behaviour is not (generally) a linear

combination of those of the constituents. It is not only the volume fraction of each constituent what defines the result, but also the way how each constituent is arranged in the new material. *Structure* comes into scene. What is the size of the inclusions of each constituent, what is their shape, how are they arranged. All this governs the way the composite is going to behave in response to mechanical, chemical, and electrical influences.

Note that the term *composite* has a precise meaning. It is well understood that most substances are not made of pure elements, but of a combination of elements forming molecules or crystals. However, those are *compounds*. By composite (or microinhomogeneous media, or media with microstructures [7, 10, 117, 216]) we refer to a structure assembled from a very large number of particles of specific materials arranged in a specific way, where each particle is assumed to be considerably smaller than the size of the domain, but large enough as to be domains of application of macroscale equations, i.e. larger than atomic or molecular sizes. The arrangement is considered regular in the sense that the microstructure can be considered periodic, quasiperiodic, or statistically homogeneous. Thus, the behaviour of a piece of a composite is representative of the behaviour of any other piece. The minimal representative piece is usually called the unit cell.

What remote sensing is all about is acquiring information on targets by studying their response to electromagnetic stimuli —sonic systems, such as sonar, excluded, unless the definition is extended to include also mechanical stimuli, such as sound waves—, no matter if the stimulus is provided by an external agent, say, the Sun, or by the remote sensing device itself, say, radar or lidar. Remote sensing allows us to obtain exhaustive large scale information about the environment, and also to investigate distant galaxies and the confines of the Universe. And wherever we look, we will find nonhomogeneous media and composites.

To be able to discover anything about a target by remote sensing, either we compare the signal returned from it with that of known targets, in the hope that the unknown target is within our collection of samples, or we unravel the complex interactions between the materials composing the target

and the electromagnetic waves we are using as probe.

Thus, there are two main approaches in the processing of remote sensing data. The one will observe the *spectral signature*, i.e. the response curve of the target along the region of interest of the electromagnetic spectrum, perhaps just a single sample (such as in radar), just a few (multispectral images), or many (hyperspectral images), even up to the degree of obtaining an accurate plot of the whole response curve (imaging spectroscopy), and compare it to others in the same image, or versus a collection in what is usually called a "spectral library", using statistical or pattern recognition tools for the matching. The other will model the way matter interacts with e.m. waves, specially composite materials, depending on the constituents, their proportions, and the way they are arranged, and will try to give responses about the targets according to the observed behaviour. The first approach will be able to give information of the type "this is the same as" or "this is similar to", or "there is as much x as" or "there is less x than"[1], which can be, and in fact it has shown to be, very important. The second approach, provided that we are able to model accurately the response of complex materials for a wide range of constituents and structures, will be able to give qualitative and quantitative information on an absolute basis, such as "the target is made up of elliptical inclusions of constituent c in a host matrix h with a volume fraction between v_l and v_h"[2]. This is, of course, a tougher way than the other, but it also opens many more possibilities. Moreover, it may be the only choice for certain applications, such as ground penetrating radar for deep exploration of the subsoil, or imaging spectroscopy of distant planets and stars.

Sometimes nonhomogeneous media are not the target themselves, but

[1]Note that sometimes we abuse the language when commenting results as those just mentioned and say "these pixels here are pines". According to the usual methodology, all we are formally authorized to state about such results is "those pixels there have a spectral signature more similar to that of those pine trees there than to any other thing in this image".

[2]Yes, the same consideration can be made regarding this second approach. The strict way to put it would be "the target has the same spectral response as a target made up of elliptical inclusions of constituent c in a host matrix h with a volume fraction between v_l and v_h would have, according to our model". Model, which, for sure, will include a series of implicit or explicit assumptions.

they are interposed between the observer and the target, as it is the case of the atmosphere in satellital remote sensing, its lower layers in aerial remote sensing, sea water in the study of the bottom of the oceans, polar ice in the study of Antarctic geology, or sand dunes in the study of the subsoil of deserts. Knowing how these nonhomogeneous media affect the passage of the remote sensing probe (the propagation of e.m. waves) is of utmost importance for the very feasibility of remote sensing.

The importance of the characterization of the e.m. response of composites is not restricted to remote sensing, however. Composites (ceramics, carbon fiber, plastic polymers) play a major role in many technological advances. Many new technologies rely on new materials, which by their specific, extraordinary characteristics regarding their electrical or mechanical behaviour, make possible the exploration of adverse environments (the outer space, the bottom of the sea, the darkest depths of Earth), the miniaturization leading to ultra portability and very high speeds, or "intelligent" materials which adapt themselves to the environment and take advantage of what before used to work against them. We are reaching a point where we can design custom materials *à la carte*, where the client specifies the kind of behaviour he wants from the material (resistivity, colour, thermal expansion, elasticity, ...) and the designer will hand him a recipe, specifying the ingredients, the proportions, and how to mix them to achieve the proper arrangement that will assure the required behaviour. This may not be a reality yet, but we are getting closer, specially regarding certain families of composites and specific properties within certain margins [225].

Certainly, we can produce samples of a composite, measure their electrical and mechanical response, then vary a little bit the recipe, produce new samples, characterize them, and proceed with this trial and error procedure until the desired performance is approximated. However, the response is far from linear, specially in the vicinity of critical phase effects, where the more interesting behaviours are usually observed. If we are able to model the composite, then we can alter the parameters in our model (constituents, volume fractions, structure) and check the behaviour in a much advantageous manner, even if still in a trial and error fashion. The step further is then

understanding the model as to be able to predict the outcomes we will get as a response to changes in the inputs (the recipe of our composite). Then we will be able to go straightforwardly for the target.

Anyway, characterizing composites and nonhomogeneous media is a necessity nowadays in many technological fields, either aiming at producing new materials, understanding the behaviour of preexisting materials, or in the quest for pure scientific knowledge. And in this task image processing has a couple of things to say.

1.2 Effective medium

It was the secrets of heaven and earth that I desired to learn;
and whether it was the outward substance of things or the inner spirit of nature
and the mysterious soul of man that occupied me,
still my inquiries were directed to the metaphysical,
or in its highest sense, the physical secrets of the world.

As stated above, nonhomogeneous media behave differently from any of their constituents. The macroscopic behaviour of a composite can be expressed as a function of the macroscopic characteristics of its constituents, but usually in a complex way which includes information on the geometry of the arrangement of the constituents that form the composite. The Effective Medium Theory (EMT, see [55] for a comprehensive introduction) is a powerful tool to describe the macroscopic properties of complex heterogeneous media. It was first suggested by Maxwell Garnett [87], and later by Bruggeman [39,40], based on Maxwell's equations for the static limit. In the specific case of electromagnetism, EMT permits to identify the *average* field propagating inside an heterogeneous medium with a field propagating inside an homogeneous medium characterized by an *effective* permittivity. Thus, nonhomegeneous media, at macroscopic scale, under the Effective Medium Approximation (EMA) behave as homogeneous media with specific mesoscopic characteristics, called *effective*. These effective parameters can be found either by analytic rules, in the simpler systems, or by empirical methods.

We are interested in the average fields because describing the field at every point in the composite is basically pointless. For most purposes we do not need a detailed description. Replacing the original system by an averaged system is called *homogenization*. We replace the microscopically nonhomogeneous material by an homogeneous material which imitates the relevant behaviour of the original system through the effective properties. In contrast with the randomly varying properties of the composite, the effective properties are constant or, in the quasiperiodic case, smoothly varying functions of the position in the domain. Note that the averaged system will not

preserve all the features in the original system, so we choose which features we want preserved in the homogenization process. The microscale information is lost by homogenization, specifically all processes determined by the individual behaviour of the inclusions: local fields, fine scale oscillations, etc. The reader interested in homogenization is directed to any of many books [7, 12, 13, 56, 59, 109, 153, 186]. A variety of homogenization methods in different fields are described in [9, 15, 31, 125, 144, 152, 192, 193, 211, 242, 247]. For some discussions on the numerical aspects of homogenization see [8, 84, 93, 94, 105, 225, 279], and in [11, 53, 58, 136, 171] some interesting instances of unexpected behaviour of homogenized systems can be found.

Generally, we want to preserve the solution of a boundary value problem. By forcing the solution in the averaged system to approach that of the original system, we get equations leading to the solution for the effective properties of interest. Homogenization is a local process. The nonhomogeneous medium is replaced by a homogeneous medium in a small neighbourhood, the *unit cell*, where the local fields are replaced by their mean values, i.e. averages over the unit cell. It is assumed that the unit cell is representative of any piece of the composite at the same scale. The size of the unit cell has to be compared with the rate of variation of the external fields. Homogenization assumes that the size of the unit cell is much smaller than the other parameters of the system, and that for a given scale the choice of the unit cell does not affect the homogenization results. Then the composite can be substituted by a regular repetition of identical unit cells, without apparent variation in the macroscale behaviour of the variables of interest.

Thus, if we are able to obtain the (generally complex) effective permittivity tensor of any given composite (a mineral, polar ice, a given soil type, ...), we can model its macroscale response to the electromagnetic field, and therefore its response as a remote sensing target. Many natural geophysical materials are heterogeneous media whose electromagnetic behavior can be characterized in an homogenized way in many applications. Soil, for instance, is a mixture of differently shaped granules, water, air, and organic structures, and for dielectric measurements, a thoroughly mixed soil, with a volume large in comparison with its constituents, can be treated as a homo-

geneous body [107].

1.2.1 The effective permittivity

Under the effective medium approximation, there exists a unit cell that reflects all the relevant macroscopic characteristics, such that the material could be replaced by a regular arrangement of repetitions of the unit cell without any change in its macroscopic (effective) response to the electromagnetic field. This assumption is facilitated by the introduction of the long wave approximation (LWA), valid whenever the features being probed are much smaller than the wavelength, neglecting the scattering effects just by assuming that the wavelength is large compared to the characteristic size of the inclusions and their separation. The LWA determines the range of applications depending on the material, and vice versa. However, it allows the treatment of common problems in geoscience, where the need for large penetration depths usually dictates low frequencies of operation. For instance, effective permittivity is a critical parameter in ground-penetrating radar (GPR) (it controls the speed of the electromagnetic wave, the reflection coefficient at the interfaces, and the imaging resolution) and has a consistent behavior over the typical GPR frequency range [165].

Several different rules for the computation of the effective permittivity $\epsilon_{\text{eff}}(\omega)$ of a composite material have been proposed in the literature, as a function of its structure and the spatial arrangement and dielectric characteristics of its constituents [185, 227]. The simple mixing model for inclusions with permittivity function $\epsilon_{\text{i}}(\omega)$ which occupy a fractional volume p in a host matrix with permittivity function $\epsilon_{\text{m}}(\omega)$:

$$\epsilon_{\text{eff}}(\omega) = p\ \epsilon_{\text{i}}(\omega) + (1-p)\ \epsilon_{\text{m}}(\omega) \tag{1.1}$$

is generalized as the power–law approximation [19, 23, 147]:

$$\epsilon_{\text{eff}}^{1/n}(\omega) = p\ \epsilon_{\text{i}}^{1/n}(\omega) + (1-p)\ \epsilon_{\text{m}}^{1/n}(\omega) \tag{1.2}$$

where different values of n give the Landau ($n = 2$) [141] and the Looyenga ($n = 3$) [146] approximations. The classical model for spherical inclusions in

an isotropic matrix is:

$$\frac{\epsilon_{\text{eff}}(\omega) - \epsilon_{\text{m}}(\omega)}{\epsilon_{\text{eff}}(\omega) + 2\epsilon_{\text{m}}(\omega) + \upsilon(\epsilon_{\text{eff}}(\omega) - \epsilon_{\text{m}}(\omega))} = \\ = p\,\frac{\epsilon_{\text{i}}(\omega) - \epsilon_{\text{m}}(\omega)}{\epsilon_{\text{i}}(\omega) + 2\epsilon_{\text{m}}(\omega) + \upsilon(\epsilon_{\text{eff}}(\omega) - \epsilon_{\text{m}}(\omega))} \tag{1.3}$$

where different values of υ produce the well known Maxwell Garnett rule ($\upsilon = 0$) [87], the Bruggeman rule ($\upsilon = 2$) [39] or Polder-van Santen mixing formula [199, 252], and the coherent potential approximation ($\upsilon = 3$) [129]. Similar expressions are those of Böttcher [29, 30], Tsang and Kong [250], Bohren and Huffman [28], and Monecke [178]. Iglesias *et al.* [113] presented a mixing rule from which those of Bruggeman, Böttcher, Tsang and Kong, and Landau can be derived.

All of these are designed for specific geometries, and in spite of the fact that they may be successfully applied to different configurations, there is no universal rule. Bergman's theorem [14] relates the effective dielectric function of a two-phase composite to its structural microgeometry through a spectral function, $G(L)$:

$$\epsilon_{\text{eff}}(\omega) = \epsilon_{\text{m}}(\omega)\left(1 - p\int_0^1 \frac{G(L)(\epsilon_{\text{m}}(\omega) - \epsilon_{\text{i}}(\omega))}{(1-L)\epsilon_{\text{m}}(\omega) + L\epsilon_{\text{i}}(\omega)}dL\right). \tag{1.4}$$

$G(L)$ is generally unknown for arbitrary systems, but it can be either analytically or numerically derived for any existing mixing rule [234]. In fact, its derivation permits the validation of mixing rules, provided that $G(L)$ has to be non-negative, normalized in $[0, 1]$, and that its first moment should equal $(1 - p)/3$. Böttcher's spectral function, for instance, only complies with the normalization restriction. However, finding the spectral function for specific nontrivial configurations is far from trivial.

A common approach for dealing with the variety of mixing rules in the literature is trying to find upper and lower bounds for $\epsilon_{\text{eff}}(\omega)$ as a function of the permittivities and fractional volumes of the constituents [228]. This allows approximating the value of the effective permittivity within a reasonable margin, and makes rigorous bounds an important tool for practical applications.

The absolute bounds are dictated by the fact that no mixture may have effective permittivity greater or lesser than any of its constituents. Tighter limits are those of Wiener [268], result of making $n = \pm 1$ in (1.2), which correspond to serial (minimum permittivity) and parallel (maximum) configurations of parallel plates. Hashin and Shtrikman formulated stricter bounds [103]: The lower bound is (1.3) with $\upsilon = 0$ (Maxwell Garnett formula) whereas the upper bound is the inverse, with inclusions treated as host and vice versa. Stronger, third order bounds have been suggested by Milton [170], Felderhof [75], or Helsing [106], which take into account the shape of the inclusions through the so-called Miller parameter.

No matter how tight the bounds, the fact is that, even when typical configurations within the midrange of the fractional volume obey particularly well the bounds, complex configurations with arbitrarily shaped inclusions that go out of bounds are not rare [228]. Losses in the material, which could even bring to the violation of the absolute limits, scattering, and magnetoelectric effects should be blamed. However, we work under the assumptions of negligible scattering (LWA) and low-loss, nonmagnetic materials without free charges. Dielectric loss is negligible if the conductivity of a material is less than 10 mS/m, as it is for many geologic materials. The low-loss and nonmagnetic assumptions are valid for most sedimentary materials containing freshwater [165]. Under these assumptions, discrepancies between the theoretical bounds and measurements occur because the theoretical bounds are based on specific configurations, and do not take into account the detailed configuration of the microstructure of any given material, which influences its effective response. The percolation threshold [233], for instance, which depends on the shape and spatial configuration of the inclusions, strongly affects the effective permittivity [226].

The alternative to mixing rules and theoretical bounds is numerical modelling taking into account the specific geometry of the problem [16–18, 38, 52, 89, 114, 122, 123, 168, 195, 219, 220, 231, 248, 249, 253, 267]. The author developed with Prof. J. F. Peón-Fernández a numerical method to estimate $\epsilon_{\text{eff}}(\omega)$ through the computation of the potential at the boundaries of inclusions embedded in a host matrix [161]. The interested reader can find in Appendix

A the details of the theoretical formulation supporting the method, which is not a requisite but can help to understand some of what follows.

The necessity to include the detailed geometry of the system in an efficient manner in the numerical method, together with an easily established computational equivalence between digital images and lattice models of composites, recommended the recourse to low level digital image processing techniques to aid in the efficient implementation of the method.

1.2.2 The numerical method

In the following I will describe with some detail the implementation guidelines of the numerical method such that the relevance of the techniques described in this report and the complexity of the problem at hand can be appreciated in all their extent.

Once the unit cell —the minimum representative volume— is established, we subject it to a potential U_0, i.e. two opposite boundaries are forced to potential 0 and U_0 respectively, without restriction on the free boundaries, and we want to obtain the potential at the boundaries of every inclusion in the unit cell. This is a Dirichlet problem that can be solved using a wealth of well known numerical methods [67], including a probabilistic solution [182] using Monte Carlo methods [98, 212], see Appendix B.

We generate a cubic mesh that partitions the unit cell into a lattice of elementary homogeneous cells, and we want to obtain the potential at every node in the mesh on the boundary of inclusions.

The potential at each node in the mesh, U, can be written as a function of the potentials at the neighbour nodes, $\nu \in \{\mathrm{x}^+, \mathrm{x}^-, \mathrm{y}^+, \mathrm{y}^-, \mathrm{z}^+, \mathrm{z}^-\}$, subject to the conservation of the normal component of $\vec{D}$ and of the tangential component of $\vec{E}$, using a Taylor series expansion where the derivatives are approximated by finite differences, i.e. the Newton-Cotes formula for Laplace's equation,

$$U = \sum_{\nu} s_\nu U_\nu, \ \sum_{\nu} s_\nu = 1 \tag{1.5}$$

where the interpolation coefficients s_ν are determined by the boundary conditions [54]. Edges connecting two nodes are interfaces between homogeneous elementary cells. The finite difference equivalent of the boundary condition of conservation of the normal component of $\vec{D}$ is obtained by applying $\iint \vec{D} \cdot d\vec{S} = 0$ to the interface [80]. Thus, for instance, in a 2D mesh, if there is a horizontal interface (along $\hat{x}$) between local permittivities ϵ_1 (above) and ϵ_2 (below), then $U = s_{\mathrm{x}^+}U_{\mathrm{x}^+} + s_{\mathrm{x}^-}U_{\mathrm{x}^-} + s_{\mathrm{y}^+}U_{\mathrm{y}^+} + s_{\mathrm{y}^-}U_{\mathrm{y}^-}$, where

$$s_{\mathrm{x}^+} = s_{\mathrm{x}^-} = \frac{1}{4},\ s_{\mathrm{y}^+} = \frac{\epsilon_1}{2(\epsilon_1+\epsilon_2)},\ s_{\mathrm{y}^-} = \frac{\epsilon_2}{2(\epsilon_1+\epsilon_2)}. \tag{1.6}$$

The rest of configurations is obtained in the same way.

To compute the potentials at the mesh nodes a classical relaxation scheme [116] can be used, such as Gauss–Seidel [121] or successive overrelaxation (SOR) [72], by repeatedly updating the potentials according to (1.5) until convergence. These methods provide the field solution; they find the potential for every node in the mesh. However, we need it only for a subset, the nodes on the boundaries of inclusions, i.e., the boundaries of foreground pixels in a equivalent image representation. Connected components labelling allows us to know to which inclusion belongs each node and which are the nodes we are interested in.

The alternative when only a few local solutions are needed is Monte Carlo random walk. In a Monte Carlo random walk solver, the coefficients s_ν in (1.5) are probabilities that guide the displacement of random walkers through the mesh. A walker in a node like that of (1.6) would move to the right with probability s_{x^+} and to the left with probability s_{x^-}.

Monte Carlo random walk works as follows: N walkers leave from a given node i whose potential U_i is to be obtained. A walker finishes its trip upon arrival to a node k on any of the boundaries of the unit cell with fixed potential, $U_k \in \{0, U_0\}$, which act as sinks. The potential at node i is estimated as

$$U_i = \frac{1}{N}\sum_k n_k U_k \tag{1.7}$$

where n_k is the number of walkers that reached the fixed potential node k.

See Appendix B for a formal treatment of the Monte Carlo solution of elliptic PDE's.

Whatever the method to solve the potential, the procedure to compute $\overleftrightarrow{\epsilon}_{\text{eff}}$ is as follows:

1. Force a unitary macroscopic field $\vec{E}_0$ in one of the three orthonormal directions of the system, say $\vec{E}_0 = (1, 0, 0)$, by forcing the two corresponding normal faces of the unit cell to potential 0 and $U_0 = 1$ respectively (all lengths are normalized with respect to the length of the unit cell).

2. Use a relaxation method or Monte Carlo random walk to approximate the potential at the boundaries of the inclusions.

3. Compute the first three components of the permittivity tensor using (A.18) with $\hat{e} = (1, 0, 0)$

$$\begin{pmatrix} \epsilon_{\text{xx}} \\ \epsilon_{\text{yx}} \\ \epsilon_{\text{zx}} \end{pmatrix} = \begin{pmatrix} \epsilon_{\text{m}} \\ 0 \\ 0 \end{pmatrix} + \frac{\epsilon_{\text{i}} - \epsilon_{\text{m}}}{V_{\text{C}}} \sum_{n}^{N} \oiint_{S_n} U(\vec{r})\, d\vec{S}. \tag{1.8}$$

4. Repeat the procedure with $\vec{E}_0 = (0, 1, 0)$ and $\vec{E}_0 = (0, 0, 1)$, by successively applying U_0 to each of the remaining pairs of opposite faces of the unit cell.

Monte Carlo vs. Relaxation

Relaxation methods provide the field solution of the potential inside the unit cell, i.e. in a single run we get the potential for every node in the mesh. With Monte Carlo, we need a run for each node of interest. The iterations in relaxation methods and Exodus are fairly similar (see Appendix C for a detailed treatment of Exodus Monte Carlo). Relaxation methods may show slow convergence, depending on the spatial arrangement of the unit cell. Successive overrelaxation [72] provides a method for accelerating convergence with respect to Gauss–Seidel, but estimating the optimum moment parameter ω

can be difficult, specially in 3D, and the wrong choice may worsen things. More sophisticated methods exist, such as multigrid [237], but at the cost of increased complexity. In serial machines, Exodus can also be accelerated by incorporating the solved nodes to the list of sinks (known potential nodes). Thus, the more nodes are solved, the more sinks for walkers, the faster the remaining nodes are solved.

Exodus is a clear choice for solving the potential of just a few single nodes. When the number of target nodes is big, a field solution through relaxation is clearly advantageous in serial machines. In between, the specific geometry of a given problem will determine the opportunity of one or the other method, through its effect on the behaviour of the random walkers and the rate of convergence of the relaxation methods.

In spite of the above considerations, the local character of the Monte Carlo random walk solution allows for greater parallelism, given the fact that the local solutions are independent, and thus achievable through independent processes that can be executed in parallel. This holds either with traditional Monte Carlo or Exodus. Exodus has the additional advantage of not requiring random walks in the strict sense, provided that random walks are difficult —but not impossible, see [275]— to parallelize. Moreover, the potentials at boundary nodes are required only to perform the integral in (1.8). This can be computed node per node, and thus distributed together with the potential estimation in a purely parallel implementation.

1.3 The role of image processing

There is an obvious role for image processing in the world of composites, and materials science in general. Image processing techniques are extensively used for the analysis of samples of composite materials, using imaging techniques such as light microscopy, from optical reflection microscopy and confocal laser scanning microscopy to novel techniques such raman microscopy, tomography, and microtomography, as well as non-optical microscopy techniques including electron microscopy, nuclear magnetic resonance, ultrasonics, and scanning acoustic microscopy [60]. Image processing allows the detection of imperfections, the characterization of average grain sizes and shapes, fiber lengths and diameters, interstitial spaces, holes, bubbles, aggregation levels, homogeneity, presence or absence of coatings, pollutants, constituents, etc. either providing processing techniques which enhance the images previous to their analysis by human operators, or well up to the last link in the chain, through machine vision systems that can analyze samples with a high degree of reliability, accuracy, repeatability, and objectivity.

However, there is also another field in composites science which can profit of image processing methods and know-how. Such a field is numerical modelling. Image processing techniques can be successfully used in the implementation of numerical methods for the analysis of composite systems when these are modeled with regular discrete grids, which, from a computational point of view, do not differ from digital images. Composites are often modelled as 2D or 3D host matrices with inclusions of one or several species, and regular lattices are used to discretize matrix and inclusions. Thus modelled, if lattice cells covering host material are interpreted as background pixels, and lattice cells covering inclusion material of different species are interpreted as pixels of different colors, the model is computationally indistinguishable from a digital binary —if only one species is present— or colour —if several species are present— 2D or 3D image. As such, there is a wealth of image processing techniques that can be used to aid in the analysis and simulation of composites.

Numerical methods running on such models usually require the identifi-

cation of the different aggregations of inclusion cells, i.e. the identification of each different granule or fiber in the host material, and their characterization, regarding their size, shape, and spatial arrangement, for instance with regard to possible clustering or regularities. For this aim there are two fundamental tasks: One is the identification and characterization of aggregates of cells belonging to the same species. This is known in image processing as object labelling or connected component labelling [222], and in statistical physics as cluster identification or cluster labelling [91]. Another common task is the characterization of the local structure for every node in the lattice, i.e. a local descriptor of the configuration of elementary cell (pixel in the equivalent image representation) neighbourhoods.

The homogenization method just described is a good example of a numerical method that can be effectively boosted by the adequate use of these techniques. As it is justified in Appendix A, the homogenization process devised to obtain the effective permittivity of a composite made up of inclusions of several species in a host matrix, all with known permittivities, requires the computation of the potential at the boundaries of the inclusions inside the unit cell, after subjecting it to a known external field, and the subsequent integrations such as that in (1.8). For that purpose, the inclusions of the different species have to be identified, and the local configuration at each location where the potential is obtained has to be taken into account for the potential estimation and for the numerical integration.

We generate a cubic mesh that partitions the unit cell into a lattice of elementary homogeneous cells, and we want to obtain the potential at every vertex (node) in the mesh on the boundary of inclusions. Given the fact that the mesh is a regular 2D or 3D grid, and the elementary cells are homogeneous, the rendered lattice is computationally equivalent to a 2D or 3D image, the equivalent image representation (EIR) of the model, with the elementary cells corresponding to pixels or voxels. In the following, in general, I will refer to both pixels and voxels as pixels, for the comfort of both the reader and the writer, but it is well understood that in 3D models the elementary cells correspond to voxels in the EIR. Mesh edges are the pixel boundaries, and the vertices or nodes the corners of the image pixels. Pixel

Model	EIR
Unit cell	Image
Boundaries	Image edges
Elementary cells	Pixels (or Voxels)
Inclusion cells	Foreground pixels
Matrix cells	Background pixels
Nodes (Vertices)	Pixel corners
Edges	Pixel boundaries
Facets	Voxel boundaries
Species	Colour
Inclusion	Label

Table 1.1. *Correspondences between the model and the equivalent image representation (EIR).*

colours or graylevels are dictated by the species to which the corresponding elementary cell belongs. Pixels with the same colour correspond to inclusions of the same species. After connected component labelling on the EIR, labels identify the individual inclusions, and colours the different species. Table 1.1 summarizes the equivalences between the composite model and the EIR.

Thus, for instance, (1.6) can be reinterpreted in EIR terms as stating that the interpolation coefficients for each pixel —or voxel— corner depend on the colour of the neighbouring pixels —or voxels— in the equivalent image representation. The reader should keep in mind that the EIR is not an image of a sample of the composite, but a convenient interpretation of the model for the purposes of applying digital image processing methods to assist the numerical methods running on the model.

A connected components labelling of the IER provides the numerical methods with information which allows including in the computations only the relevant data. For instance, during integration of the potential, identifying the inclusion to which each boundary node belongs, such that the potential can be integrated on a per inclusion (grain, fiber, ...) basis.

The other relevant image processing technique is the encoding of the local structure around each node, i.e. the encoding of the configuration of local pixel neighbourhoods in the EIR. No matter what method is used to solve

the potential, the set of s_ν coefficients in (1.5), six in 3D systems and four in 2D, has to be determined for each node in the mesh. The determination of these coefficients requires the inspection of all eight or four neighbour elementary cells of each node to determine their permittivity, as in (1.6). This can be done once per iteration, which is a considerable overhead, or only once at the beginning if a six or four dimensional array is used to store the coefficients of each of the L^3 (L^2) nodes in the mesh, what can take a considerable amount of memory. The solution is to encode the local configuration of the elementary cells surrounding each node according to an efficient scheme, such that later, during the iterative computations, the code associated to each node will determine the coefficients from a predefined finite set of possibilities.

The same applies for solving the integrals in the right hand side of (1.8). To evaluate the contribution of a node to each of the components of the corresponding integral, we have to determine to how many free facets of inclusion cells, normal to each direction, belongs the node. In EIR terms, we have to determine to how many foreground pixels or voxels in the boundary of a connected component belongs a given pixel or voxel corner. This again depends on the local configuration.

1.4 Original contributions

Within the framework just described, where a general problem in —but not restricted to— remote sensing is dealt with a numerical method based on a theoretical formulation for the electromagnetic homogenization problem in microstructures, implemented with the aid of original image processing methods, there are a number of significant contributions, namely:

- An original theoretical formulation for the homogenization problem of the effective permittivity.
- A numerical method with an efficient implementation of the formulation above.
- A new connected component labelling with various advantages with respect to the classical methods.
- A local configuration encoding scheme with a number of characteristics that make it suitable for different applications, including image coding and pattern recognition.
- A 3D chain code for arbitrary 3D shapes.
- Some numerical results for regular composites demonstrating the feasibility of the methods above from which some interesting results can be concluded (see Appendix E).

These have been published in a number of refereed journals and conferences:

- J. Martín-Herrero, 2006. "Hybrid object labelling in digital images", *Machine Vision and Applications Journal*, (in press).
- J. Martín-Herrero and J. F. Peón-Fernández, 2005. "Computation of longwave electromagnetic response of nonhomogeneous media", *IEEE Transactions on Geosciences and Remote Sensing*, 43(7):1479–1489.

- J. Martín-Herrero, J. F. Peón-Fernández, T. P. Iglesias, 2005. "The effective permittivity of lossless granular composites: From periodic to random distributions", *Proceedings of the XXVIII General Assembly of the International Union of Radio Science (URSI-GA2005).*

- J. Martín-Herrero, 2005. "Two low level image processing techniques for the characterization of composite systems", *Proceedings of the* 5^{th} *International Conference on Composite Science and Technology.*

- J. Martín-Herrero, 2004. "Hybrid cluster identification", *Journal of Physics A*, 37:9377–9386.

- J. F. Peón-Fernández, J. Martín-Herrero, N. Banerji, T. P. Iglesias, 2004. "Numerical study of effective permittivity in composite systems", *Applied Surface Science*, 226:78–82.

- J. Martín-Herrero, J. F. Peón-Fernández, N. Banerji, T. P. Iglesias, 2002. "Monte Carlo estimation of the effective electric response in composites", in S. Bandyopadhyay (ed.) *Composite Systems: Macrocomposites, Microcomposites, Nanocomposites.* 300–306.

while others have been submitted and are awaiting revision.

This report will be focused on the connected component algorithm and the local configuration encoding scheme, detailing them and their advantages, and providing examples of application in fields out of the scope of the specific environment which originated them, namely machine vision, image synthesis, and image coding.

Chapter 2 deals with the connected component labelling algorithm, and Chapter 3 deals with the local configuration encoding scheme. The formal treatment of some ancillary aspects is presented for reference in the Appendixes, and also some results of the homogenization method which benefits of the techniques detailed in the text.

Chapter 2

Hybrid Connected Component Labelling

. . . for nothing contributes so much to tranquillize the mind
as a steady purpose —a point on which the soul may fix its intellectual eye.

2.1 Introduction

Labelling the connected components of a digital image is a fundamental low level technique in image processing and machine vision, as an intermediate step allowing further processing at object level. A connected component is a set of connected pixels in a binary image such that a connected path made of neighbouring foreground pixels exists between every two pixels in the set. The notion can be easily extended to non binary images by repeating the process for each digital level, colour, or group of colours. A connected components labelling of a binary image is a labelled image in which the value of each foreground pixel is the label of its connected component. Connected component labelling in digital images is just a specific application of the more generally known as cluster labelling or cluster identification algorithms, of primary use in statistical physics[1]. Several algorithms for serial machines

[1]Cluster identification is a low level technique fundamental in spatial analysis. Its purpose is assigning the same label to every connected site having the same state in a

have been described for this task, which can be broadly classified between recursive and iterative techniques, and between single pass and two pass (or more). Parallel implementations are not discussed here; the interested reader is directed to [5].

2.1.1 Connectivity in digital images

A continuous image is an intensity function $f(\mathbf{x})$, $0 < f(\mathbf{x}) < \infty$, that gives the intensity of (usually, but not necessarily) light at the points with spatial coordinates $\mathbf{x}$. A digital image F is obtained by regular spatial sampling and intensity level quantization of f. A digital image is then a tessellation of the space into regular p-gons, polygons with p edges, where the amplitude of f remains constant. These p-gons are denoted pixels —picture elements— in 2D, when $\mathbf{x} \in \mathbb{R}^2$, and voxels —volume elements— in 3D, when $\mathbf{x} \in \mathbb{R}^3$.

Connectivity in F is defined in terms of adjacency relations among p-gons. A *neighbourhood rule* will determine which p-gons are connected. Two p-gons are neighbours if they share an edge, a vertex, or both. From now on, without loss of generality, we will restrict the discussion to square pixels. The extension to cubic voxels is trivial, and the generalization to p-gons follows straightforwardly. Square pixels are generally assumed to be centered at consecutive integer lattice points of the plane, and the pixel with coordinates (i, j) in the plane is denoted as $p_{i,j}$. Dealing with square pixels, a pixel will have four neighbours if we consider adjacency in terms of edges, or eight neighbours if we consider edges and vertices.

Two pixels $p_{i,j}$ and $p_{u,v}$ are 4-neighbours if $|i - u| + |j - v| = 1$ and 8-neighbours if $0 < \max\{|i - u|, |j - v|\} \leq 1$. Two pixels obeying a given common condition $\mathcal{A}$ are connected by a 4(8)-path if there exists a sequence of pixels p_{i_k,j_k}, $1 \leq k \leq n$, obeying $\mathcal{A}$, such that $p_{i_{k-1},j_{k-1}}$ is a 4(8)-neighbour of p_{i_k,j_k}. The usual condition $\mathcal{A}$ is often having a given intensity level or colour, for instance, black, but many others may exist depending on the

lattice. Connectivity is defined according to a neighbourhood rule, which depends on lattice dimension and geometry, usually a 4-nearest neighbour or 8-nearest neighbour rule in square two dimensional lattices.

application, e.g. texture similarity in a neighbourhood window in textural region segmentation.

Let $C_{\mathcal{A}}$ be the connectivity relation defined on F as follows: For all pairs of pixels $p, q \in F$, $(p, q) \in C_{\mathcal{A}} \iff p$ and q obey $\mathcal{A}$ and are connected by a path in F. Then $C_{\mathcal{A}}$ is an equivalence relation that partitions the pixels of F obeying $\mathcal{A}$ into distinct equivalence classes. Each equivalence class is called a connected component, object, figure, or cluster, depending on the context. I will use interchangeably the first two terms, sometimes just abbreviated as components.

The labelling problem is concerned with assigning a unique label to each connected component in the image. In a labelling of F two pixels obeying $\mathcal{A}$ have the same label if and only if they are in the same connected component. Some definitions follow: The internal distance between two pixels obeying $\mathcal{A}$ is the length of the shortest 4- or 8-path connecting them. The internal diameter of a connected component is the maximum of all internal distances among all pairs of pixels in the component. Let $P_{\mathcal{A}}$ be the set of pixels in F obeying $\mathcal{A}$. A pixel in $P_{\mathcal{A}}$ belonging to component c under 4(8)-neighbourhood is a boundary pixel of component c if one of its 8(4)-neighbours is in $\overline{P}_{\mathcal{A}}$. The centroid of a connected component is the average location of all the pixels in the component. The bounding box is the minimum rectangular box enclosing the component. A *spanning* or *percolating* component is a component with pixels in two opposite boundaries of F. Alternatively, two sides of the bounding box of a spanning component are in opposite boundaries of F. Percolation Theory is the field of science dealing with the phenomena associated with percolating clusters in general lattices.

For the purposes of labelling, all the information needed about the pixels is if they obey or not obey $\mathcal{A}$. Thus, no matter if we are dealing with a binary image, a grayscale image, a colour image, or a multi or hyperspectral image, and no matter how simple or complex $\mathcal{A}$ is, the labelling only requires a binary mask of the image, where usually pixels in $P_{\mathcal{A}}$ are referred as *foreground* pixels and pixels in $\overline{P}_{\mathcal{A}}$ as *background* pixels.

2.1.2 The classical recursive approach

One of the common techniques for object labelling is the classical recursive connected component labelling algorithm [149,239], which can be traced back to the 70's [3, 35, 142, 240], or even earlier. The operation of the recursive technique is as follows: The labelling is initialized by assigning a negative integer to all foreground pixels, and 0 to the background pixels. A recursive function assigns the current label, an ordered positive integer, to the current pixel and scans all of its neighbours in search of another negative label. If any is found, the function calls itself to repeat the procedure there. A single pass over the image, in which the function is invoked for every foreground pixel, labels every object in the image with consecutive labels, such that at the end of the pass the total number of objects in the image is known. The extension to non binary masks is straightforward, by using as many different negative labels as identity criteria $\mathcal{A}_n$, for instance different digital levels, colours, or group of colours in the image. Explicit algorithms and detailed treatment of the pure recursive technique for cluster labelling and percolation detection in normal and huge images can be found in [155].

The advantages of the recursive technique are that it performs component labelling and, optionally, straightforward component characterization and percolation detection during the labelling in a single pass over the image, and its comparative simplicity, because it does not need handling any additional data structure as it is the case with two-pass iterative algorithms. The recursive technique allows object characterization (e.g. area, centroid, bounding box) and detection of spanning objects in parallel with the labelling in a single pass over the image. It also allows the labelling and characterization of a given component without having to label all the image. The ability to characterize objects during the labelling, such that a given object can be identified and processed without being forced to label all objects in the image, can be a great advantage in time-critical applications, such as many real time machine vision applications.

The major factor limiting the applicability of the recursive technique is usually considered to be the entire image having to fit in random access

memory. This could be a concern some years ago, but not anymore in most applications, due to the exponential development of the hardware. The real limiting factor nowadays is the stack. The major drawback of the recursive technique is the intensive use of the stack, which threatens processes with being interrupted by a stack overflow. This is due to the high number of consecutive recursive calls (*recursion depth*) issued by the algorithm when labelling big components. The recursive technique requires an oversized stack to store local and state variables for a large number of consecutive recursive function calls when dealing with images containing big objects. The stack is usually limited by the system, and the risk that a stack overflow may interrupt lengthy unsupervised or critical real time processes appears. One solution to this problem is avoiding recursion by converting the algorithm to iterative, which requires creating and managing a custom stack in the heap, a dedicated data structure that may require more room than the image itself.

2.1.3 The classical iterative approach

The classical alternatives to recursive labelling are two pass iterative techniques [70,99,110,150,208,209] which require the entire labelling of the image with temporary labels, followed by a second pass during which some of the labels are corrected. Thus characterization of objects during the labelling becomes fairly complicated [111, 112] when compared with the recursive approach, and the labelling of just a single object is not possible. In exchange, they allow the labelling of huge images with small memory usage, as they usually need to have in memory just two lines (in 2D) and an equivalence table.

In Rosenfeld and Pfaltz's algorithm [208], during a first pass each foreground pixel is labelled with the labels of the pixels immediately preceding it in row-major order. When a pixel is attributed more than one label, the minimum of them is selected, and the location is marked as a "merge" point. Thus, after the first pass, different pixels in the same component may have different labels, and merge points are possibly scattered all over the image. Thus a second pass is necessary where pixels are relabelled using the merge

points until all pixels in the same component have a unique label. The merge points are processed such that every time a merge point is found, an equivalence pair is obtained by collecting the labels of the neighbour pixels preceding the merge point in row-major order. This equivalence pair is used to relabel all the pixels in the image, and the procedure is repeated for all merge points. This is obviously less than optimal, and Rosenfeld and Pfaltz already discussed using global equivalence tables. Hoshen and Kopelman [110] developed the idea to produce one of the more used labelling algorithms, known as Hoshen-Kopelman algorithm (HK hereafter), a two pass algorithm that first propagates cluster labels sequentially to the forward neighbours while constructing a global equivalence table for conflicting labels, and then translates all labels to their definitive values according to the equivalence table on a second pass.

Lumia *et al.* [150] suggested using a small local equivalence table instead of a large global equivalence table. The equivalence table contains all equivalence pairs just for the current scan line, and the labels are propagated from one line to the next during the scan. All equivalence pairs in the current line are collected in the equivalence table using Union-Find operations, taking advantage of Tarjan's almost linear Union-Find algorithm [241] to handle the equivalence table. Union-Find algorithms allow efficient construction and manipulation of equivalence classes using data trees, as first suggested by Galler and Fischer [90]. Then the line is re-scanned to relabel all pixels according to the equivalence table. This method is suitable for implementation on computers with very small memory, but, as stated above, nowadays memory is not the major concern it used to be.

Using Tarjan's Union-Find operations greatly improves the handling of equivalence tables, and therefore nowadays the best iterative performance is achieved with global Union-Find equivalence tables. Union is used to add new equivalence pairs every time a new label conflict (merge point in Rosenfeld and Pfalzt's notation) is encountered, and Find is used in the second pass to relabel the entire image [70]. However, HK remains one of the most popular, if not the most, labelling algorithms, specially within physicists. The Swendsen-Wang cluster update algorithm [238] uses HK to update clus-

ter states in the generation of samples from the Ising model [115] and the generalization for more than 2 states, the Potts model [200]. The Ising and Potts models are of interest for image processing because of their ability to produce image samples with interesting statistical properties. See [272], [88], or [95] for a review.

Less popular approaches to connected component labelling use boundary tracing to surround the component and then fill in the interior [1, 47, 92], and also quadtree and bintree representations based on successive subdivision of the image into quadrants [214, 215].

2.1.4 Recent developments

In spite of its fundamental role as a basic low level image processing technique, connected component labelling has not received much attention from the image processing community, at least when compared to the statistical mechanics community. This is probably due to the fact that in computational physics the challenge is way bigger, due to typical size —while images in the order of Gigapixels are not yet expected in the short term, statistical mechanicians are usually interested in simulating phenomena in near infinite media—, lattice structure —images are more than usually regular grids, crystal lattices show very different arrangements—, and dimension —images are not expected to go over three dimensions in the near future, physicists' lattices in 5D are not rare. Also because the image processing community seems to be satisfied with the performance of the iterative algorithms, once the recursive algorithm is almost automatically rejected from consideration due to memory limitations, except for very specific applications. However, with the advent of real time computer vision, all critical low level techniques have to be fine tuned to the limit of their possibilities, in order to achieve the processing abilities required from modern applications in negligible time, and this automatic selection of the method of choice must be revised.

Percolation theory is probably the field that has been more prolific in connected components identification and related algorithms. Percolation deals

with connectivity in lattices, of which regular square grids —images— are a particular case. A lattice is said to percolate when a connected component spans the entire lattice between two opposite borders. This is of interest in many fields as disparate as soil permeability, mining, fire spread, conductivity, or magnetism, see Stauffer [233] for a readable introduction. Detecting spanning objects is a common task in many image processing applications related to spatial analysis, such as geographical information systems (GIS) and landscape analysis [270], metapopulation ecology [176], or risk fire assessment [124], to mention just a few. Within machine vision, it usually appears as the need to check the continuity of parts under inspection, such as wirings, pipes, or beams.

There have been recent advances in simulation techniques for the study of percolation at different occupation densities and lattice geometries, with considerable gains in performance at the cost of increased specificity. Works like those of Paul, Ziff, and Stanley [194] or Babalievski [6] focus on high dimensional lattices. The former use a pure recursive technique to grow the lattices on the spot; they do not label pre-existent components. Babalievski compares spanning-tree approaches with HK, and presents a depth-first method to identify loops of occupied sites. Sheppard et al. [224] report an efficient algorithm for simulating invasion percolation in big lattices. Rappaport [202], Moukarzel [180], Tiggemann [244], or Moloney and Pruessner [177] have also devised methods to deal with huge lattices, some of them implementing HK on parallel machines. Parallel implementations of HK can also be found in Flanigan and Tamayo [77], Constantin et al [65], or Teuler and Gimel [243]. Al-Futaisi and Patzek [4] present an extended version of HK to deal with non-lattice environments. Newman and Ziff [187] provide a method (including code) to study percolation for all possible occupation densities, p, at a time in random uniform images built during the process, which outperforms the classical methods by several orders of magnitude[2]. However, this performance comes from the specific purpose of the method, as the authors recognize when they state that typical depth-first or breadth-first methods

[2]As a marginal note, I find their average 4.5 seconds for each possible value of p on a 1000^2 lattice using a typical depth-first search on a 2001 computer considerably high.

are optimal for a single value of p.

In spite of these many developments, algorithmic efficiency has many readings. Algorithms may exist which may be faster for a given task, but using quite complex data structures, rather obtrusive for the non specialized computer programmer. Or the task may be so specifically constrained that the algorithms can not be used for general purposes. Thus everything depends on what is expected from a given technique. The fact is that HK is used by a great number of researchers, to the point of being "still the standard technique for identifying clusters in percolation" [177], in spite of the aforementioned works. This reflects that not everybody has access to parallel supercomputers, the convenience of simple, understandable, customizable code, and that many applications require the labelling of pre-existent lattices, such that "growing" the lattice on the spot is unsuitable.

Therefore, typical general purpose connected component labelling algorithms remain necessary tools, and the simplicity and elasticity of recursive labelling would made it a good alternative to the iterative techniques if its stack limitations were overcome. The pure recursive technique needs a stack of about 5 Mb to prevent overflow when processing a 512×512 image. That is too big a stack for not such a big image. The aim is maintaining the simplicity of the recursive approach, without any other data structure than the image itself, while dealing with this major drawback, stack abuse. Also preserving the single pass, one component at a time, character of the recursive technique, which makes it so flexible and suitable for many purposes.

The answer is hybrid object labelling [159,164], a combination of recursion and raster scanning, such that raster scanning is used along one direction, the rows, and recursion is reserved for the other(s), to change row. The hybrid technique can be directly substituted into any program using the recursive technique, just by replacing the recursive function, without any further modification, but a significant decrease in the number of consecutive recursive function calls, and enhanced performance, as I will show.

2.2 The algorithm

In the following, while not stated otherwise, I assume a binary mask stored into random access memory as a two-dimensional integer array, $p_{x,y}$, where every pixel is located by row y and column x. The labelling is initialized with negative labels for foreground pixels and 0 for background pixels. Object labels are positive integers. The hybrid technique, as the pure recursive technique, assigns labels consecutively, such that when the labelling is complete, the last label equals the total number of objects in the image.

The pure recursive technique has a main loop that scans the labelling in search of a negative label and calls a recursive function, which assigns the current cluster label to the site and explores its neighbours. If a negative label is found, the recursive function calls itself for that neighbour and the process is repeated. When the last site of the component has been thus labelled, the recursion ends, the current label is increased and the scan along the image is continued in search of the next component. Thus, a single pass over the image suffices to label all components with definitive consecutive labels, and each component is wholly labelled at one stroke, allowing component characterization (computation of component parameters) and percolation detection (checking if opposite ends of the image are reached) during the labelling. It also permits labelling a single component without having to label the whole image, which is, for instance, the idea behind Wolff's cluster update algorithm [271] for generation of samples of the Potts model. The drawback is that it requires a recursive call for each site in the component, many of them consecutive, and, therefore, the recursion depth can be very high, thus demanding a big stack, even in moderately sized images, if a high fraction of foreground pixels or very compact components are expected.

The solution: Let's use iterative scanning to explore along a direction, say, without loss of generality, along rows, and let recursion for the rest of directions (one in 2D, two in 3D). Thus, the number of recursive calls, now used only to change row, is drastically reduced. Let a *burst* be a set of connected pixels along a row bounded by background pixels or the image boundaries. I use a recursive function which iteratively labels every pixel in a burst before

exploring the adjacent rows (two in 2D, three in 3D). If a foreground pixel is found, a recursive call is issued for the burst on that row. Thus, the number of recursive calls is reduced from one per pixel in the component to one per burst in the component. Small components may have very few pixels per burst, and thus the improvement may not be that significant for them, but small components never cause recursion depth troubles (stack overflow). Big components, more prone to requiring a high number of consecutive recursive calls, will also have a greater number of pixels per burst, and, therefore, the decrease of the recursion depth is very significant. Regarding speed, the overhead of repeated function calls is higher than that of iterative scanning, and, therefore, the decrease in the number of recursive function calls balances the slightly increased complexity of the recursive function when compared to the pure recursive technique. Thus, the relaxation on stack requirements should come at no cost in overall performance.

Any program using the recursive technique can be updated to hybrid labelling just by replacing the recursive function at the core of the algorithm, here called Label(), without any further change. The pseudocode in Algorithm 1 illustrates the algorithm for 4-neighbourhoods in 2D images.

Algorithm 1 2D hybrid labelling for 4-neighbourhoods

$0 \rightarrow label$
for all x, y **if** $p_{x,y} < 0$ {**increase** *label*, Label(x, y)}

Label(x, y)
    while $p_{x-1,y} < 0$ **decrease** x
    $x \rightarrow m$
    while $p_{m,y} < 0$ {$label \rightarrow p_{m,y}$, **increase** m}
    while $x < m$
        if $p_{x,y-1} < 0$ Label$(x, y-1)$
        if $p_{x,y+1} < 0$ Label$(x, y+1)$
        increase x
end of Label

The image is scanned in search of foreground pixels, recognized by their

negative labels. Any arbitrary search order may be used, thus priorizing specific areas of the image, or a given object. When the recursive function, Label(), is called, it labels the entire object, using the value stored in *label*, a global variable, before returning control. After the scan, all objects are labelled and *label* contains the total number of objects in the image. Label() scans the image along x (decreasing x, *backscan* hereafter) until the first pixel in the burst is found. Its location in the row is kept in x. Then the scan proceeds forward from x (*forward scan* hereafter), assigning the current label to the pixels in the burst, until its end. This end location is kept in m. Last, the forward scan is repeated (*second forward scan*) to search for foreground pixels in the adjacent rows ($y-1$ and $y+1$)[3]. If any is found, Label() calls itself to repeat the procedure on the new burst. Thus the number of recursive calls is reduced to at most one per burst, because Label() enters every burst only once. The order in which backscan, forward scan, and second forward scan are executed ensures the complete labelling of the whole object. To label a single, given object, for instance in response to a mouse click from the user of a graphic interface, the scan of the whole image is replaced by a single call to the recursive core of the algorithm, Label(), with the coordinates of the mouse.

A slightly faster version labels pixels in the backscan, thus shortening the length of the forward scan. This is accomplished by keeping in m the location of entry in the burst, then backscanning in x, followed by a forward scan in m, and a second forward scan from x to m, see Algorithm 2.

The reader may be wondering why if Algorithm 2 is faster should anyone be interested in Algorithm 1, when they are virtually identical. Well, there is a subtle difference: In the first, slightly slower version, all pixels are labelled by the same instruction. In the faster version, pixel labelling takes place in

[3]The keen programmer will easily notice that a sound implementation deserves a single decrease of y before entering the second forward scan to convert the search in $y-1$ and $y+1$ into a search in y and $y+2$. This decreases in $2N-1$, with N the number of pixels in a component, the number of integer subtractions per component. No need to obscure the pseudocodes above, however. Also note that too keen a programmer may be thinking in introducing a local variable in Label() to hold $y+1$, thus affording an additional $2N-1$ integer additions per component. He should think twice, however, and remember the main goal: Decreasing stack use.

Algorithm 2 Faster 2D hybrid labelling for 4-neighbourhoods

$0 \rightarrow label$
for all x, y **if** $p_{x,y} < 0$ {**increase** $label$, Label(x, y)}

Label(x, y)
    $x \rightarrow m$
    while $p_{x-1,y} < 0$ {**decrease** x, $label \rightarrow p_{x,y}$}
    while $p_{m,y} < 0$ {$label \rightarrow p_{m,y}$, **increase** m}
    while $x < m$
        if $p_{x,y-1} < 0$ Label$(x, y-1)$
        if $p_{x,y+1} < 0$ Label$(x, y+1)$
        increase x
end of Label

two different places in the code (backscan and forward scan). This may be an annoyance when component characterization involves complex operations and labelling speed is not that critical.

Using 8-neighbourhoods only requires changing the limits of the second forward scan. This is achieved just by changing the while conditions in the backscan and the second forward scan. See Algorithm 3 for the faster version.

The extension to 3D images is straightforward. A new index appears, and the second forward scan grows to explore the two additional adjacent rows in the new index. See Algorithm 4 for 4-neighbourhoods.

The pseudocodes are particularized for binary masks for clarity. If there are more than two classes in the mask, $p_{x,y}$ can be initialized with as many negative integer labels as different identity criteria $\mathcal{A}_n$ are in the image, and Label() adapted to search for the negative label that triggered the first call into the current component.

For obvious reasons, hybrid labelling requires a programming language allowing recursive function calls. I recommend C or C++, because their function calling protocol is very efficient [236]. The function calling protocol has special relevance when dealing with techniques using recursivity, and the

Algorithm 3 Fast 2D hybrid labelling for 8-neighbourhoods

$0 \rightarrow label$
for all x, y **if** $p_{x,y} < 0$ {**increase** *label*, Label(x, y)}

Label(x, y)
    $x \rightarrow m$
    while $p_{x,y} < 0$ {$label \rightarrow p_{x,y}$, **decrease** x}
    while $p_{m,y} < 0$ {$label \rightarrow p_{m,y}$, **increase** m}
    while $x \leq m$
        if $p_{x,y-1} < 0$ Label$(x, y-1)$
        if $p_{x,y+1} < 0$ Label$(x, y+1)$
        increase x
end of Label

Algorithm 4 Fast 3D hybrid labelling for 4-neighbourhoods

$0 \rightarrow label$
for all x, y, z **if** $p_{x,y,z} < 0$ {**increase** *label*, Label(x, y, z)}

Label(x, y, z)
    $x \rightarrow m$
    while $p_{x-1,y,z} < 0$ {**decrease** x, $label \rightarrow p_{x,y,z}$}
    while $p_{m,y,z} < 0$ {$label \rightarrow p_{m,y,z}$, **increase** m}
    while $x < m$
        if $p_{x,y-1,z} < 0$ Label$(x, y-1, z)$
        if $p_{x,y,z+1} < 0$ Label$(x, y, z+1)$
        if $p_{x,y+1,z} < 0$ Label$(x, y+1, z)$
        if $p_{x,y,z-1} < 0$ Label$(x, y, z-1)$
        increase x
end of Label

different calling conventions offered by a given compiler should be checked for optimized performance. In C, hybrid labelling can be implemented with very compact, clean code, even shorter than the pseudocodes above. Alternatively, the simplicity of the algorithm renders assembler a reasonable option.

One of the main features of recursive labelling is its ability to easily characterize the objects during their labelling, in a single pass over the image. In classical recursive labelling the recursive function is called once for every pixel belonging to an object, and computations such as increasing object area, averaging pixel coordinates, or recording extreme coordinates for the bounding box, can be performed anywhere within the recursive function. Such computations are equally possible with the hybrid technique, provided that they are performed at the right place. In Algorithm 1 a single call to Label() involves the labelling of several pixels, a whole burst, in the forward scan, which is the right place for the computations, to ensure that they are performed once for each pixel. If a computation involves pixel coordinates, just note that during the forward scan in Algorithm 1 they are referred as (m, y), because x is used to keep the initial position for the second forward scan. See Algorithm 5 for an example of characterization during labelling. If the faster version is used (Algorithms 2 to 4), the characterization code has to be repeated in the backscan and the forward scan.

Algorithm 5 2D hybrid labelling and surface area for 4-neighbourhoods

$0 \rightarrow label$
for all c $0 \rightarrow area_c$
for all x, y **if** $p_{x,y} < 0$ {**increase** $label$, Label(x, y)}

Label(x, y)
    while $p_{x-1,y} < 0$ **decrease** x
    $x \rightarrow m$
    while $p_{m,y} < 0$ {$label \rightarrow p_{m,y}$, **increase** $area_{label}$, **increase** m}
    while $x < m$
        if $p_{x,y-1} < 0$ Label$(x, y-1)$
        if $p_{x,y+1} < 0$ Label$(x, y+1)$
        increase x
end of Label

The ability of the recursive techniques to sequentially label entire objects with a definitive label in a single pass, permits very fast detection of spanning objects (percolation check). When using two-pass algorithms the

whole image has to be labelled before checking for percolation, because several conflicting labels may be temporarily assigned to different objects. The presence of spanning objects is detected searching for matching labels in opposite borders of the image. The whole table of equivalence classes has to be built before the check, in order to correct every conflicting label in the borders. With hybrid labelling, though, spanning objects can be detected during the single labelling pass, rendering the final search in the borders unnecessary. If the algorithm is configured such that the raster scans proceed along the direction of interest, two flags can signal if two opposite borders of the image are reached during the labelling of an object. If both flags are set after labelling an object, it is a spanning object, identified by the current label.

However, a different, straightforward approach to percolation detection can be used. By definition, only the objects touching a border of the image can be spanning objects. By restricting the labelling to the first line in the image, we only have to check if we reach the last line during the labelling. Thus, at most only the objects touching the first line of the image are labelled, in general just a fraction of the objects in the image. Moreover, if percolation does exist, the labelling can be stopped as soon as a pixel of the spanning object in the last line is found. In this way, recursive percolation check significantly improves the iterative approach. While the technique can be equally used with both the recursive and the hybrid approach, the advantageous use of the stack by hybrid labelling prevails. And it is also faster, because it uses iterative scanning to move along the direction of interest. This is not the only alternative to labelling the entire lattice. Ziff, Cummings and Stell [277] proposed a hull method, only for two dimensional images, that surrounds the spanning component, thus checking for percolation without having to label all the image. Depending on the specific image configuration, this may be a faster approach, but whenever a direct path, or a quasi direct path, from border to border exists, more probable at higher densities, a hybrid percolation check just cuts through from border to border in one or a few iterative `for` loops. Thus, while the hull method runs at least in time $O(N^{7/8})$ in the criticality region, hybrid percolation check could be

running in time $O(N^{1/2})$ or close.

Thus, checking for percolation during the labelling just requires checking if the extreme coordinates, x and m, reach opposite boundaries of the image within a component. However, if percolation check is the only purpose of the labelling, the algorithm is greatly simplified and accelerated: Scanning the last line in the image is enough; if x reaches the first line during the labelling, the image percolates and the labelling can be halted. Using the last line as the "entry" point is justified by the reverse tracking of the backscan. See Algorithm 6 for a very fast percolation detector.

Algorithm 6 Very fast 2D percolation detection

$0 \rightarrow$ *percolation*
for all y **while** not *percolation* **if** $p_{x_{\max},y} < 0$ Label$(x_{\max}, y) \rightarrow$ *percolation*

Label(x, y)
    while $p_{x-1,y} < 0$ **decrease** x
    if $x = 0$ **return** 1
    $x \rightarrow m$
    while $p_{m,y} < 0$ $\{1 \rightarrow p_{m,y}$, **increase** $m\}$
    while $x < m$
        if $p_{x,y-1} < 0$ Label$(x, y-1)$
        if $p_{x,y+1} < 0$ Label$(x, y+1)$
        increase x
    return 0
end of Label

2.3 Benchmark results

As it can be shown by the pseudocode in the section above, the hybrid algorithm succeeds in retaining the advantages of the recursive approach (simple, compact code, labelling a single object without the need of labelling the entire image, using definitive, consecutive labels in a single pass, no need of additional data structures, easy object characterization during the labelling, including detection of spanning objects), while reducing the number of recursive calls, theoretically, at least, in a considerable extent. However, to prove the validity of the idea and the convenience of the algorithm, further evaluation beyond theoretical formulations or intuition is needed. It is yet to prove the impact on speed due to the decrease of pressure on the stack, and also to quantify this claimed enhancement on stack use.

2.3.1 Algorithmic performance

It is way too usual that systematic evaluation of algorithms for image processing and computer vision is neglected. For anybody arriving to the discipline with a background in physics or a formal computer sciences education, this is a source of frustration. The reasons for this state of affairs may be varied, but they often reduce to a common ground. First, there is no doubt that some problems in computer vision are as hard as new, and classical rigorous error or performance analysis is out of discussion. Second, much emphasis is put on mathematical rigor in mathematical formulations supporting new methods or strategies. However, there seems to be a large breach between the foundations of methods and approaches and their algorithmic embodiment. Once the theoretical ground is established, several algorithms will come to flourish on it with more or less fortune, without anybody asking for proof of performance beyond the convenience of use due to this or that feature. This is generally due to the fact that there is usually as much distance from the theoretical formulation of a problem to the specific implementations as there is from the images to the real world they try to reflect. This tendency has somehow reversed in the past few years, where we have assisted, for instance,

to real time demonstrations of performance of given methods during conferences or scientific meetings, even if, in general, within the framework of very specific applications. Proof of the growing concern about this extreme, are the efforts to establish generally accepted rules for the performance analysis of computer vision algorithms [100]. The three major criteria for the performance of computer vision algorithms are:

- Successful solution of task. This is a top priority. But also the precise definition of the tasks for which it is suitable and what are the limits.
- Accuracy. This includes an analysis of the statistical and systematic errors under carefully controlled conditions.
- Speed. One of the major criteria defining the applicability of an algorithm.

There are different ways to evaluate algorithms according to the criteria above. Ideally this should include three classes of studies:

- Analytical studies. Mathematics to verify algorithms, check error propagation, and predict catastrophic failures.
- Performance tests with computer generated images. These provide performance data under carefully controlled conditions.
- Performance tests with real-world images. The final test for practical applications.

There is no reason to exclude connected component labelling algorithms from this sort of analysis. Proof of successful solution of task is in most of cases self-evident, as it is the case with the hybrid technique. Explaining how the algorithm works, supported on some pseudocode, as in the section above, renders any proof of workability superfluous[4].

[4]Well, this assertion should be tinged. A necessary condition is that the audience knows about computer programming, including recursion. This, for instance, was not the

Accuracy is not of application in the domain of working connected components algorithms. Either they work or not. If a connected component algorithm fails to label some connected components, it is plainly failing. This is not the case of either classical techniques, nor is it the case of hybrid labelling. A different subject matter is when a connected component algorithm is at the core of a segmentation algorithm, say textural region growing, where the criteria for defining "foreground" and "background" (in labelling terminology) pixels on the fly, i.e. a set of probably complex identity criteria $\mathcal{A}_n$, will determine the degree of accuracy of the global segmentation technique. But this is not related to the accuracy of the labelling. Once a pixel is defined as foreground or background, a working labelling technique will never fail to ascribe it to the adequate component.

However, limits of application, task suitability, and, of course, speed, are major issues regarding connected component labelling which deserve careful study. Analytical study of algorithmic complexity to compare the performance of the recursive, the iterative, and the hybrid techniques is not adequate, though. Mathematical analysis would in this case be restricted to speed because accuracy is out of scope. Still, the different nature behind each technique (recursivity and iterativity) renders complexity order estimations (the "big O" analysis) less than significant. The number of times a pixel is visited, either for checking its label, either to assign or reassign a label, can be estimated for the general case for each algorithm —and I say "estimated" because it depends on the structure of the components in the image, and the precise implementation of each algorithm—, but the results would induce wrong performance estimations, as the basic technique used for those "visits" —iterative loops in the iterative techniques, recursivity in the recursive techniques, a mixture in the hybrid approach— is different and therefore involves different basic processor operations. Moreover, the general estimation would greatly differ from concrete application scenarios, because the influence of the structure of the components in the image, and there-

case of a computer vision journal referee which asked for proof that the hybrid technique would work on a ring-shaped object, because he expected "the various instances of the algorithm working in opposite directions to collide at the bottom"! The author decided he was addressing the wrong journal...

fore of the image itself, is far too large. An extreme instance of this is just another case of critical phase transition in percolation: The behaviour of many magnitudes and processes changes abruptly in the vicinity of percolation. Connected component labelling is just another process running on the lattice, subject to the same critical phase transition effects. The density of foreground pixels, and thus the probability of percolation, affect therefore the behaviour of the labelling techniques in a non trivial way.

This is where synthetic images, also called neutral scenarios, or neutral landscapes, depending on the context, enter the game. Sets of random samples of images provide controlled environments with statistical significance to study and compare the performance of algorithms. Synthetic random image models adequatelly designed yield general results independent of specific applications and conditions, while allowing to analyze the expected behaviour in a family of uses or field of application by carefully designing the models so that the image samples have general characteristics corresponding to those expected in the specific fields of application. In the following I will describe several image sets I generated to test the hybrid technique versus the classical approaches, and show the results of the tests.

2.3.2 Sample image sets

I generated three sets of synthetic binary images (masks) of size 512×512: the Uniform set, the Ising set, and the Block set. Uniform is composed of images with a pseudorandom uniform distribution of foreground pixels, with density ranging from 5% to 95% in steps of 5%, 1 000 different images for each density. Figure 2.1 shows some sample Uniform images with different densities of foreground pixels. A magnification of a 25×25 window is included for each image, to better appreciate the shapes of the objects. Figure 2.2 shows the average number of objects per image, their average area, the area of the bigger object, and the percentage of images with at least one spanning object —"percolating" images. A threshold, called the percolation threshold, p_c, is clearly visible at about 60% of foreground pixels. Uniform images are the equivalent of the so-called neutral landscapes in landscape analysis [269], and provide images without any recognisable spatial structure where the behaviour of the algorithms is clearly determined by the density of foreground pixels, i.e. they act as "control" images.

The Ising set is made of samples of size 512×512 of the Ising model,

$$P(\sigma) \propto \exp(\beta \cdot \sum \delta[\sigma_i, \sigma_j]), \beta = \frac{J}{k_B T} \tag{2.1}$$

The parameter β governs the average cluster size in a binary random distribution of pixels. I generated the samples with the Swendsen-Wang algorithm [238], a Markov chain Monte Carlo simulation [148, 169] of the Ising model [88]. The Ising set is composed of images with β from 0.10 to 1.00 in steps of 0.05, 1 000 images for each value of β. Figure 2.3 shows some sample Ising images with different β values. Figure 2.4 characterizes the objects in the Ising set. A phase transition occurs when β is about 0.85, where a spanning object suddenly appears. The Ising set represents a neutral approximation to typical scenes with a very high number of components of intricate and arbitrary shape, such as thematic maps, crystal lattices, or satellite imagery.

The Block set is composed of images with a pseudorandom distribution of 30 square blocks of foreground pixels with overlap allowed, with block size

ranging from 10×10 to 190×190 in steps of 10, 1 000 different images for each block size. Figure 2.5 shows some sample images from the Block set with different block sizes, figure 2.6 the characterization data. Block images represent a highly structured, specific spatial configuration, resembling scenes with relatively few, compact objects, typical of machine vision applications.

I labelled the three sets with the recursive technique, the iterative technique —the fastest iterative Union Find implementation I have been able to write—, and the hybrid technique. According to their different approaches to the problem, the performance of the recursive technique should depend mainly on the number of foreground pixels, while the hybrid and iterative techniques should also show a dependence with the spatial arrangement. Thus, the Uniform set is *a priori* the worst case scenario among those proposed for the hybrid and iterative techniques, while the classical recursive technique at the same density of foreground pixels should not show any marked preference. I recorded the average time per image for each algorithm, and monitored the recursive function calls of each of the two algorithms issuing recursive calls. The total number of recursive function calls influences the speed. The maximum number of consecutive recursive function calls is the recursion depth, which determines the size required for the stack and thus is directly related to the probability of stack overflow. All data that follows was obtained on a desktop PC with processor iPentium4 2.4 GHz, 512 Mbytes RAM, under OS MS-WindowsXP, with the C/C++ compiler of the Borland C++Builder 6 package.

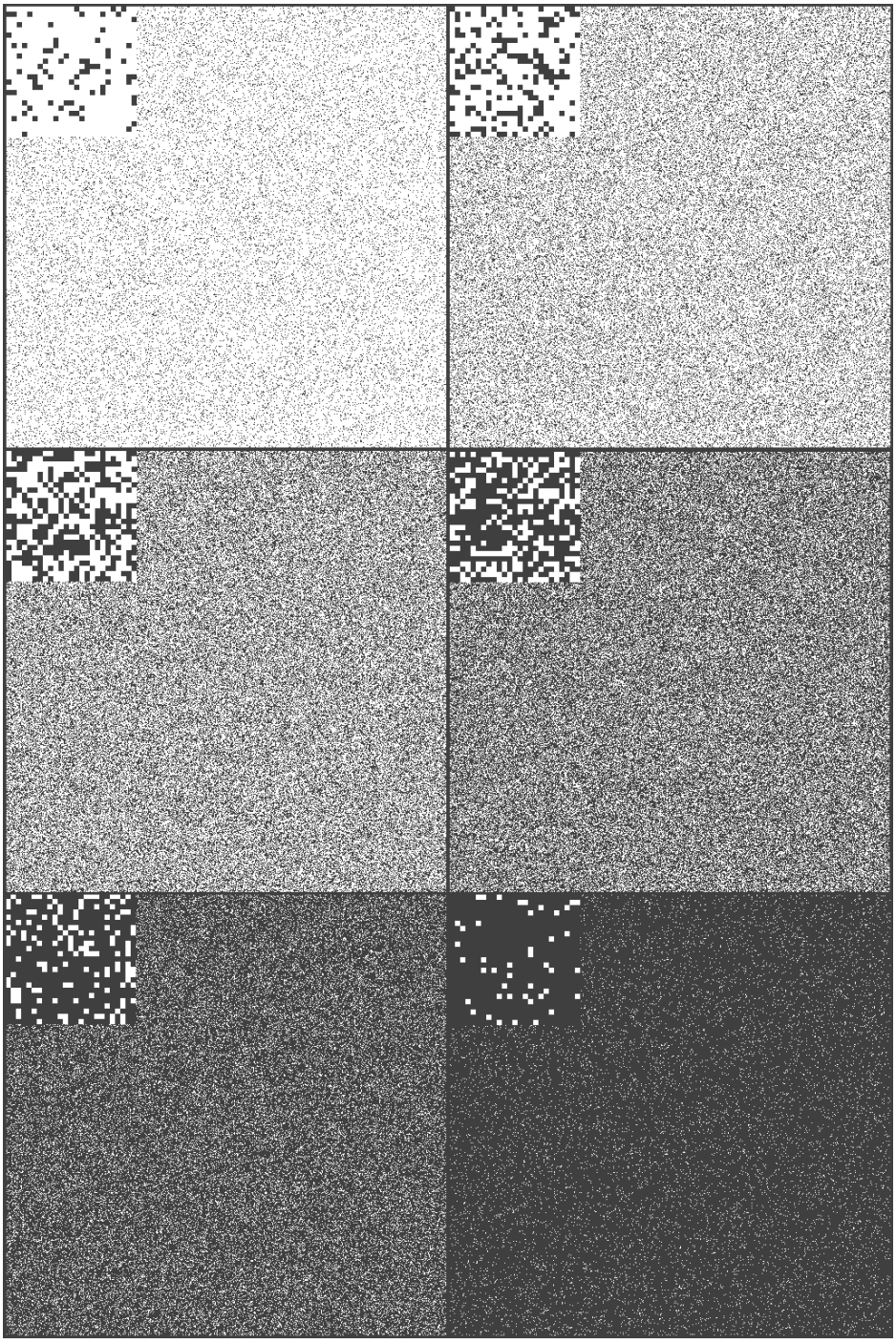


Figure 2.1. *Some samples from the Uniform test sets, density of foreground pixels 10%, 25%, 45%, 60%, 80%, and 90%. Samples include a magnification of the detailed structure.*

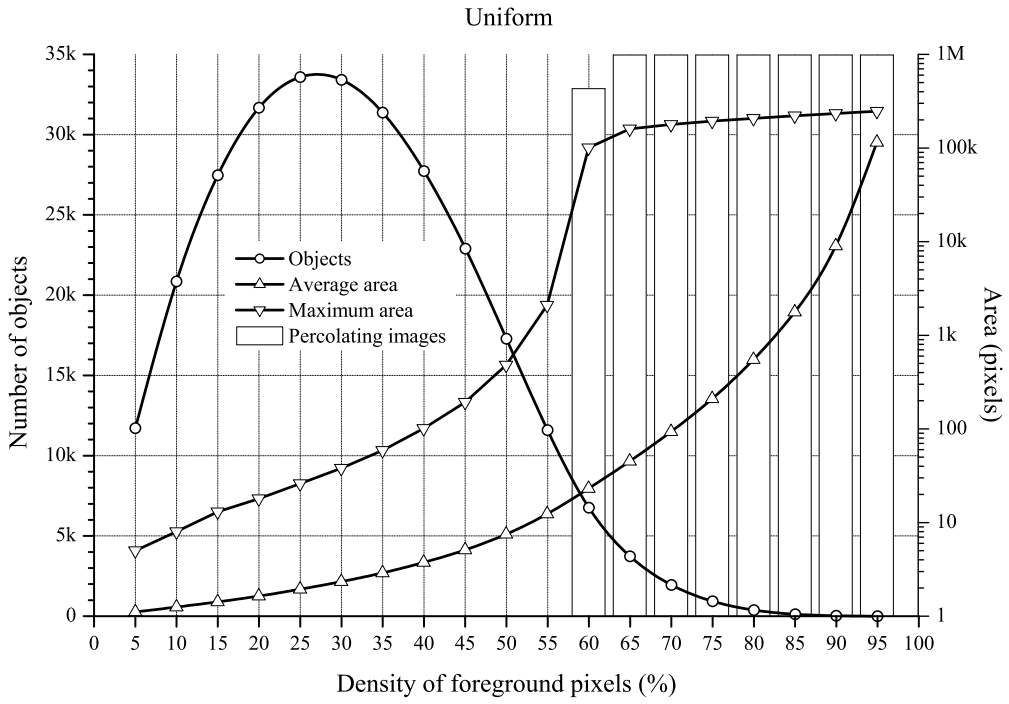


Figure 2.2. *Characterization of the Uniform set: Number of objects, average object area, average area of the larger object, and percentage of percolating images. The linear scale for the percolation data is not shown, but the rightmost bars are 100%.*

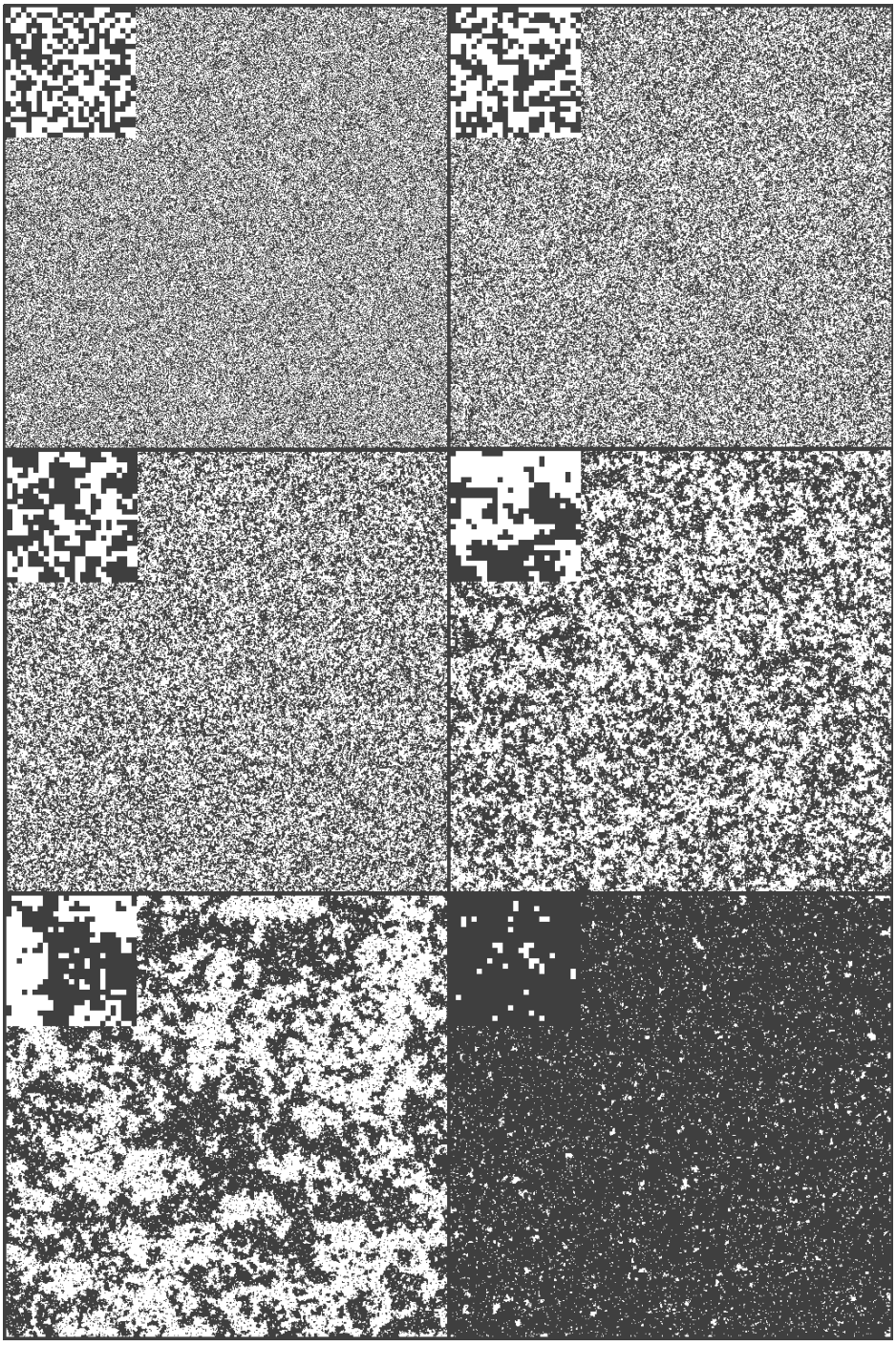


Figure 2.3. *Some samples from the Ising set, β values* 0.05, 0.30, 0.50, 0.75, 0.85, 0.95.

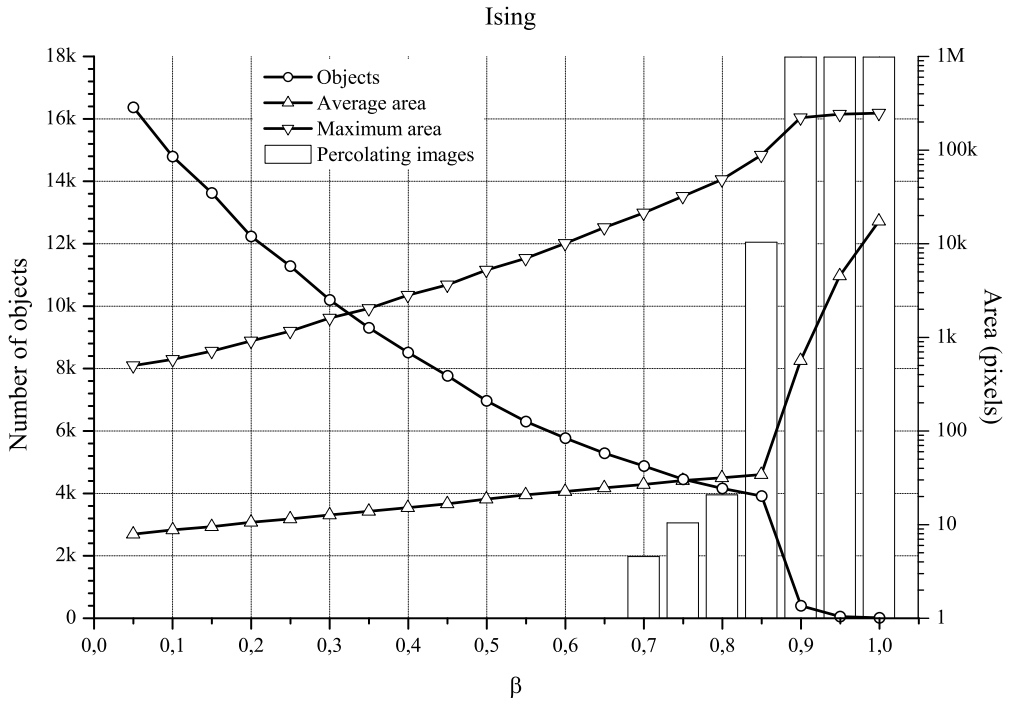


Figure 2.4. *Number of clusters, average cluster area, average maximum cluster area, and probability of percolation for the Ising set. The highest bars indicate 100%.*

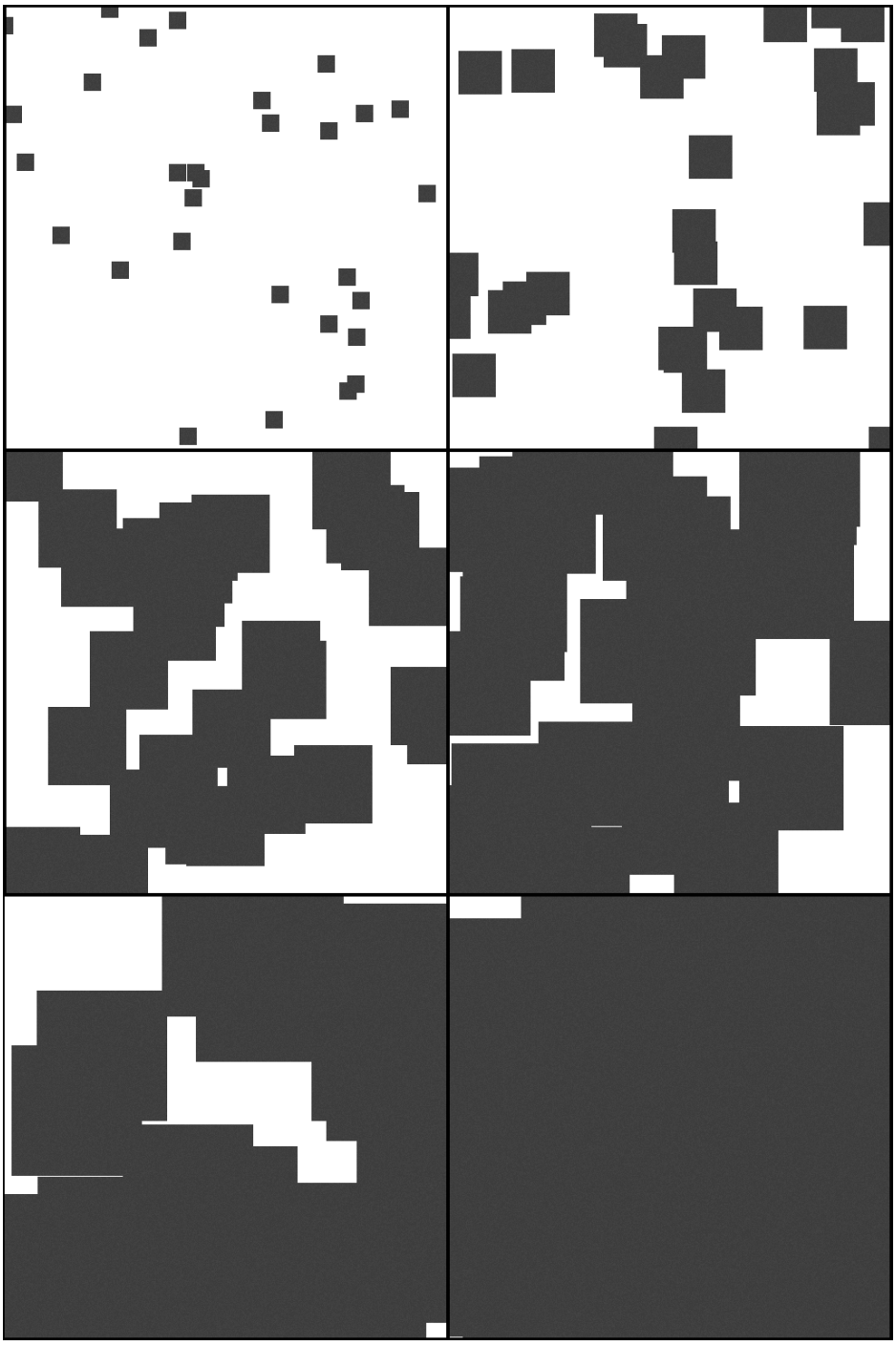


Figure 2.5. *Some samples from the Block set, block sizes* 20×20*,* 50×50*,* 90×90*,* 120×120*,* 150×150*, and* 180×180*.*

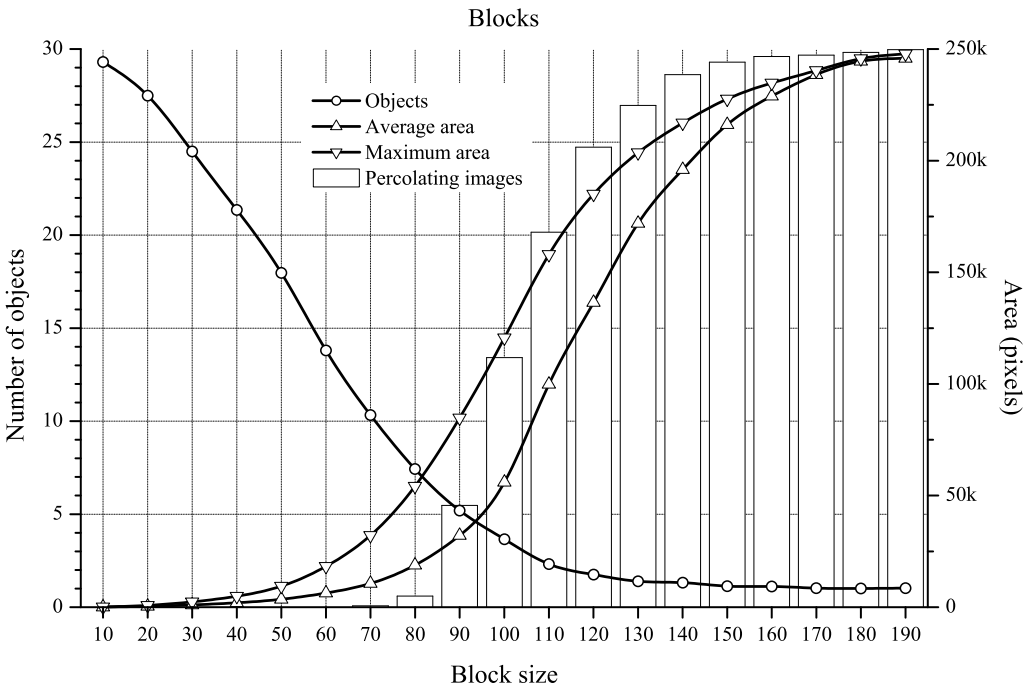


Figure 2.6. *Characterization of the Block set: Number of objects, average object area, average area of the larger object, and percentage of percolating images. The linear scale for the percolation data is not shown, but the rightmost bars are 100%.*

2.3.3 Stack use

Figure 2.7 shows the recursive function calls data with the Uniform images. As expected, hybrid labelling uses less recursive function calls per image, but the critical figure is the recursion depth, which determines the required stack size. Recursive labelling needed a stack big enough for 223 351 consecutive recursive function calls (at the highest density of foreground pixels, 95%) while the hybrid technique needed room for just 20 820 consecutive recursive function calls, at a density of 75%, which is 9%. Thus, hybrid labelling required a stack 11 times smaller than recursive labelling, which needed 5 Mbytes to avoid stack overflow errors during the tests with the 512×512 images.

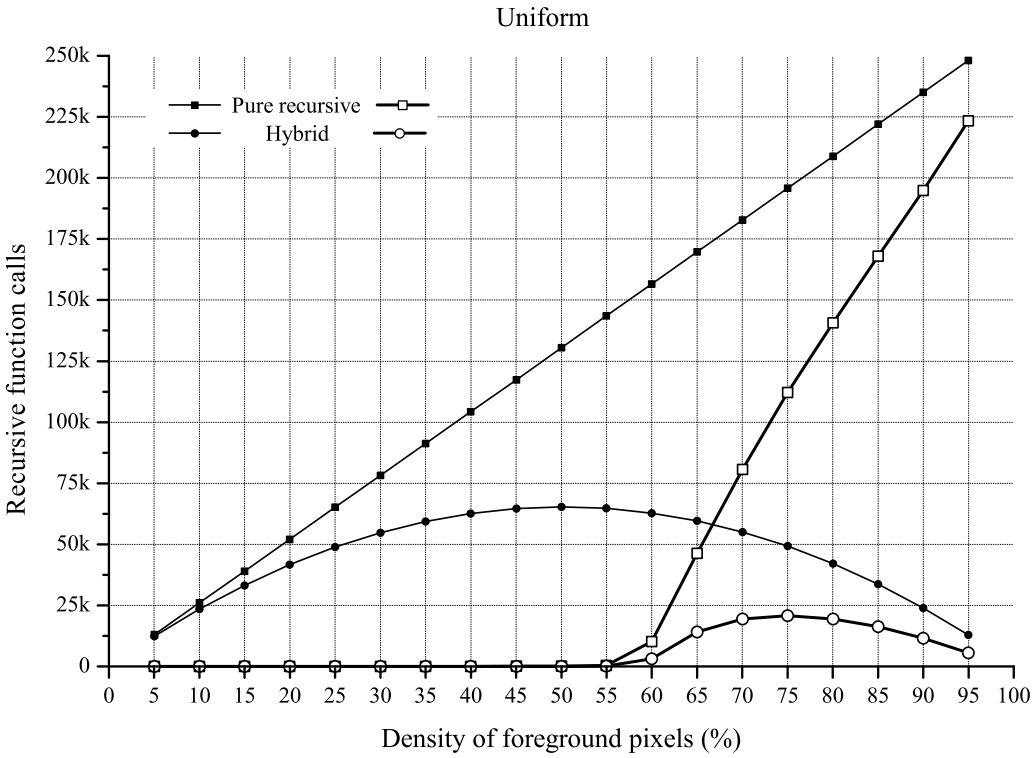


Figure 2.7. *Number of recursive calls and recursion depth for the Uniform set. The thicker lines with hollow symbols show recursion depth (maximum consecutive recursive calls).*

The recursion depth of the hybrid technique reaches the thousands only

above p_c (figure 2.2), while it is only 168 just before, at 55%. This is explained by the sudden appearance of a large, spanning object, as shown by the curve of average area of the biggest object, figure 2.2, logarithmic scale. Below p_c, the Uniform images are composed of an increasing number of small, sparse objects, which keep steadily growing and becoming more and more complex, but still spanning a relatively low number of rows. At p_c, there is a large probability of a big number of individual objects to grow so close to each other that they become a single, large spanning object, with very intricate shape. Above p_c, this large object dominates the scenario, and keeps growing and becoming more compact —less, longer bursts—, thus requiring less recursive calls.

In the Ising set, Figure 2.8, recursive labelling results are more or less the same than those for the Uniform set if the figures are compared using the density of foreground pixels, which in Figure 2.8 is shown by the bars in the background. The maximum recursion depth was 219 513 calls for the recursive technique, at $\beta = 1.00$, while the hybrid technique decreased to a maximum 6 037, at $\beta = 0.90$, which is less than 3%, thus requiring a stack 36 times smaller, i.e. assuming the same behaviour for both algorithms with size (more about this in a few moments), the hybrid algorithm is able to label Ising images greater than $3\,000 \times 3\,000$ with the same stack required by the classical recursive technique for images of size 512×512.

Figure 2.9 shows recursive function calls curves for the Block set. Note the broken vertical axis and the corresponding scale change. They evidence the fact that hybrid labelling takes advantage of relatively compact objects. The more intricate an object, the more bursts per row, the more recursive function calls are needed to label it, and vice versa. Thus, the maximum consecutive recursive function calls required by the recursive technique with the Block set was 243 071 while that of the hybrid technique was 520, 0.2%. This implies a stack about 500 times smaller.

Figures 2.9 and 2.8 show average percentages of foreground pixels in the Ising and Block images, so that they can be somewhat related to the Uniform graphs. Thus the effect of spatial structure may be separated from the

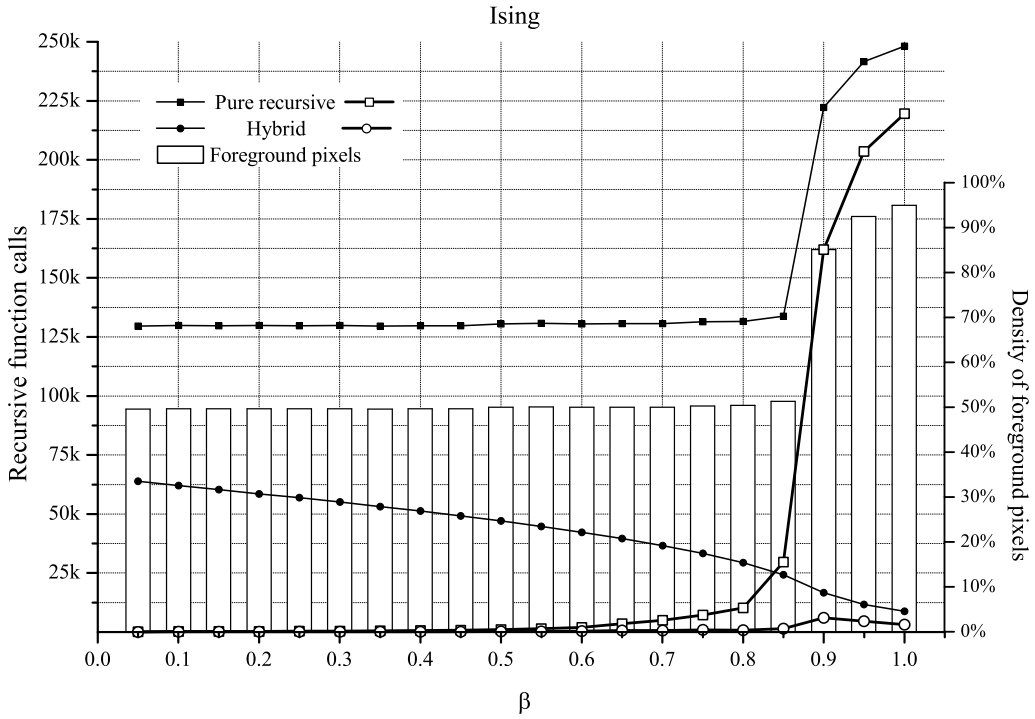


Figure 2.8. *Number of recursive calls and recursion depth for the Ising set. The thicker lines with hollow symbols show recursion depth (maximum consecutive recursive calls). The bars show the average density of foreground pixels.*

dependence on the quantity of foreground pixels. With the recursive technique, the total number of recursive calls does not depend on the spatial arrangement of the foreground pixels, while recursion depth depends on the average object size. With the hybrid technique, total recursive function calls and recursion depth are clearly dependent of the spatial structure of the image. This dependency is positive: In the worst case (Uniform set), hybrid labelling clearly outperforms recursive labelling in stack use; from there on, spatial structure only increases this advantage.

I also performed some tests with the iterative version of the classical recursive technique. The iterative version is obviously not recursive, but it still uses a stack —a custom stack, with the corresponding overhead. Therefore, the iterative version can be compared to the recursive algorithms by means of stack depth, the counterpart of recursion depth. They denote in both cases

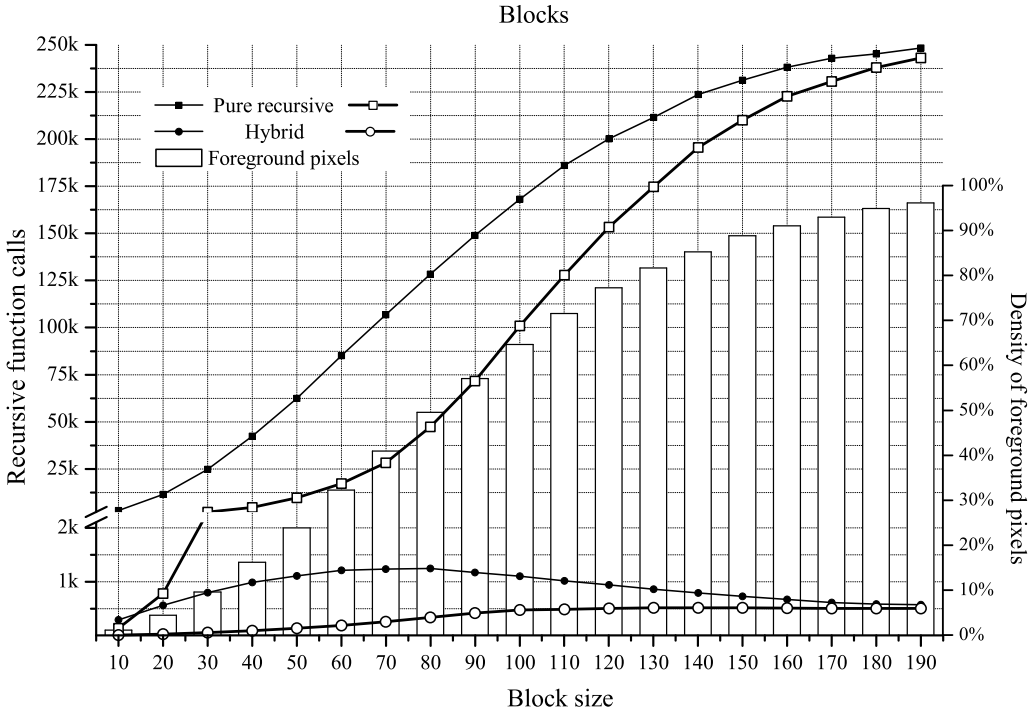


Figure 2.9. *Number of recursive calls and recursion depth for the Block set. The thicker lines with hollow symbols show recursion depth (maximum consecutive recursive calls). The bars show the average density of foreground pixels.*

the maximum number of entire sets of variables per pixel location pushed into the stack (thus measured in "calls", not in bytes). However, the sets are smaller in the iterative version, where only the pixel location coordinates are stored, than in the recursive algorithms, where also state and local variables have to be stored. Therefore, note that the same depth implies different space requirements in each case.

The iterative version required less stack depth than the recursive version for all the test sets. This is due to the order in which labelling and exploration is forced in each version. The original, recursive version priorizes search directions which it follows to the boundary of the component before visiting the rest of neighbours of the initial pixels, which remain in the stack. The iterative version labels all neighbours of each pixel and pushes them into the stack before proceeding to the next pixel in the stack, and each pixel

is popped out of the stack before exploring its neighbours. This results in less consecutive pushes, although total pixel locations pushed are the same, because all pixels in each component are pushed once and only once into the corresponding stacks in both algorithms. In spite of this reduction, stack depth is still considerable: The iterative version required for the Uniform set a stack able to hold 115 558 pixel locations (52% of the recursion depth of the recursive version, but also more than 5 times the recursion depth of hybrid labelling), i.e. to label the 256 kb Uniform images it required a 903 kb data structure, whereas hybrid labelling, according to the recursion data above, transparently used less than 460 kb of the system stack. In the Block set, things were a little worse, with a stack depth of 129 000 locations, 53% of the recursive version, but almost 250 times the recursion depth of hybrid labelling, i.e. 1 008 kb, 4 times the size of the images, whereas hybrid labelling used less than 11 kb of the system stack.

Therefore, hybrid labelling clearly reduces the pressure on the stack. The question then is whether it pushes the limit beyond the reach of practical applications. Figures 2.10 and 2.11 show the variation of recursion depth —stack depth in the iterative version— with image size, using nine Uniform and five Block sets, with different image size, 1 000 images per size and set. The graphs show the experimental data points and polynomial curves fitted to the data. Next to each curve is its equation, the squared multiple correlation coefficient[5], and the standard deviation in thousands of calls. Note that the independent variable in the fits is image length $x = L$ in square $L \times L$ images. In the Uniform set, Figure 2.10, the three algorithms show a quadratic behaviour, i.e. recursion and stack depth grow with image size L^2, but the corresponding coefficient is one order of magnitude less for the hybrid method than for the recursive method, whereas for the iterative version is nearly half. Extrapolating these results, to reach the same recursion depth than the classical recursive method for Uniform images of size 512×512, hybrid labelling would require images bigger than $1\,700 \times 1\,700$. At that image sizes, the iterative version of the classical recursive technique would require a stack 1 180 334 pixel locations deep, i.e. a 9 Mb data structure for

[5]Coefficient of multiple determination, R^2, the closer to 1 the better the fit.

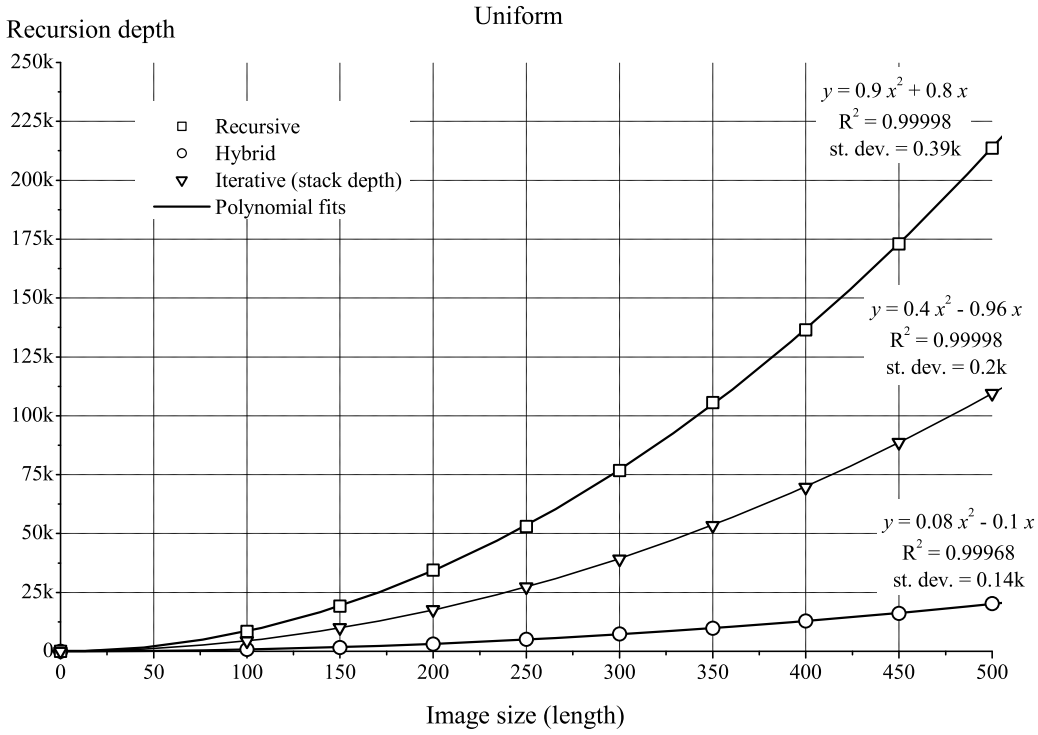


Figure 2.10. *Recursion (recursive and hybrid method) and stack (iterative version) depth (data markers) for the labeling of Uniform sets with different image sizes. The curves are polynomial fits to the data. The coefficient of determination and the standard deviation of the fits are shown with the corresponding equations. Horizontal axes show image length L corresponding to square $L \times L$ images.*

images of 2.8 Mb.

With the Block set, Figure 2.11, the hybrid trend experiences a qualitative change: Recursion depth grows with the root of image size, i.e. with image length L. Compact objects have very few, usually only one, bursts per row, and in consequence recursive calls grow with object length, not object surface area, as it is the case with the recursive technique and its iterative version. Proceeding in the same fashion as above, we can estimate that hybrid labelling would only reach similar recursion depths to that of the classical recursive method for 512×512 Block images with images bigger than $230\,000 \times 230\,000$, which is about 50 Gb of image, well over the usual capacity of nowadays systems' RAM, but also over normal (binary) image

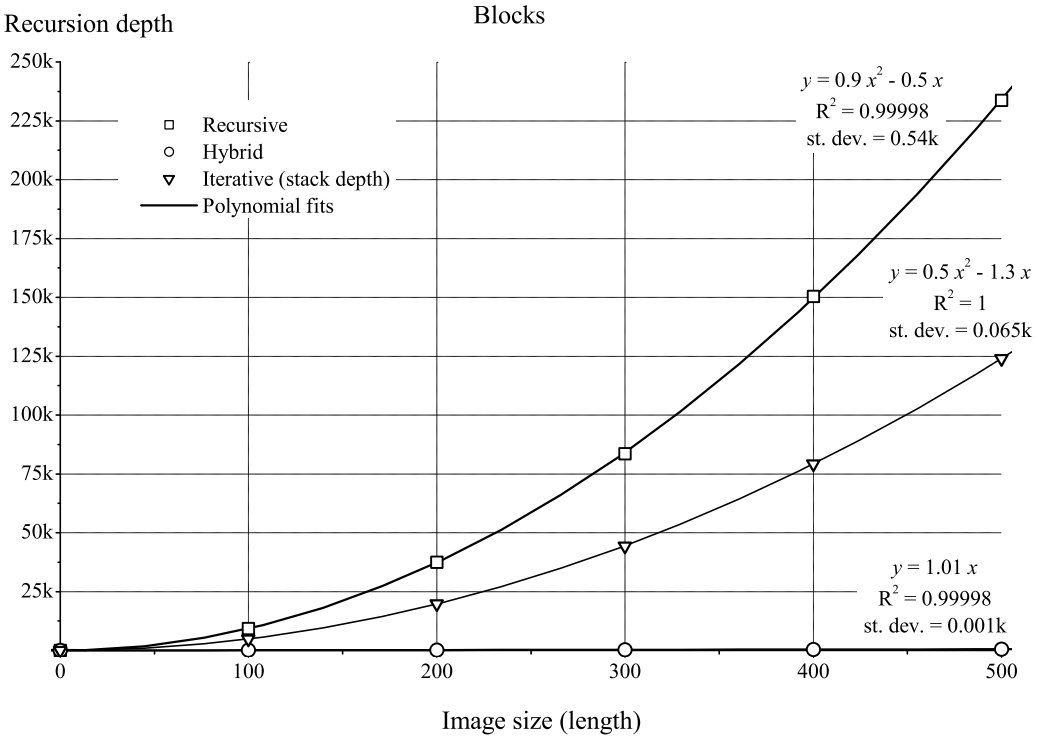


Figure 2.11. *Recursion (recursive and hybrid method) and stack (iterative version) depth (data markers) for the labeling of Block sets with different image sizes. The curves are polynomial fits to the data. The coefficient of determination and the standard deviation of the fits are shown with the corresponding equations. Horizontal axes show image length L corresponding to square $L \times L$ images.*

sizes. At that image sizes, the iterative version would require a stack deep enough for more than $2.72 \cdot 10^9$ pixel locations, i.e. a data structure occupying more than 200 Gb.

The curves give us upper (Figure 2.10, worst case, Uniform set) and lower (Figure 2.11, favourable case, Block set) limits for the recursion depth of hybrid labelling. With unstructured, purely random uniform image data, hybrid labelling will be working on or near the lower curve in Figure 2.10, recursion depth in the order of $0.08L^2$. The more structured and compact the components in the image, the closer the work point will move to the lower curve in the graph of Figure 2.11, recursion depth in the order of $1.01L$.

The main goal is thus achieved, the risk of a stack overflow error interrupting a critical process or preventing the labelling of moderately sized images is greatly diminished, if not gone. Additionally, less memory dedicated to the stack implies more memory available for other purposes. Anyway, the maximum stack size is usually not limited by the amount of available RAM, but by the system, and this upper limit is usually too low for images slightly larger than those of the test sets (512×512). In fact, I was not able to test classical recursive labelling on $1\,000 \times 1\,000$ Uniform images due to the limitation imposed by my linker on the maximum stack size (10 Mb). Hence my asseveration that nowadays the real limitation of the recursive technique is not the capability to hold the entire image in memory, but the room required in the stack. Table 2.1 summarizes the recursive calls data for the three sample sets.

Recursion depth (512×512)	**Uniform**	**Ising**	**Block**
Recursive	223 351	219 513	243 071
Hybrid	20 820	6 037	520
$\times$ **Stack reduction** (512×512)	11	36	500
Trend with size (images $L \times L$)			
Recursive	$0.9L^2$		$0.9L^2$
Hybrid	$0.08L^2$		$1.01L$

Table 2.1. *Summary of the recursion and stack data.*

2.3.4 Speed

Note that different implementations, levels of optimization, compilers, and, of course, platforms produce different results. Therefore, precise timings have to be taken just as indications of expected performance.

Figure 2.12 shows the average time per image consumed by the three techniques (classical recursive, two pass Union Find iterative, and hybrid labelling) in the labelling of the Uniform set, i.e. in absence of spatial structure. The three algorithms show different behaviour above and below the percolation threshold, p_c (see Figure 2.2). As already mentioned, this is just another instance of critical phase transition. The recursive technique

follows the curve of average object size, and is the fastest below p_c. However, its performance is no longer the best above p_c, and it clearly shows the worst performance when more than 80% of the image is covered with foreground pixels. Th eiterative version of the recursive technique (not shown), shows similar behaviour. The reduced stack depth results advantageous at the higher densities, but the custom stack shows to be slower than the system stack at the lower densities. On average, it consumed 96% of the time required by the recursive version.

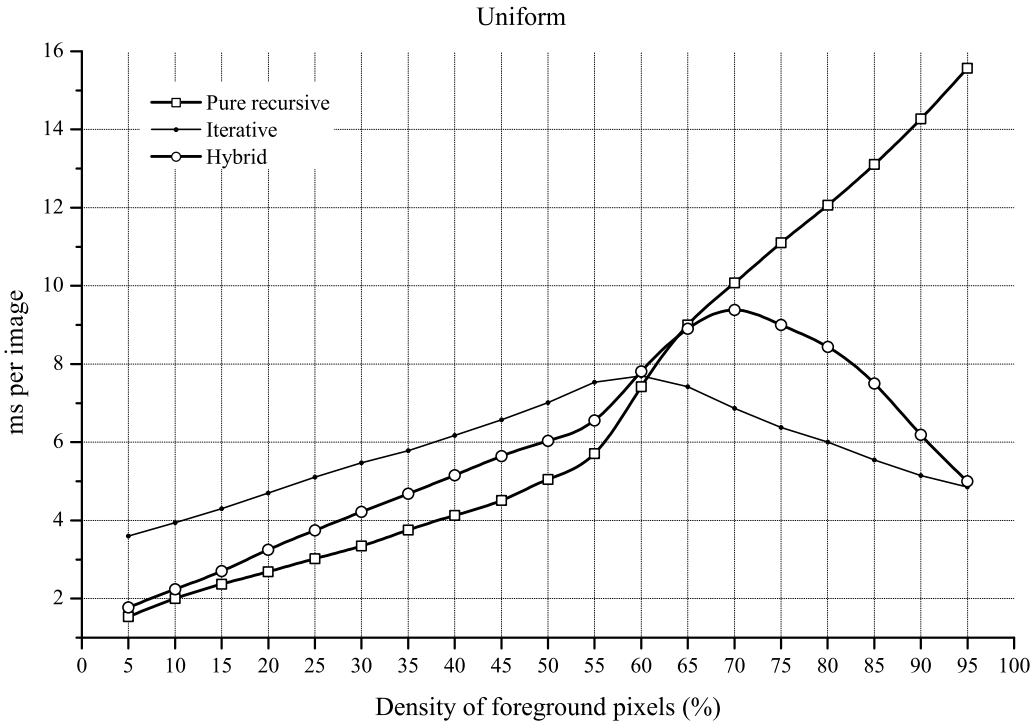


Figure 2.12. *Average time per image used by the three techniques in the labelling of the Uniform images.*

The behaviour of two pass iterative labelling reflects the cost of managing an equivalence table and the need of a second pass to translate conflicting labels. The hump around p_c is due to the highly connected "salt and pepper" configuration typical of those densities (figure 2.1). Further up, above 70% of foreground pixels, the net of objects becomes extremely dense, such that the number of equivalence classes decreases drastically. In the limit, iterative labelling of an image with no background pixels is reduced to scanning the

whole image assigning the same label to each pixel after a trivial check of the labels of the previous (in scan order) neighbours, followed by a trivial second pass where no label translation is performed. However, this does not seem to be the most interesting of scenarios.

The hybrid technique shows an intermediate curve which peaks between 65% and 75%. Here again the influence of the percolation threshold appears. There are two major factors involved: On one side, the lesser foreground pixels, the faster the labelling. On the other side, the more intricate the shape of the objects, the slower the labelling. Below p_c the former is the factor with more weight, because when the objects are small their shape is not that important. Above p_c, it is the latter, because one large spanning object appears whose shape and size determine the performance of the labelling technique.

Therefore, in absence of any spatial structure, well below p_c, when small objects predominate, hybrid labelling pays a price for its lower requirements on the stack, up to 31% more time at a density ranging from 35% to 45%. However, above p_c hybrid labelling is faster than recursive labelling. It needs just a third of the time at 95%. The bigger differences at higher densities produce an overall result in favour of the hybrid technique, which averages 83% of the time used by the recursive technique, 86% of the iterative version. The average performance of the hybrid and two pass iterative techniques is virtually the same: hybrid labelling averaged 99% of the time used by the Union Find iterative technique, even when their behaviour along the range of densities is clearly different, in response to their different *modus operandi*.

When confronted with the Ising set, Figure 2.13, the spatial structure begins to exert its positive influence on the hybrid technique and the differences increase: Hybrid labelling needed just 74% of the time used by the calssical recursive technique, and 91% of the time used by the classical iterative technique. The effect of the phase transition is clearly visible, specially affecting to the recursive technique due to the sudden increase of the number of foreground pixels, and causing a slight time increase in hybrid labelling, which is visibly attenuated with respect to the corresponding hump in Figure

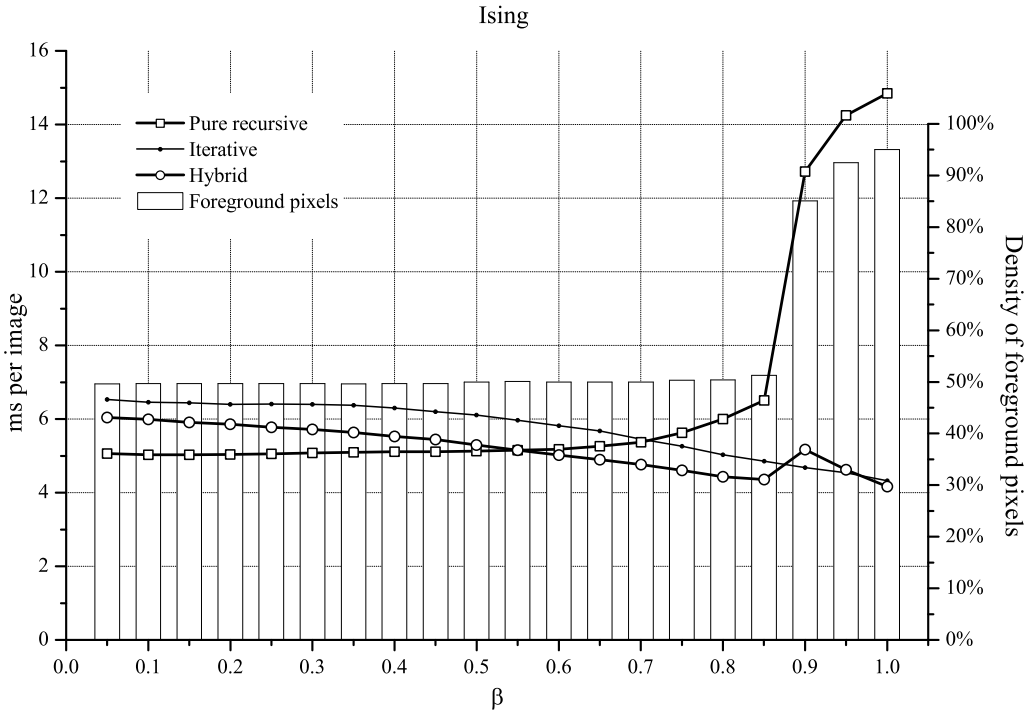


Figure 2.13. *Average time per image used by the three techniques in the labelling of the Ising samples. The bars show the average density of foreground pixels in the Ising set.*

2.12. The iterative technique is the less affected by the phase transition because the decrease in the equivalence classes balances the increased number of foreground pixels.

With the Block set, Figure 2.14, hybrid labelling definitely shows the best performance of the three techniques. The relatively flat behaviour of the Union-Find iterative and hybrid techniques shows how the spatial configuration of the foreground pixels alleviates the effect of their increasing density, while the curve of the classical recursive technique still follows the density of foreground pixels. The more foreground pixels, the more time needed by the recursive technique, with little influence of shape or spatial configuration. The iterative version shows the same trend, but modulated by the effect described for the Uniform set: a slower stack dominating the lower densities, and the lesser depth governing the higher densities, but this time the overall effect is clearly better, 59%. On average, hybrid labelling

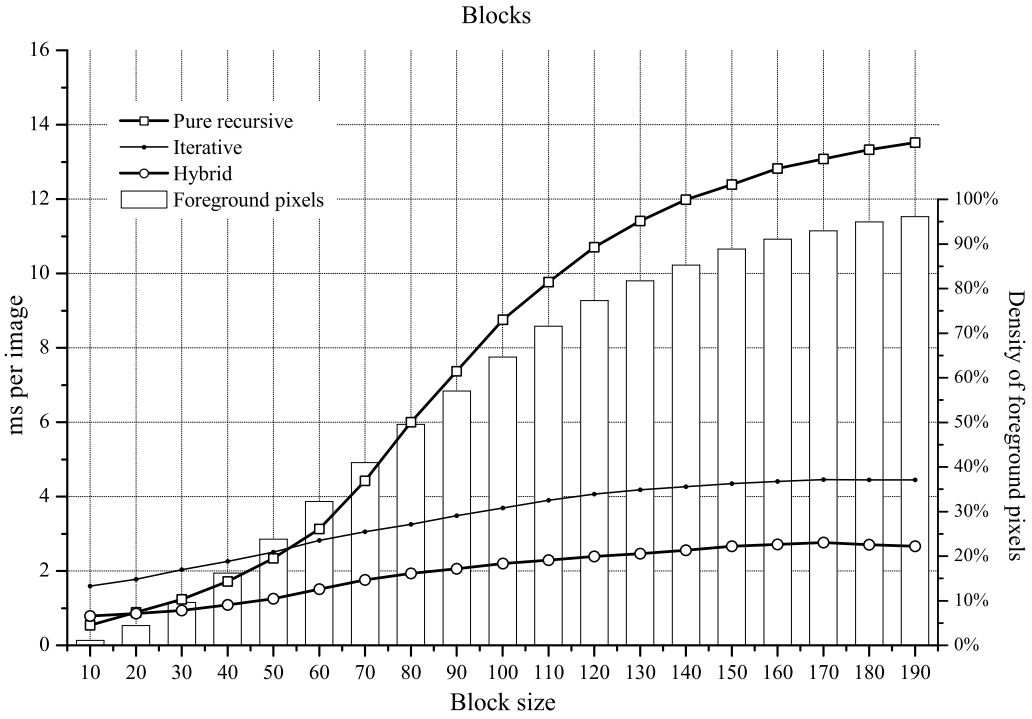


Figure 2.14. *Average time per image used by the three techniques in the labelling of the Block set. The bars show the average density of foreground pixels in the Block set.*

consumed 26% of the time used by recursive labelling (44% with respect to the iterative version), and 58% of the time used by Union Find labelling.

Again the bars in the background of the Ising and Block graphs allow linking the graphs by the density of foreground pixels. It can be seen that, in spite of the by far less dependence on spatial configuration of the classical recursive technique, there is an offset of about 1 ms between the Block and Uniform timing curves for the classical recursive technique. A quick look at Figures 2.2 and 2.6 can help to understand the reason. Regarding speed, classical recursive labelling pays little attention to the fact that pixels may be arranged in compact squared shapes or in intricate dendritic shapes while jumping recursively from neighbour to neighbour. However, the recursive exploration of neighbourhoods is preceded by an iterative scan of the image looking for foreground pixels which trigger the recursive action. The Block set averages less than 30 components per image, while the Uniform set averages

more than 20 000 components per image. The offset in the classical recursive curve accounts for the influence on the image scan of the number of times that foreground pixels are met, the label index is incremented, and the recursion initiated.

Differences with the Block and the Uniform set for the hybrid and Union-Find algorithms are determined by the different spatial structures of the test sets. Hybrid labelling will also show the same effect of the number of objects on the image scan in search of foreground pixels. However, it is quite clear that at the lower densities in the Uniform set, when objects average very few pixels in size, it cannot take advantage of the mixed iterative recursive strategy, which even becomes a burden regarding execution time. In the Block set, however, save for the smallest blocks, even at relatively low densities components have a reasonable size and compact shape which allow the hybrid technique to outperform the recursive technique, and improve its Uniform results. Regarding the two pass method, the number of equivalences in the Uniform set and therefore the time spent in the handling of the equivalence table is much higher in the Uniform and Ising sets than in the Block set, as it can be deduced from the details in the image samples (Figures 2.1, 2.3, and 2.5). Tables 2.2 and 2.3 summarize the performance data discussed.

	Uniform	**Ising**	**Block**
Average ms/image			
Recursive	6.88	7.06	7.65
Two pass iterative	5.79	5.75	3.42
Hybrid	5.74	5.23	1.99
Max. ms/image			
Recursive	15.62	14.85	13.52
Two pass iterative	7.65	6.53	4.46
Hybrid	9.38	6.04	2.81

Table 2.2. *Summary of the performance data for the classical techniques and the hybrid technique with the three* 512×512 *image sets.*

Hybrid	**Uniform**	**Ising**	**Block**
Average time			
Of recursive	83.4%	74.1%	26.0%
Of two pass iterative	99.1%	91.0%	58.2%
Max. time			
Of recursive	60.1%	40.7%	20.8%
Of two pass iterative	122.6%	92.5%	63.0%

Table 2.3. *Performance of the hybrid technique with respect to the classical techniques with the three* 512×512 *image sets.*

Image size

I repeated the performance tests with Uniform, Ising, and Block sets with different image sizes (100×100, 200×200, 300×300, 400×400, and 500×500), 1 000 images per size and set, Figures 2.15, 2.16, and 2.17, and the trend is similar to the behaviour shown in Figures 2.12, 2.13, 2.14. This shows that the behaviour detailed for the 512×512 images holds for other image sizes and can be generalized.

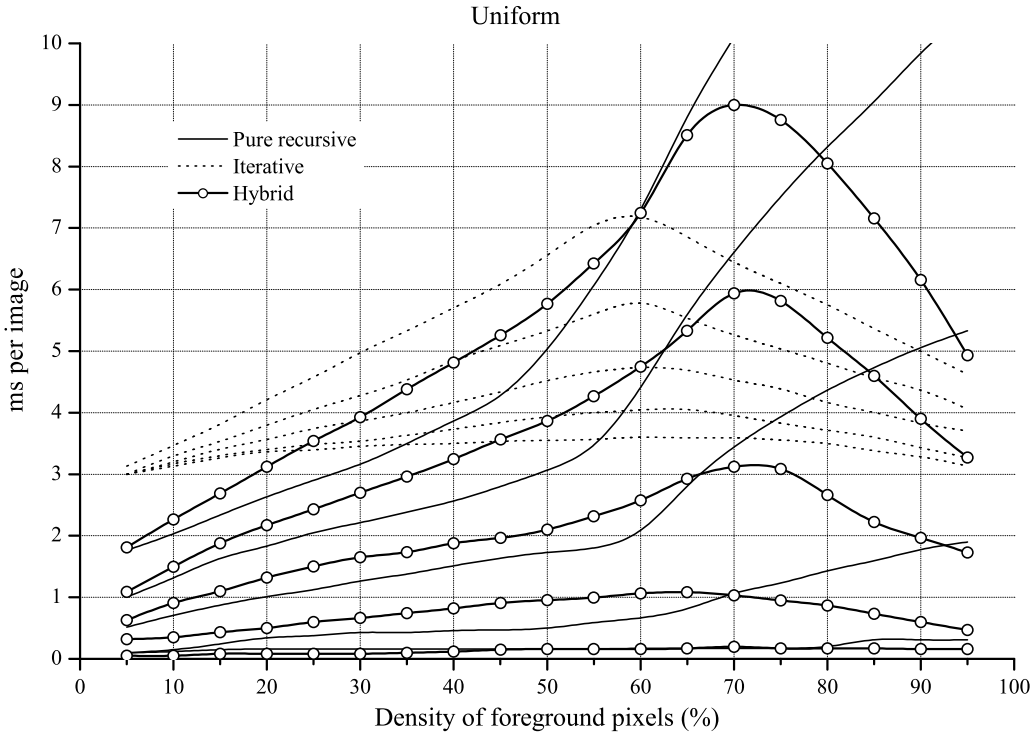


Figure 2.15. *Average time per image consumed in the labeling of several Uniform sets with different image sizes (*100×100 *to* 500×500*, in steps of* 100*). The lower the curve the smaller the images.*

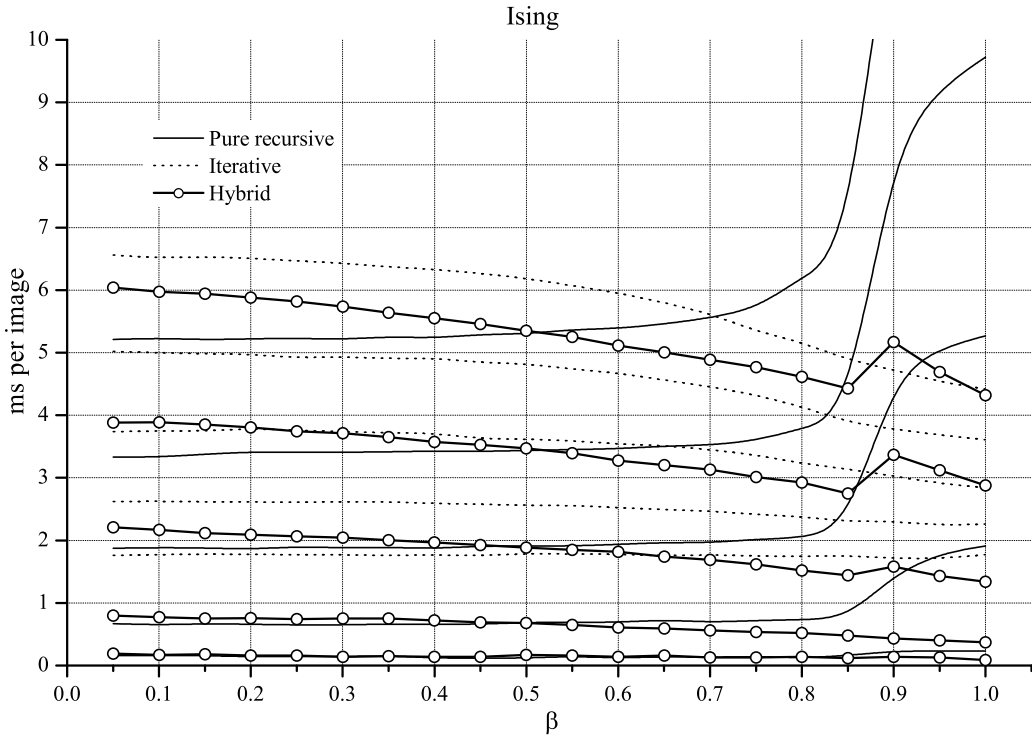


Figure 2.16. *Average time per image consumed in the labeling of several Ising sets with different image sizes (*100×100 *to* 500×500*, in steps of* 100*). The lower the curve the smaller the images.*

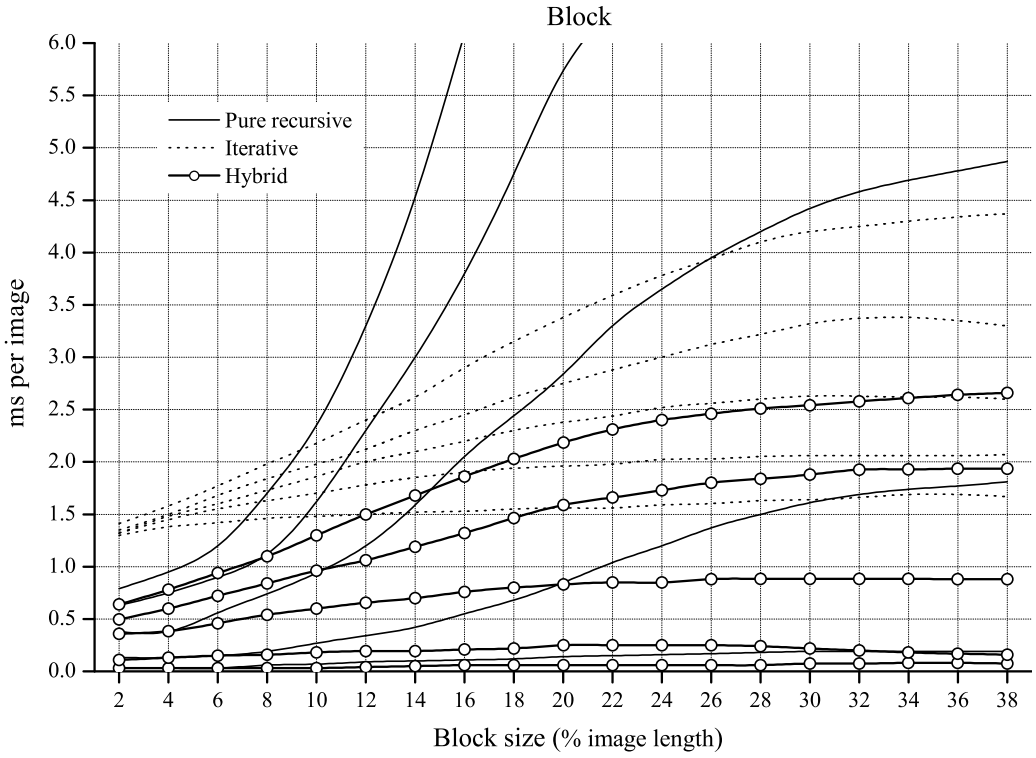


Figure 2.17. *Average time per image consumed in the labeling of several Block sets with different image sizes (*100×100 *to* 500×500*, in steps of* 100*). The lower the curve the smaller the images. Block size is expressed as a percentage of image length.*

2.3.5 Performance detecting spanning objects

I also tested the performance for spanning object detection, Algorithm 6 in section 2.2, on the three image sets. Naturally, both recursive and hybrid labelling are clearly faster than two pass iterative labelling for detecting the presence of spanning objects: The iterative technique requires as much time as for labelling the entire image plus the label matching search on the two opposite image borders after the labelling, while the recursive algorithms only need to label, at most, the objects touching one of the borders of the image, as explained at the end of section 2.2.

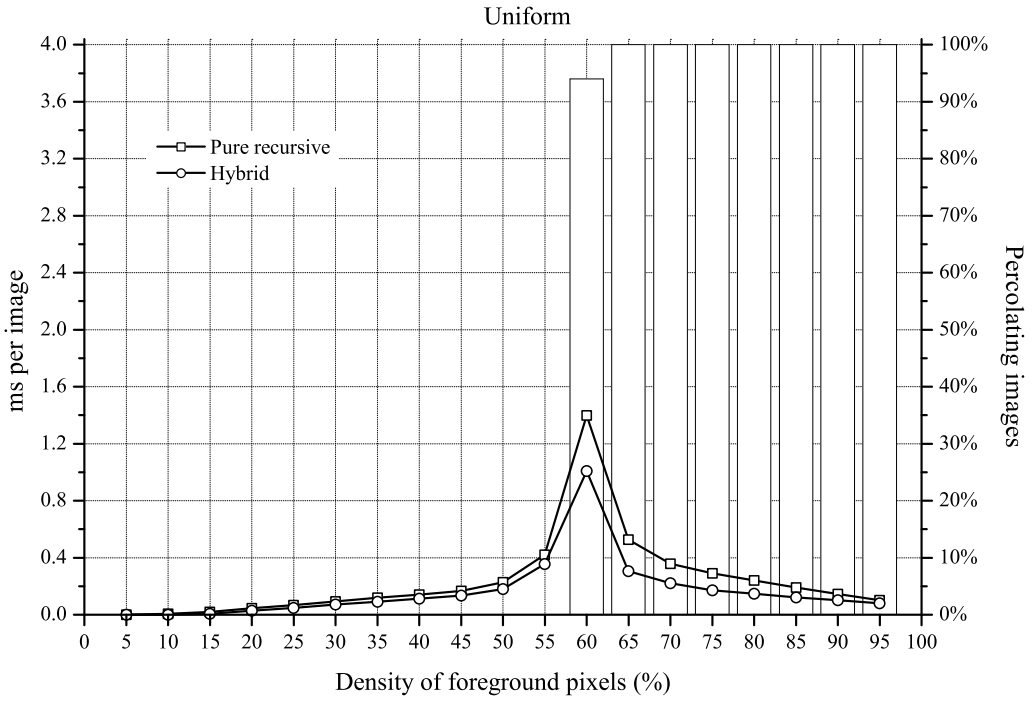


Figure 2.18. *Average time per image consumed in the detection of spanning objects in the Uniform set. The bars show percolation data: Percentage of images containing at least one spanning object.*

Graphs in this section show percolation data (the percentage of images with at least one spanning object) and the time used by the pure recursive and the hybrid percolation detectors in each of the three image sets. Iterative time is not shown because of 1) the exaggerated differences in magnitude,

and 2) iterative time is the same as that for labelling already plotted in Figs. 2.12, 2.13, 2.14 with a small overhead due to the final label matching search.

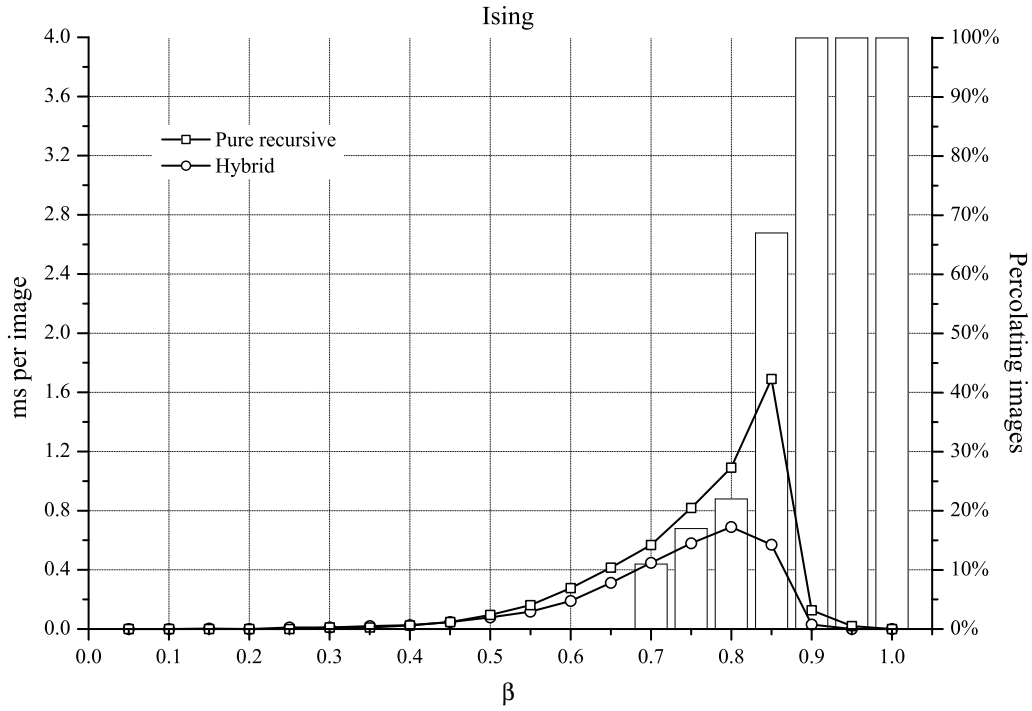


Figure 2.19. *Average time per image consumed in the detection of spanning objects in the Ising set. The bars show percolation data: Percentage of images containing at least one spanning object.*

On the Uniform set, Figure 2.18, below the percolation threshold, p_c, clearly shown by the bars in Figure 2.2, objects are small and very little work has to be done —negligible time when compared to iterative labelling. Well above p_c, when every image percolates, direct paths across the image are very likely, and the job can again be done very fast. Then hybrid labelling is clearly faster than classical recursive labelling, because one or a few bursts will bring it to the opposite border, while the recursive technique needs a recursive call for each step along the way. Around the percolation threshold, only one, quite ramified spanning cluster is usually present, such that a straight path across the image is less likely, and more pixels have to be labelled before getting to the opposite border. Here hybrid labelling needed 1 ms per image, recursive labelling 1.4 ms, and iterative labelling 7.6 ms. Overall, hybrid labelling

needed 3% of the time required by iterative labelling and 70% of the classical recursive labelling. This is a significant improvement when dealing with large images, a large number of images, or real time applications, to be added to the improved stack use.

The time required for the recursive percolation check on the Ising images, Figure 2.19, was less than 0.02 ms per image for β outside $[0.45, 0.95]$, where the iterative technique required more than 6 ms. In the region of the phase transition, Figure 2.4, which corresponds also to the percolation threshold, the recursive and the hybrid techniques took up a maximum of 1.69 and 0.69 ms per image, respectively, while the iterative technique required 4.77 ms. On average, hybrid percolation check required 60% of the time used by the pure recursive technique, and 2.8% of the time required by the iterative technique.

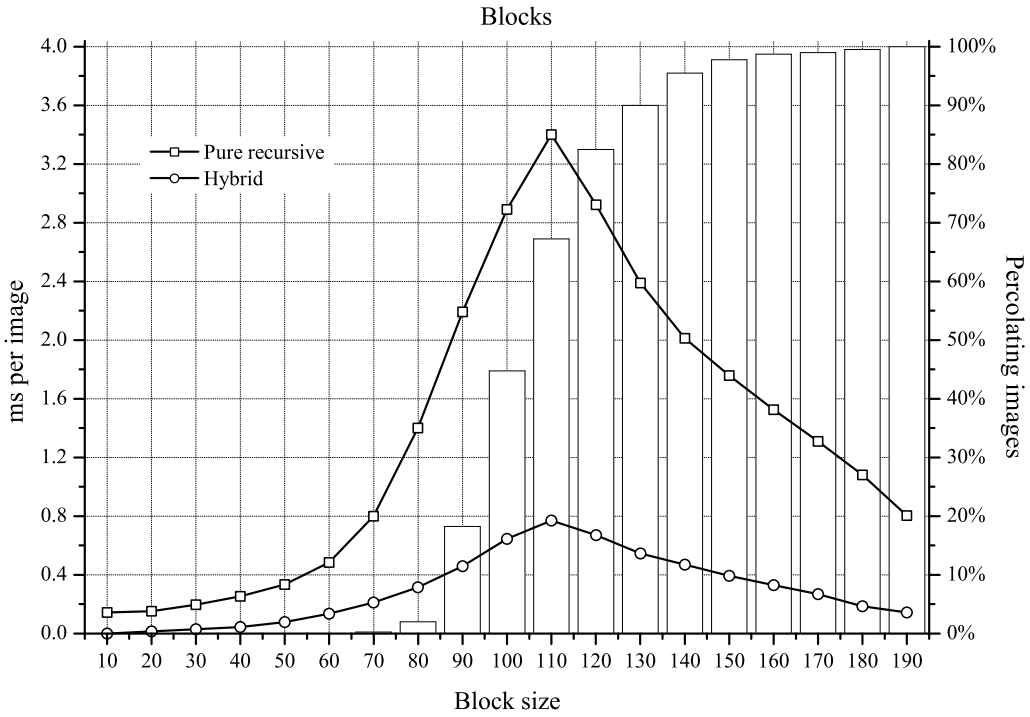


Figure 2.20. *Average time per image consumed in the detection of spanning objects in the Block set. The bars show percolation data: Percentage of images containing at least one spanning object.*

Regarding the Block set, Figure 2.20, for the smaller blocks, with low

densities of foreground pixels (figure 2.9) and low number of small objects (figure 2.6), recursive and hybrid labelling consumed negligible time. When bigger objects and some probability of percolation appears, hybrid labelling moves away from recursive labelling, consuming at the point of worst performance (blocks of size 110×110) just 23% of the time needed by recursive labelling. Overall, hybrid percolation detecion averaged 22% with respect to recursive percolation detection and 9% with respect to iterative labelling in the Block set. Tables 2.4 and 2.5 summarize the results. The data for the iterative technique is that of the labelling of the entire images. For percolation detection a small overhead for the matching of labels between the borders of the image should be taken into account.

	Uniform	**Ising**	**Block**
Average ms/image			
Recursive	0.24	0.27	1.44
Iterative	5.79	5.75	3.42
Hybrid	0.17	0.16	0.32
Max. ms/image			
Recursive	1.39	1.69	3.40
Iterative	7.65	6.53	4.46
Hybrid	1.01	0.69	0.77

Table 2.4. *Summary of the performance data for detection of spanning objects with the three* 512×512 *image sets. Iterative data for the labelling of the images has been included for reference.*

Hybrid	**Uniform**	**Ising**	**Block**
Average time			
Of iterative	2.9%	2.8%	9.4%
Of recursive	70.8%	59.3%	22.2%
Max. time			
Of iterative	13.2%	10.6%	17.3%
Of recursive	72.7%	40.8%	22.6%

Table 2.5. *Relative performance of hybrid detection of spanning objects with the three* 512×512 *image sets. Iterative data is only for the labelling of the images, label matching excluded.*

2.3.6 One dimensional arrays for image storage

So far I assumed a image stored in a bidimensional array of integer values, $p_{x,y}$, such that two indexes, corresponding to column and row, located each pixel. However, in many applications images are acquired through frame grabbers, which frequently use one dimensional arrays to store the frames. One dimensional arrays may also be favoured by the programming language, or the final purposes of the processing in a given application. When a $W \times H$ image is stored in a one dimensional array p_n, pixel (x, y) is indexed by a single index, $n = y \times W + x$. Two sums (indirections) are implicit in the notation $p_{x,y}$: p_y is an array of pointers pointing to each row, such that y has to be added to the base pointer p for it to point to the required row, and, then, x has to be added to the resulting pointer to point to the required line into the row. With the p_n notation, one of the sums becomes explicit, allowing the programmer to perform it only when he wants to. The y sum is only necessary to change the row, and it is not needed while working on pixels along the same row. With the $p_{x,y}$ notation, on the contrary, even when accessing consecutive pixels on the same row, two sums are performed in each access to every pixel, due to the double indirection.

While this may sound far too obvious to the experimented programmer, it has special relevance when dealing with the hybrid technique. The hybrid technique favours repeated moves back and forth along the same row. It is clearly suitable for using one dimensional arrays, and thus be directly applied to the native format of frame grabbers. Row changes only during the second forward scan, and can be done by adding or subtracting the row length W to or from the current pixel index. The final result is that as many sums are eliminated from the backscan and from the forward scan as foreground pixels are in the image, and a single loop suffices to scan the whole image, see Algorithm 7. Therefore, hybrid labelling on one dimensional arrays is faster because it needs less operations. It needed 13% less time for the labelling of the test sets than the 2D array version. But there is yet an additional advantage not related to speed: A single index to address the pixels also means a single argument in the recursive function, and this means that the

same recursive calls fit into a smaller stack, because one less variable — usually 4 bytes— is stored per call. Thus, one dimensional hybrid labelling is faster and relaxes even further the pressure on the stack.

Algorithm 7 2D hybrid labelling for 4-neighbourhoods in 1D image arrays

```
0 → label
for all n if p_n < 0 {increase label, Label(n)}

Label(n)
    while p_{n-1} < 0 decrease n
    n → m
    while p_m < 0 {label → p_m, increase m}
    while n < m
        if p_{n-W} < 0 Label(n − W)
        if p_{n+W} < 0 Label(n + W)
        increase n
end of Label
```

2.3.7 Hybridizing the Wolff cluster algorithm and the Swendsen-Wang algorithm

Hybrid labelling is not only a valid alternative for the labelling of images *per se*, it can also substitute any of the classical labelling algorithms at the core of any higher level algorithm using labelling techniques. To show this, in the following I will describe the "hybridization" of two well known Monte Carlo algorithms to sample the Ising [115] and Potts [200] models, and how it introduces the aforementioned advantages of the hybrid technique into the parent methods.

In the classical statistical mechanics interpretation of the Ising and Potts models, individuals in a lattice, called *sites*, can be in a series of Q different *states*, $Q = 2$ in the Ising model, $Q \geq 2$ in the Potts model, which is a generalization of the Ising model. The probability of a site to be in a given state depends on the states of its neighbours. Due to the applications to the theory of ferromagnetism, states are usually referred as *spins*, and connected sites with the same spin form *clusters*, but we will use here the terms "level", "pixel", and "component" for the usual "spin", "site", and "cluster", because our major interest in the Ising and Potts models is synthetic image generation. The Ising model allows us to generate binary images with known statistical properties —such as those in the previous section, see Figure 2.3 in 2.3.2. Potts models colour images, with Q different colours, see Figure 2.21.

Wolff's

Wolff's cluster algorithm [271] generates samples of the Potts model using the classical recursive technique to change the level of *part* of a component. This is achieved by introducing a probability into the decision of entering —issuing the corresponding recursive call— a neighbouring pixel with the same level. The major inconvenient of the algorithm is the possibility of stack overflow when big components and high probabilities are involved. Therefore, substituting the pure recursive technique by the hybrid technique should improve

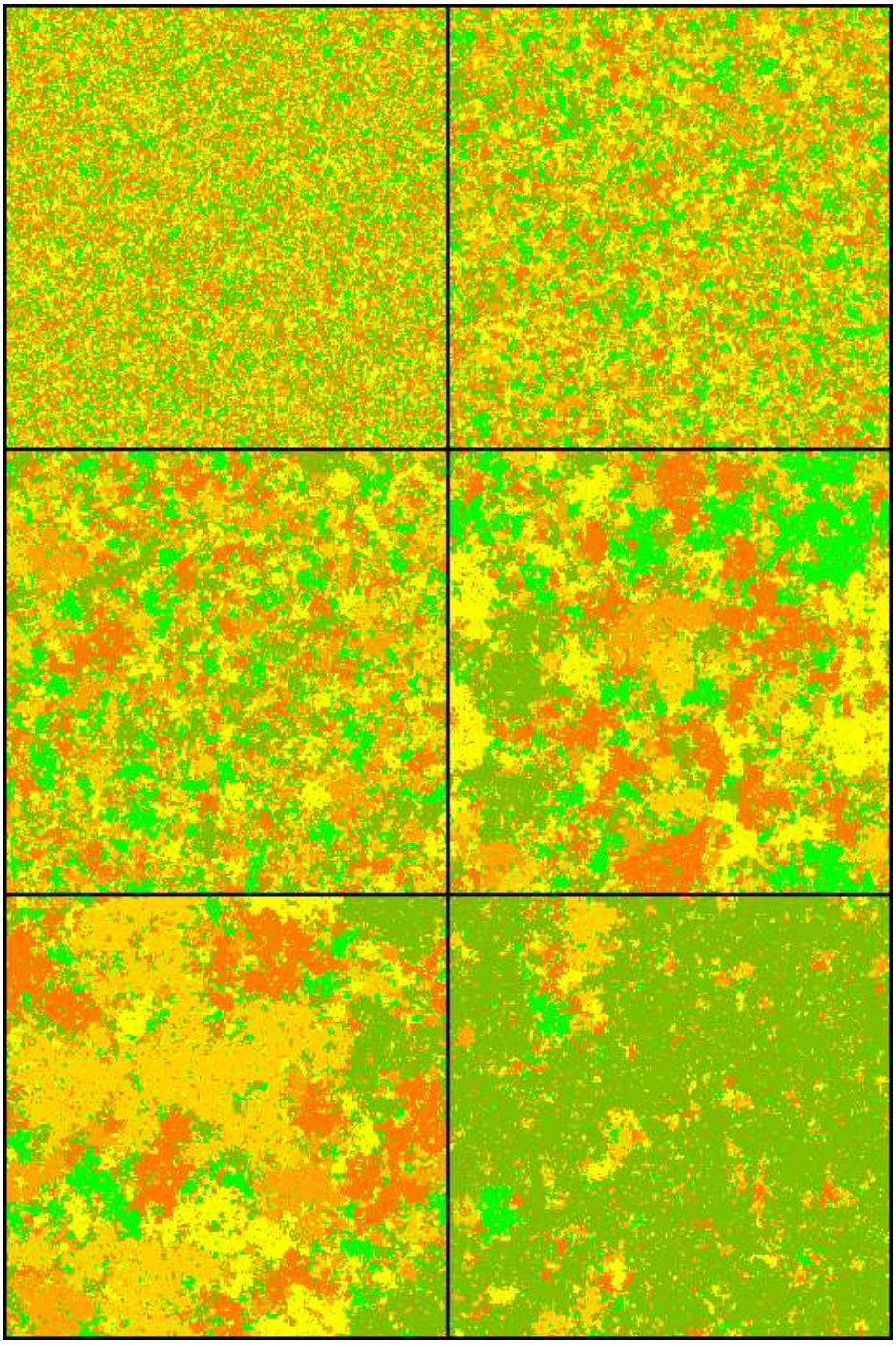


Figure 2.21. *Some samples of the Potts model with six colours ($Q = 6$) and different values of β.*

the algorithm. We have just seen that the hybrid technique is also faster than the pure recursive technique, an additional argument for introducing the hybrid approach into the Wolff algorithm, taking into account that this is a Monte Carlo technique, therefore in need of several iterations to achieve the desired state. Periodic boundaries, i.e. pixels in a border of the image are neighbours of the pixels in the opposite border, are usually assumed in Potts models to minimise edge effects. This, together with the probability check —the typical "coin toss" of Monte Carlo simulations— are the only modifications required to include the hybrid algorithm in Wolff's. Periodic boundaries are obtained by replacing the boundary checks needed every time a site coordinate is incremented or decremented (not explicit in the pseudocode in section 2.2) by the appropriate conditional increments or decrements. Probability checks are ANDed to every check of the level of a pixel, see Algorithm 8.

Algorithm 8 2D hybrid Wolff's

for all n **initialize** p_n
repeat MC times
    select random n
    $p_n \rightarrow s$
    Flip(n)

Flip(n)
  while ($p_{n-1} = s$ and Rand(2) $< P$) **decrease** n
  $n \rightarrow m$
  while ($p_m = s$ and Rand(2) $< P$) $\{\tilde{s} \rightarrow p_m$, **increase** $m\}$
  while $n < m$
    if ($p_{n-W} = s$ and Rand(2) $< P$) Flip($n - W$)
    if ($p_{n+W} = s$ and Rand(2) $< P$) Flip($n + W$)
    increase n
end of Flip

I introduced hybrid labelling with these modifications into Wolff's algorithm, and obtained a 25% time decrease with respect to the original version in the generation of Ising samples of size 512×512. The considerations of

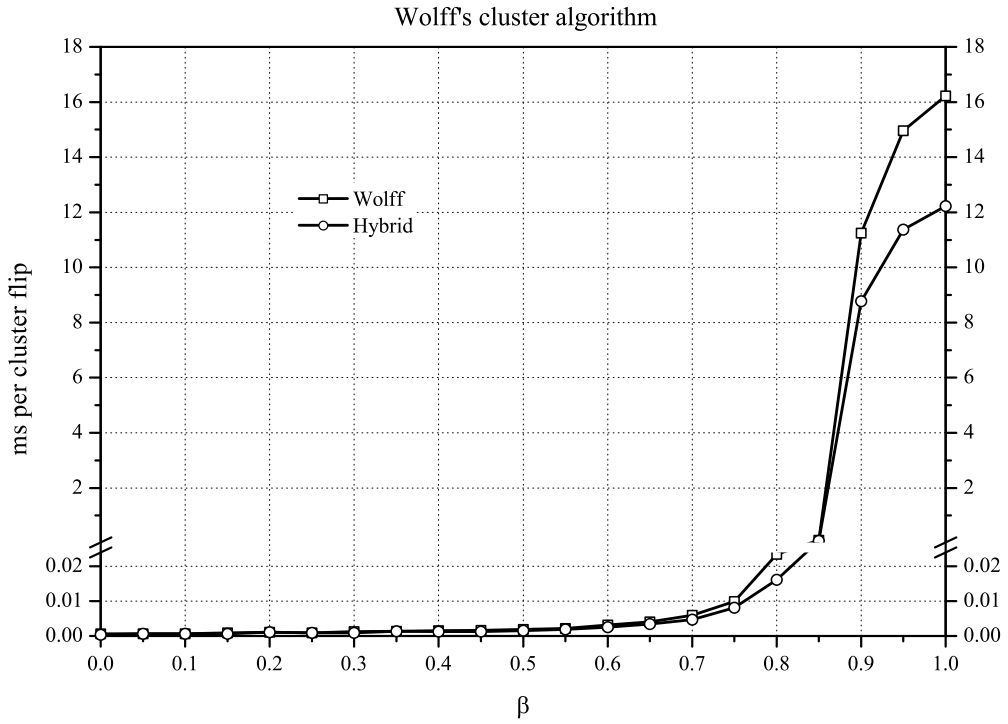


Figure 2.22. *Average time per component flip with original Wolff and hybrid Wolff in the generation of Ising ($Q = 2$) samples of size* 512×512 *for a range of values of* β*.*

section 2.3 regarding the decrease in the risk of a stack overflow in the Ising set, or the capability to generate much bigger samples with the same stack size apply exactly the same. Figure 2.22 details the results.

Swendsen-Wang

The Swendsen-Wang algorithm uses another approach to Ising simulations, which involves the concept of bond connectivity. Bonds are located between pixels sharing the same level with a given probability (function of $\beta = J/kBT$). Pixels connected by bonds form components. Then, the level of the components defined by the bonds is randomly updated and the procedure is repeated for the new configuration. The implementation of Wang [259] uses the Hoshen-Kopelman (HK) algorithm to identify the components and update their level. To replace HK with the hybrid approach, this has to be

modified to work with bond connectivity instead of pixel connectivity, and then the entire Swendsen-Wang has to be adapted to work with the recursive technique. In Wang's implementation, first all components in the image are labelled with HK, then the equivalence table is rearranged (conflicting labels "translated" to their corresponding "canonical" or "proper" labels, in usual notation), and then the labels in the equivalence table are randomly assigned the new levels, such that, last, a scan over the image assigning these "labels" to the pixels performs the component update. With the hybrid labelling technique, randomly assigning one of the permitted levels to the current label previous to labelling each component suffices to perform cluster identification —and characterization, if necessary— and update at the same stroke.

Algorithm 9 2D hybrid Swendsen-Wang

for all n **initialize** p_n
repeat MC times
    for all n
        $(p_n = p_{n+1})$ and $(\text{Rand}(2) < P) \rightarrow h_n$
        $(p_n = p_{n+W})$ and $(\text{Rand}(2) < P) \rightarrow v_n$
    for all n **if** (h_n or v_n) $\{\lfloor \text{Rand}(Q) \rfloor \rightarrow \mathit{label}, \text{Label}(n)\}$

Label(n)
    $n \rightarrow m$
    while $h_{n-1} < 0$ {**decrease** n, $\mathit{label} \rightarrow p_n$, $0 \rightarrow h_n$}
    $\mathit{label} \rightarrow p_m$
    while $h_m < 0$ {**increase** m, $\mathit{label} \rightarrow p_m$, $0 \rightarrow h_m$}
    while $n < m$
        if v_n $\{0 \rightarrow v_n, \text{Label}(n + W)\}$
        if v_{n-W} $\{0 \rightarrow v_{n-W}, \text{Label}(n - W)\}$
        increase n
end of Label

To work with bond connectivity, first an array of bonds has to be used to store the bonds, and then the pixel checks have to be replaced with bond checks. Also the bond which takes us to a pixel has to be removed after entering the pixel, or the labelling will endlessly recur to the same pixels.

The pseudocode in Algorithm 9 is an implementation of Swendsen-Wang with hybrid labelling.

Decrements and increments of the location index are of course conditioned by the usual periodic boundary assumption, but this is not explicit in the pseudocode in Algorithm 9 for clarity. The image is stored in a one dimensional array p_n, and two auxiliary boolean arrays, h_n and v_n, are used to store the horizontal and vertical bonds. Rand(sup) is a pseudorandom generator for a uniform distribution in $[0, sup)$. W is the length of a row, Q the number of levels or colours, P the bond probability, and MC the number of Monte Carlo steps.

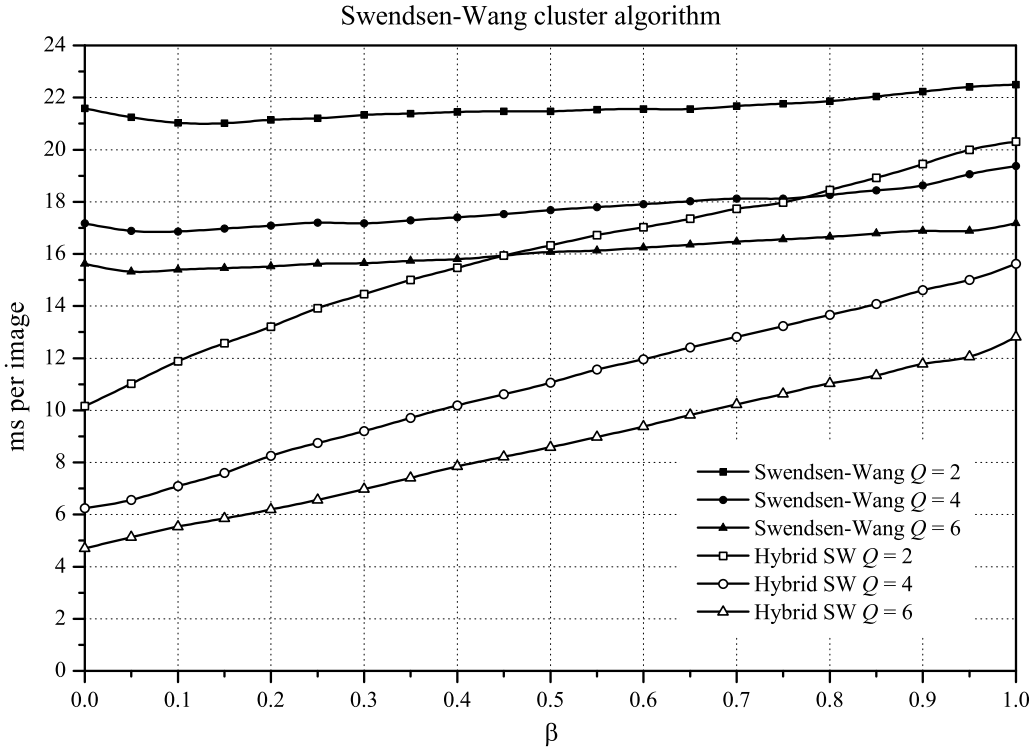


Figure 2.23. *Average time per Monte Carlo step after convergence with Wang's Swendsen-Wang and hybrid Swendsen-Wang for* $Q = 2, 4, 6$*, and* β *from* 0.00 *to* 1.00*.*

I ran Wang's implementation of the Swendsen-Wang algorithm particularized for 2D images and my hybrid implementation for Potts with 2, 4, and 6 colours, and the hybrid version took just an average 63% of the time

consumed by Wang's (73% with 2 colours, 61% with 4 colours, and 53% with 6 colours). Figure 2.23 details the results. Wang's Swendsen-Wang has a flatter behaviour along β, but hybrid Swendsen-Wang performs significantly faster. Besides, component characterization and percolation check are far simpler with the hybrid version than with the original HK Swendsen-Wang.

2.4 Performance in real world computer vision applications

Up to now I have analyzed the behaviour and performance of the hybrid technique in neutral scenarios, with promising results. However, it is on real images in real applications where a technique ultimately demonstrates its suitability and convenience. In the following, I will describe the performance of hybrid labelling in two real computer vision applications, one in 2D and the other in 3D.

2.4.1 A real-time 2D computer vision application

I implemented hybrid labelling on a real-time machine vision system for the quality inspection of canned tuna [157, 158]. The system inspects tuna cans right before the addition of sauce or oil previous to closure, at a high can rate. It checks the geometry and edge of the can, and its contents, regarding the quality of the meat and its appearance, and the presence of holes, strange bodies, blood stains, and bruised tissue. The whole procedure from acquisition to decision taking and subsequent action is performed by a single computer, and thus the application is algorithmically very demanding and the time constraints very tight.

The system achieves an inspection rate of over 1 000 cans per minute for RO-100 cans (round cans 7 cm in diameter), which implies less than 60 ms per can for the entire process: Can detection, bus transfer, image processing (can border extraction and parameterization, contents segmentation, and analysis), acceptance/rejection decision, and, if necessary, activation of the rejection system. The system's performance on a iPentium IV 2.4 GHz MS-Windows XP based PC is at the time of writing below 50 ms per can. Single production line rates of 1 000 cans per minute are well above the usual current rate in the canned tuna industry. The production lines of the canned fish factories that collaborated in the development of the system have typical work rates in the order of 300 cans per minute per line, quite high considering that

the current quality assurance is entirely manual[6].

Object labelling is required within the system for the characterization and classification of dark features in the meat —holes, blood stains, bruised tissue—, Figure 2.24, after segmentation of the histogram into three classes: Light meat, dark meat, and dark features.

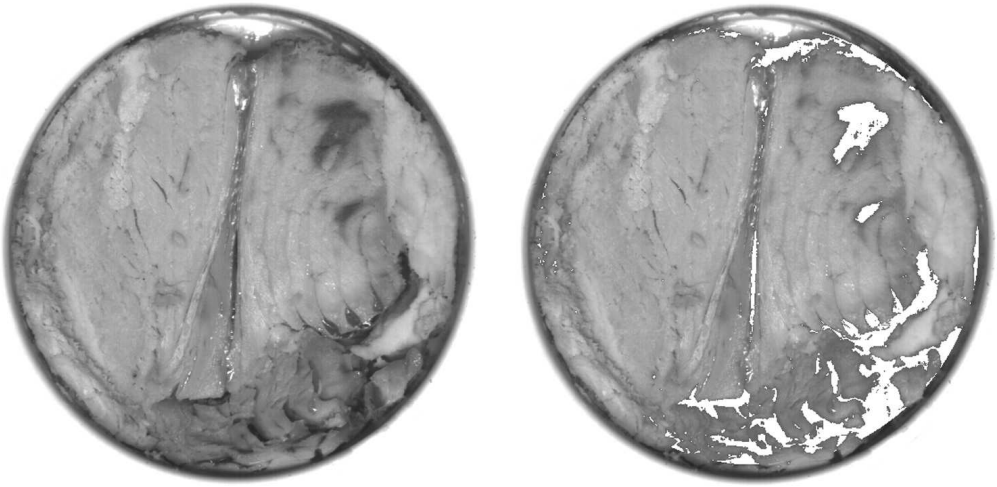


Figure 2.24. *Typical dark features in canned tuna (left) and the corresponding mask for* $\mathcal{A} = \{$*gray level in the "dark feature" class*$\}$ *(right).*

After defining the region of interest (ROI) within the image, extraction and analysis of the border of the can, and analysis of the global characteristics of the contents, if the meat structure is satisfactory, we segment the ROI's gray level histogram using an adaptation of the Expectation-Maximization (EM) maximum likelihood algorithm [68, 102], such that three major types of features can be distinguished: 1) light tuna meat, 2) dark tuna meat, and 3) holes, crevices, blood stains, and bruises (dark features), Figure 2.25. The EM algorithm allows the statistical parameterization of each of the three features, by modelling the histogram as a mixture of three Gaussian distributions characterized by their corresponding mean —average gray level of the corresponding feature—, standard deviation, and weight —proportion of the ROI corresponding to each feature. From them we can draw some conclusions

[6] And thus, in view of the production rates, clearly insufficient.

about the quality of the tuna meat in the can, specifically about the relative proportion of good quality tuna —light meat— and poor quality tuna —dark meat—, and the quantity of holes and bruises. It also gives us an adaptive gray level threshold for the extraction of holes, crevices, blood stains, and bruises, as the maximum likelihood (ML) threshold between the two lower mean Gaussians. A fixed threshold would require very stable lighting conditions, usually a risky assumption in a real-world industrial application, and would assume that the same kind of tuna is processed all the time, what is generally not true. In the thresholded ROI we can distinguish and measure the extent and number of the blood stains and bruises via a specifically designed algorithm requiring the previous delimitation and characterization of the morphology of the dark features. Blood stains and bruises do not pose a risk for the health of the consumer, but they cause a negative impact in the appearance of the product and may affect its organoleptic properties.

First, we assume that the histogram of our image is composed of a mixture of Gaussian distributions, each corresponding to each type of feature: A taller Gaussian distribution, with a higher mean, for healthy light meat; a lesser mean, for the darker healthy meat, and the smaller Gaussian corresponding to holes, crevices, and bruises. The EM algorithm particularized for one dimensional discrete data is able to distinguish the three Gaussians conforming the histogram, and to estimate their relative weights, means, and variances. The EM formulation for separation of three modes in a greyscale histogram $H(l)$ is as follows [157].

We want to model our histogram $H(l)$ by a PDF $p(l)$ such that

$$p(l) = \sum_{i=1}^{3} w_i \Psi_i(l, \mu_i, \sigma_i) = \sum_{i=1}^{3} \Omega_i(l), \quad l \in \{0, 255\}, \tag{2.2}$$

so that we may use the EM algorithm with the update equations

$$f^n(\Omega_i|l) = \frac{\Omega_i^n(l)}{p^n(l)}, \tag{2.3}$$

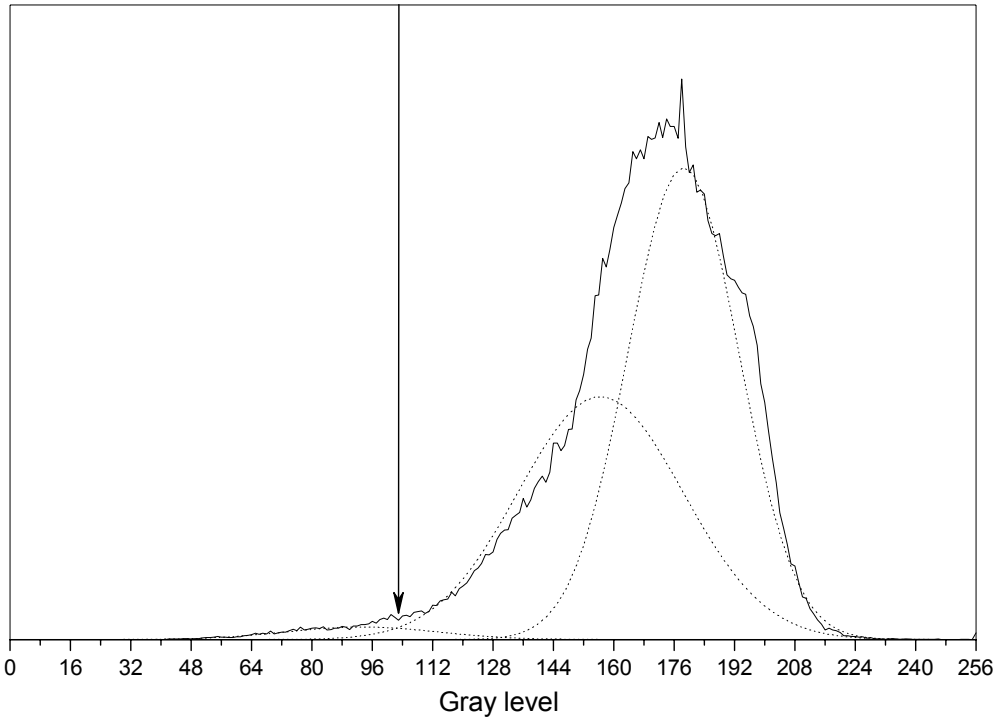


Figure 2.25. *EM segmentation of the histogram of the contents of a tuna can in three classes: "Light meat", "dark meat", and "dark features". The ML threshold for the "dark features" class is shown by the vertical arrow.*

$$w_i^{n+1} = \frac{\sum_{l=0}^{255} H(l) \cdot f^n(\Omega_i|l)}{\sum_{l=0}^{255} H(l)}, \tag{2.4}$$

$$\mu_i^{n+1} = \frac{\sum_{l=0}^{255} H(l) \cdot l \cdot f^n(\Omega_i|l)}{\sum_{l=0}^{255} H(l) \cdot w_i^{n+1}}, \tag{2.5}$$

and

$$\left(\sigma_i^{n+1}\right)^2 = \frac{\sum_{l=0}^{255} H(l) \cdot \left(l - \mu_i^{n+1}\right)^2 \cdot f^n(\Omega_i|l)}{\sum_{l=0}^{255} H(l) \cdot w_i^{n+1}}, \tag{2.6}$$

where $n = 0, \ldots$ stands for iteration. The three Gaussians modelling the histogram are ordered such that $\mu_1 < \mu_2 < \mu_3$ and then the ML threshold for

the lower Gaussian provides the condition $\mathcal{A} \equiv \{p_{x,y} \leq l_d \mid \Omega_1(l_d) = \Omega_2(l_d)\}$ for the labelling. Convergence to a good enough solution for the problem at hand —locating the darker mode such that a threshold can be deduced to extract holes and bruises— is usually achieved in a small number of iterations. In spite of its computational overhead due to the fact that floating point arithmetic is involved, average execution time per can is below 3 ms.

The strong time constraints discard two-pass iterative labelling: At any stage of the analysis, as soon as a single defect above tolerance is detected, the can is rejected and focus moved to the next can. Labelling every dark spot in the sample prior to analysis is an unacceptable waste of time. The analysis of the dark features requires characterization through measuring the extent and shape of the dark feature under analysis, and computing the bounding box. Additionally, under certain modes of operation of the application, the dark features causing rejection, among all dark features, have to be emphasized in the image. All these are typical requirements for recursive labelling.

In order to achieve the robustness required for an industrial quality assurance system, almost anything has to be expected to appear in a can. And this includes cans whose entire content is a dark feature. Therefore, the labelling algorithm has to be able to cope with big objects without risking a stack overflow —the system uses 1 Mpixel frames from a 1 024 pixel line scan camera. Thus, the alternative to iterative labelling, classical recursive labelling, was not entirely suitable. It would comply with the requirements under normal operation, but would be a potential source of trouble under exceptional, but not impossible, conditions. Thus, for instance, due to the specific illumination method, an empty can would suffice to trigger a stack overflow. Safeguard code can be added to check for the presence of big objects in order to prevent stack overflows, by halting the labelling in case a given upper bound in surface area or recursion depth is reached, but 1) it is an avoidable overhead in a demanding, optimized to the μs, real-time application, and 2) it is an emergency exit, a *patch*, not a solution. The solution is hybrid labelling.

Image frames are 1024×1024. Tuna images, or rather, dark features in

tuna images, show considerable degree of spatial aggregation when compared to the Uniform images (see the mask in Figure 2.24). In the limit, the dark feature caused by an empty can resembles a highly compact object as those in the Block set at the higher densities. According to the graphs in Figures 2.10 and 2.11, and the subsequent analysis in 2.3.3, a stack overflow is not possible with hybrid labelling and 1024×1024 images, even in the unlikely event of a can filled with "uniform random" tuna.

Hybrid labelling not only solved the stack overflow problem, thus improving the robustness of the system, but also speeded up the procedure. I performed several tests during the in-line testing of a prototype in a real factory. I measured the maximum recursion depth and the average time required both by the classical recursive technique and the hybrid technique for the labelling of dark features in 10 runs of 1 000 cans each. During the tests, all dark features in each can were labelled and characterized, an average 232 dark features per can, with an average surface area of 36 pixels.

Classical recursive labelling averaged 322 ± 100 μs per can for the labelling and characterization (area, perimeter, and bounding box) while hybrid labelling averaged 158 ± 47 μs per can, 49% of the average time required by classical recursive labelling. The average saving in computing time was 164 μs per can, with a maximum saving of 343 μs. Maximum recursion depth for the classical technique was 3 033 consecutive recursive calls —no exceptional cans appeared during the tests—, while for hybrid labelling it was 255 consecutive recursive calls, 8%.

A closer look to the performance data reveals that the best relative performance of hybrid labelling was 36.6% of the time used by the classical technique, while the worst was 61.2%. Hybrid recursion depth was at its best with 4% of the classical technique recursion depth, and at its worst with 16%, averaging 9%. With respect to the extreme values, the aforementioned hybrid and classical maximum recursion depths where achieved on the same can. The minimum recursion depth of the hybrid algorithm was 66, while on the same can the classical technique recorded 1 693. Vice versa, the classical algorithm used a minimum recursion depth of 588 consecutive recursive calls

where the hybrid algorithm recorded 71. Table 2.6 summarizes the results.

	Classical	**Hybrid**	
Speed			
μs per can	322 ± 100	158 ± 47	49%
Recursion depth			
Average	$1\,392 \pm 642$	125 ± 50	9%
Maximum	3 033	255	8%

Table 2.6. *Summary of benchmarking data in a real time machine vision application for the quality assurance of canned tuna.*

2.4.2 A real world 3D computer vision application

I used hybrid labelling for the extraction of fibre bundles in a microtomography [163] of a Carbon-Carbon composite (C/C) [41, 42], made of carbon fibres embedded in a continuous matrix of carbon using chemical vapour infiltration (CVI) [20, 21, 41, 254]. The fact of both matrix and fibres being fabricated from carbon produces a unique combination of properties, including very low thermal expansion coefficients and high thermal conductivity, retaining their mechanical properties at high temperatures ($> 2000^\circ$ C, in nonoxidizing atmospheres), high specific strength, excellent resistance to abrasion, high resistance to thermal shock, very high elastic modulus, low-weight, high electrical conductivity, low hygroscopicity, nonbrittle failure, resistance to biological rejection and chemical corrosion, and reasonable machinability [57].

These properties vary depending on the fibre fraction and type, textile weave type, and the individual properties of the fibres and matrix materials. The matrix precursor material and the manufacturing method have a significant impact on composite strength. Strong composites require sufficient and uniform densification. Traditional fabrication techniques may chemically or mechanically damage the fibres and cause high porosity leading to poor mechanical properties. Hence the popularity of CVI, the leading fabrication process for fibre reinforced composites. CVI consists in subjecting a fibrous preform to chemical vapour infiltration of relatively high vapor pressure gases, called precursors, gaseous compounds of the matrix material that under suitable operating conditions, with comparatively low stress and temperature, deposit the desired matrix phase on the surface of the preform, until complete densification is achieved. CVI is a variant of Chemical Vapour Deposition (CVD), a method to build dense structural parts or coatings on a substrate surface, by transporting gaseous compounds to the substrate surface where deposition occurs. CVD and CVI use similar equipment and the same precursors, but CVD implies deposition onto a substrate surface, whereas CVI implies deposition within a porous body [260].

Fabrication of a C/C composite by CVI is as follows (see Figure 2.26).

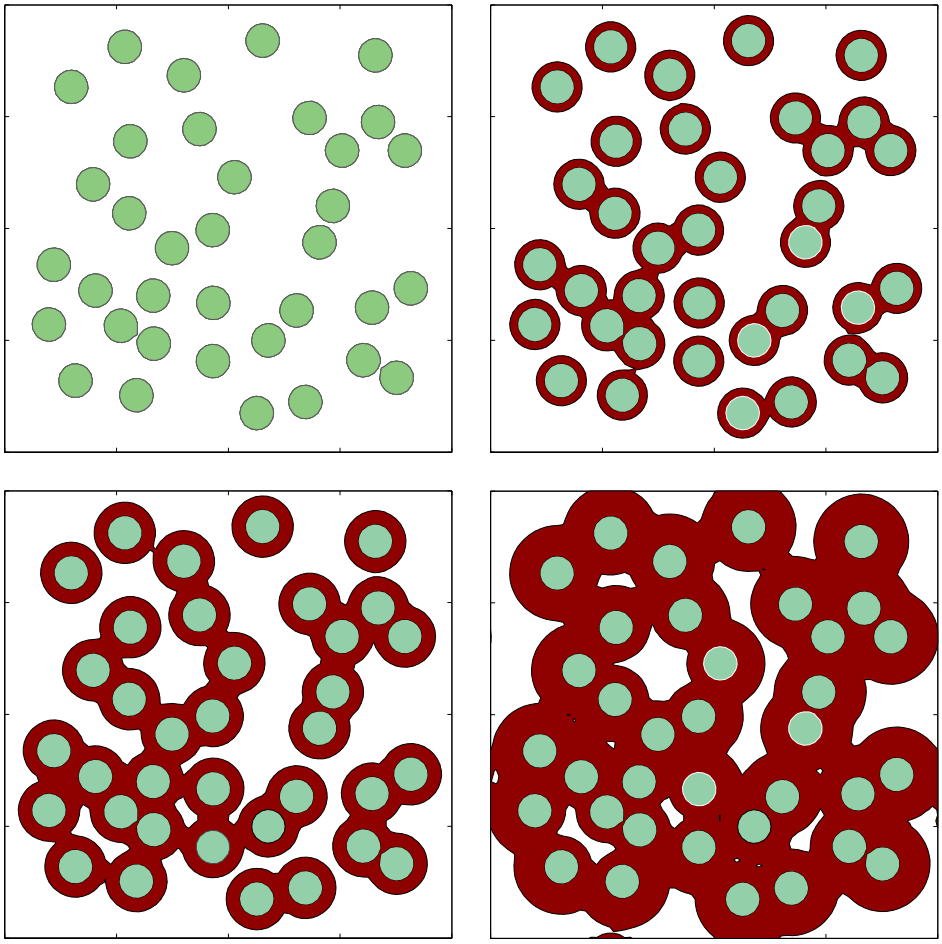


Figure 2.26. *Chemical Vapour Infiltration. The light circles represent the fibres in cross section, the matrix is shown in dark colour. Note the formation of cavities. (Adapted from [119]).*

The preform, consisting of a network of fibres of carbon in a porous arrangement, is placed in a high temperature furnace. A vapour precursor of matrix material flows into the furnace penetrating the pores of the preform. It dissociates at the fibre surfaces and deposits carbon as a solid coating onto it, "growing" the fibres and filling the inter-fibre spaces. The more the fibres grow, the smaller the space between them. The corridors allowing the precursor flow become narrower and begin to close, forming cavities inaccessible

to the gas. Deposition on the surfaces inside inaccessible cavities stops. In the end the external surface of the preform is completely sealed, preventing further densification of the interior and ending the process. Percolation theory plays a major role here, the lowest porosity that permits percolation of a fluid through the porous space is the percolation threshold of the porous solid. This is a key factor in CVI as it determines to a great extent the final microstructure of the composite and hence the mechanical properties of the material.

Several methods have been developed to model the CVI process [97, 118–120, 204, 205, 230, 232, 260, 276]. Reliable models can help to optimize the fabrication process minimizing the number and extent of cavities, decreasing production costs and increasing production quality. Early models tended to avoid introducing the detailed geometry of the microstructure because its complexity prevented the use of analytical formulations, a common problem when dealing with composites (see 1.2.1). Latter models, introducing elements from percolation theory and effective medium theory, already take into account the geometry of the problem, but usually the preform is modelled as a regular array of elongated cylinders. Efforts have also been made to derive macroscopic properties from the microstructure of CMC materials, to establish relationships between their microstructure and measurable macroscopic parameters. A common approach to simulate the local microstructure is the Expanding Overlapping Circle model [206], which simulates the microstructure of a cross section of an uniaxial bundle of fibres by a computer generated digital image of a set of expanding overlapping circles. Pore structure and surface area are modeled using an analytical expression for cylindrical fibres by directly utilizing geometric properties of circles and cylinders. This is the underlying framework of many models for the study of properties of composites, such as thermal conductivity and gas permeability [229], effective diffusivity [245], or reaction and diffusion kinetics [205, 223].

Once the detailed geometry of the composite is accounted for in the model, the next step is to feed the models with real microstructures, in order to 1) validate the models —otherwise discrepancies between predictions and measures can always be blamed on the approximations implicit in idealized

microstructures—, and 2) obtain more accurate predictions on the behaviour and performance of the materials and the fabrication processes. And here is where computer vision enters again the world of composites. Modern microscopy techniques, either optical and non optical [60], allow detailed 2D and 3D imaging of composites, with enough spatial resolution as to resolve their microstructure. Blanc *et al.* [25] developed an image processing method to estimate the true principal directions and fibre orientation distribution from a single section of a composite using optical microscopy. The method applies to reinforcements composed of several main fibre directions and with cylindrical fibres bundled in threads. Although it provides some useful 3D parameters, the 2D approach fails to depict completely the complex 3D geometry of the material.

Tomography and other 3D imaging technologies are widely used to analyze volumetric samples of composites, but the large volumes of data for even moderately sized samples discourage human operated exhaustive analysis or repeated measures for statistically sound characterization. However, relative limited resolution, low dynamic range, few quantization levels, and significant noise typical of this type of images make difficult the automated analysis of the samples. A previous attempt has been made to extract the geometric properties of a C/C composite [255]. However, the carbon matrix is very difficult to separate from the carbon fibres, and only the separation of cavities from the solid phases has been performed. Thus, exploiting at full length the wealth of information contained in a tomography becomes a challenging task, in the quest to put an end to the use of tomographic images as collections of 2D images from series of slices under the microscope, thus preventing the full development of all their potential.

The sample

The French aeronautics industrial company Snecma Propulsion Solide has more than 30 years of experience in isothermal CVI for the development of proprietary C/C, C/SiC, and SiC/SiC composites. Figure 2.27 shows a $200 \times 200 \times 200$ portion of a synchrotron microtomography of a sample of

Figure 2.27. *A synchrotron microtomography of a* 3.3 mm^2 *(side length* < 1.5 *mm) cubic sample of a C/C composite. Voxel size is* 7.45 μm.

one of SPS's C/C composites. Figure 2.28 shows a 200×200 slice of the microtomography in Figure 2.27.

The image was obtained at the European Synchrotron Radiation Facility (ESRF) ID19 High Resolution Diffraction Topography Beamline, dedicated to radiography (absorption and phase contrast imaging) microtomography, and diffraction imaging (topography, analyzer-based imaging) experiments [62, 63]. C/C composites, due to the close densities of inclusions and matrix, pose a great challenge to X ray imaging systems. The X ray beams produced at third generation synchrotron radiation facilities such as the ESRF at Grenoble (France) have a high degree of coherence. This results from the small source size, σ, about 50 μm, and the large source to sample

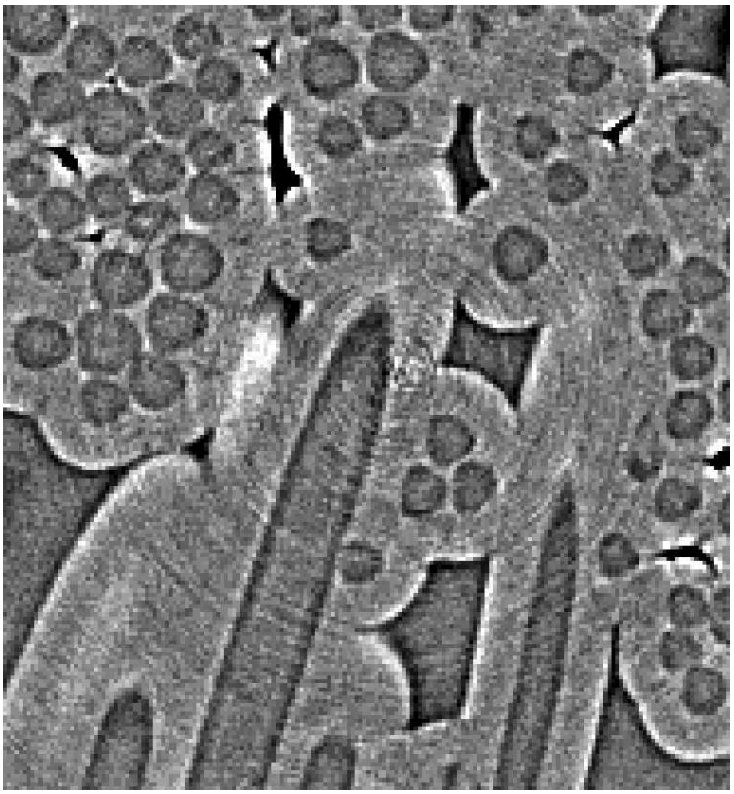


Figure 2.28. *A slice (2.22 mm^2) of the microtomography in Figure 2.27.*

distance, L, in the 100 m range, so that the transverse coherence length, $\mathrm{dc} = \lambda L/2\sigma$, is in the 100 μm range.

Phase jumps occur at the edges of a particle or porosity embedded in a matrix having a different index of refraction, and phase retrieval allows obtaining the local phase shift, which is proportional to the density. Besides, sensitivity is increased by varying the distance from sample to detector [62], either for light materials such as polymers, or for composites made up of materials with neighbouring densities such as C/C composites. However, this technique applies only on very small samples for which all the fibres show the same orientation. Indeed, in larger samples, where fibres perpendicular to the tomography axis are likely to occur, these fibres cause very high cumulated phase values which go beyond the linear range of the detector. The resulting image is blurred, rendering the separation of fibres and matrix impossible [61]. The image shown in this paper was obtained using phase contrast with fixed

distance from sample to detector [63].

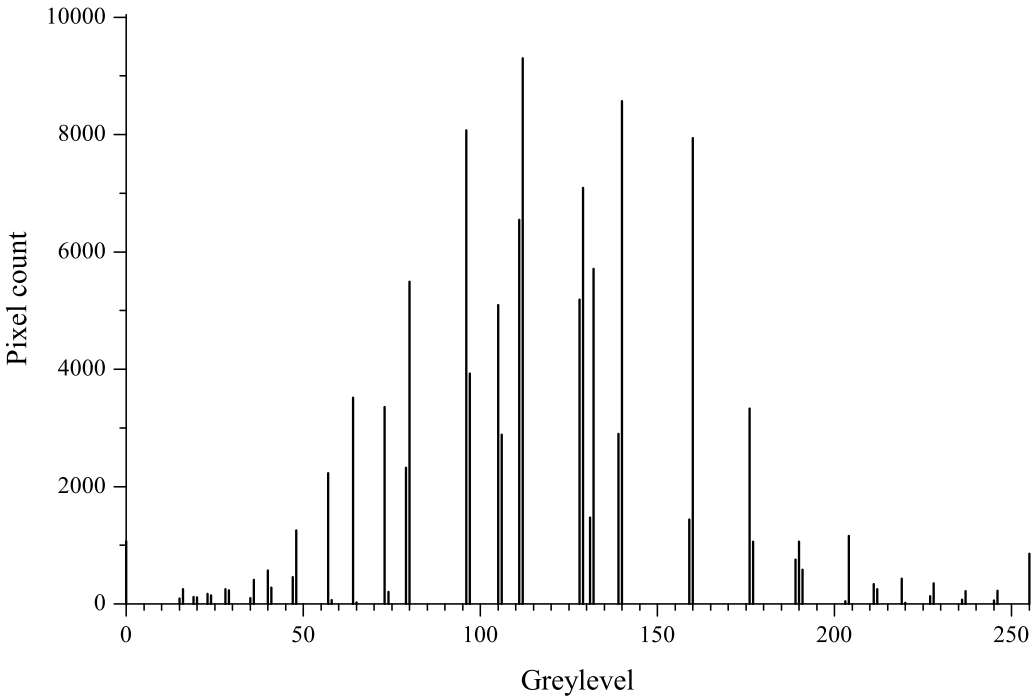


Figure 2.29. *Histogram of the image in Figure 2.28. There are 53 levels out of 256 possible levels in the dynamic range, grouped in 27 groups, thus reducing the effective dynamic range to 27 levels.*

Given that we are dealing with a C/C composite, there is no doubt that the image in Figure 2.27 provides a very good discrimination to the human observer between fibres and matrix. Cavities can also be clearly seen in the image. However, Figure 2.27 also shows a high level of noise, poor use of the dynamic range, and moderate resolution. Resolution is expected to improve in the near future. X ray detection is performed through film or visible light scintillators, and, therefore, the system is diffraction limited. X ray lenses to magnify the image before the detector, such as Kirkpatrick-Baez focusing devices, may overcome this limitation. Nevertheless, resolution is also limited by data bandwidth and storage capabilities.

However, limited resolution is not the worst characteristic of the image. Noise levels and poor quantization are. Figure 2.29 shows the histogram of the slice in Figure 2.28. Out of 256 possible levels in the 8 bit quantization,

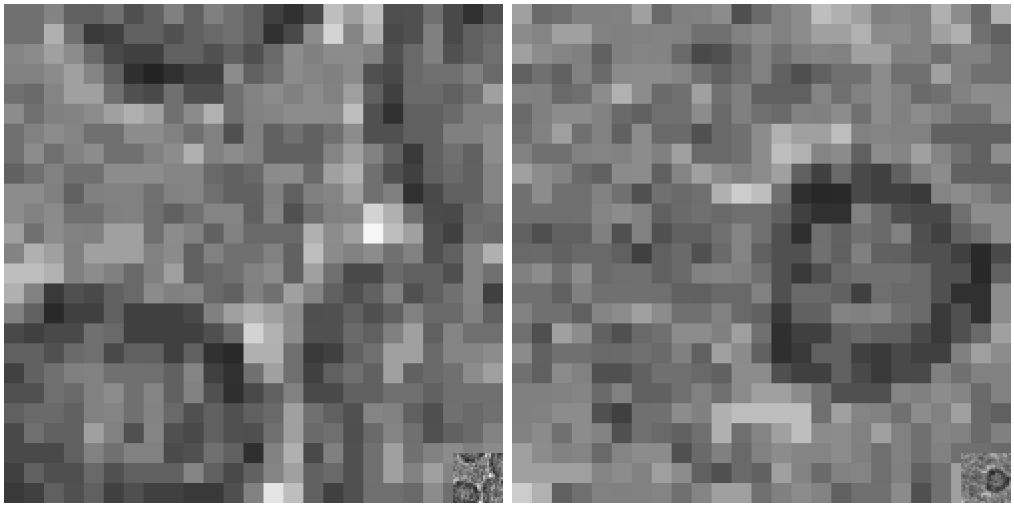


Figure 2.30. *Details of fibres/matrix in the image in Figure 2.28. The small windows at the lower right corners show the entire detail at normal scale.*

only 53 have a pixel count greater than 0. Moreover, 51 of the 53 levels are grouped in 25 groups, thus reducing the effective dynamic range to 27 levels, i.e. less than 6 bit quantization. The effect of the noise and the reduced dynamic range can be seen in the details in Figure 2.30. Grayscale based per pixel segmentation of fibres from matrix is simply not possible. Clearly the human eye (see the small window at the lower right corners) is performing high level processing involving edge detection and pattern matching —the fibres are expected to be circular/elliptical— to segment the image. See the effect that just a couple of brighter pixels have on the recognition of the fibre in the lower right corner of the detail on the left of Figure 2.30.

Within this context, our aim is using computer vision to automatically extract the fibres from the matrix, to allow the detailed characterization of the material. Once the fibres have been extracted from the matrix, precise measures such as volume fraction, aggregation, distance between fibres, or curvature can be performed. But even more, the target is literally to strip them naked, so that the real microstructure of the composite can be captured in 3D, ready to be fed into any of the models mentioned above. This also opens the doors to the use of enhanced reality computer graphics techniques to provide designers and manufacturers with increased insight on the

structure of the composite.

Preprocessing

As we have just seen, individual greylevel pixels are not suitable to separate fibre pixels from matrix pixels. It is the edges —contrast between adjacent regions in the image— that determine the boundaries of the fibres. The rest of the fibre is restituted by our visual system from the edge information. Thus, we have to use the edges to define the fibres in the image. Reducing the noise level always helps, but any noise filtering must respect the edges in the image. Otherwise, we would be losing the information we are seeking, the basis of the segmentation procedure. Thus, for instance, the noise is high frequency, but low pass filtering would also blur the edges. Median filtering could preserve to some extent the edges, but the resolution is too poor: The edges are too narrow as to not be wiped away by any but the smallest —and useless— of windows. One such a noise reduction method, efficiently removing high frequency noise while preserving the edges, is anisotropic diffusion [24, 188, 197, 246, 265].

Perona and Malik [197] noted that the convolution of an image $I_0(x, y)$ with a Gaussian kernel

$$K_\sigma(x, y) = \frac{1}{2\pi\sigma^2} \exp(-\frac{|x|^2 + |y|^2}{2\sigma^2}) \tag{2.7}$$

with standard deviation σ yields the same result as the solution of the isotropic diffusion PDE (heat) equation

$$\frac{\partial I(x, y, t)}{\partial t} = \text{div}(\nabla I(x, y, t)), \tag{2.8}$$

where $I(x, y, t)$ is the image $I(x, y)$ at time $t = 0.5\sigma^2$, with initial conditions $I(x, y, 0) = I_0(x, y)$, and ∇I is the image gradient.

Introducing in (2.8) as *diffusion conductance* or *diffusivity*, $g(s)$, a rapidly decreasing function of an edge detector such as the gradient magnitude, $s = |\nabla I|$,

$$\frac{\partial I(x, y, t)}{\partial t} = \text{div}(g(|\nabla I(x, y, t)|)\nabla I(x, y, t)), \tag{2.9}$$

smoothing on both sides of edges becomes much stronger than across them. A diffusivity constant with time but varying with location (x, y) would make (2.8) a linear nonhomogeneous diffusion equation. However, if g is made a function of time, as in (2.9), the diffusion equation becomes nonlinear and nonhomogeneous, referred as *anisotropic* in the image processing literature, even when conventional PDE terminology reserves the term for the case where the diffusivity is a tensor, varying both with location and direction[7].

Perona and Malik suggested

$$g(s) = \frac{1}{1 + s^2/\kappa^2} \tag{2.10}$$

and

$$g(s) = \exp(-s^2/\kappa^2) \tag{2.11}$$

as diffusivity functions. Since then, many other diffusivity functions have been suggested [24, 264]. In fact, Black *et al.* [24] demonstrated that anisotropic diffusion in the sense of (2.9) is the gradient descent of an estimation problem with a robust error norm induced by $g(s)$, thus providing a sound theoretical foundation to choose adequate diffusivity functions.

As several authors have revealed [45, 126, 188, 274], (2.9) is an ill-posed problem, in the sense that images close to each other are likely to diverge during the process [274], but it can be stabilized by regularization. One common approach [45] is to smooth the variable of the diffusivity, i.e. to use a smoothed version of the image for the gradient in each step, as in

$$\frac{\partial I(x, y, t)}{\partial t} = \operatorname{div}(g(|\nabla I_\sigma(x, y, t)|)\nabla I(x, y, t)), \tag{2.12}$$

where $I_\sigma = K_\sigma * I$ with a suitable local convolution kernel K_σ of width σ, for instance a Gaussian kernel. However, Weickert and Benhamouda [263] proved that a standard spatial finite difference discretization is sufficient to turn (2.9) into a well posed system of nonlinear ordinary differential equations.

[7]The consequence of this "pseudoanisotropy" is that only the magnitude but not the direction of the diffusion flux can be controlled. Noise close to edges remains unchanged due to the small flux in the vicinity of edges. To enable smoothing parallel to edges, (2.9) must be generalized with a diffusivity matrix G with nonzero off diagonal elements (see, for instance, [79, 261, 262, 264]), thus rendering it truly anisotropic.

Therefore, direct implementations of the Perona-Malik filter tend to work reasonably well because of the regularizing effect of the discretization.

The extension to D dimensions is straightforward. The diffusion process is described by the equation

$$\frac{\partial I(\mathbf{x},t)}{\partial t} = \mathrm{div}(G \cdot \nabla I(\mathbf{x},t)), \tag{2.13}$$

where $\mathbf{x} \in \mathbb{R}^D$ and G is a square $D \times D$ diffusivity matrix. Equation 2.9 is (2.13) with G a diagonal 2×2 matrix with equal diagonal elements $g(|\nabla I|)$.

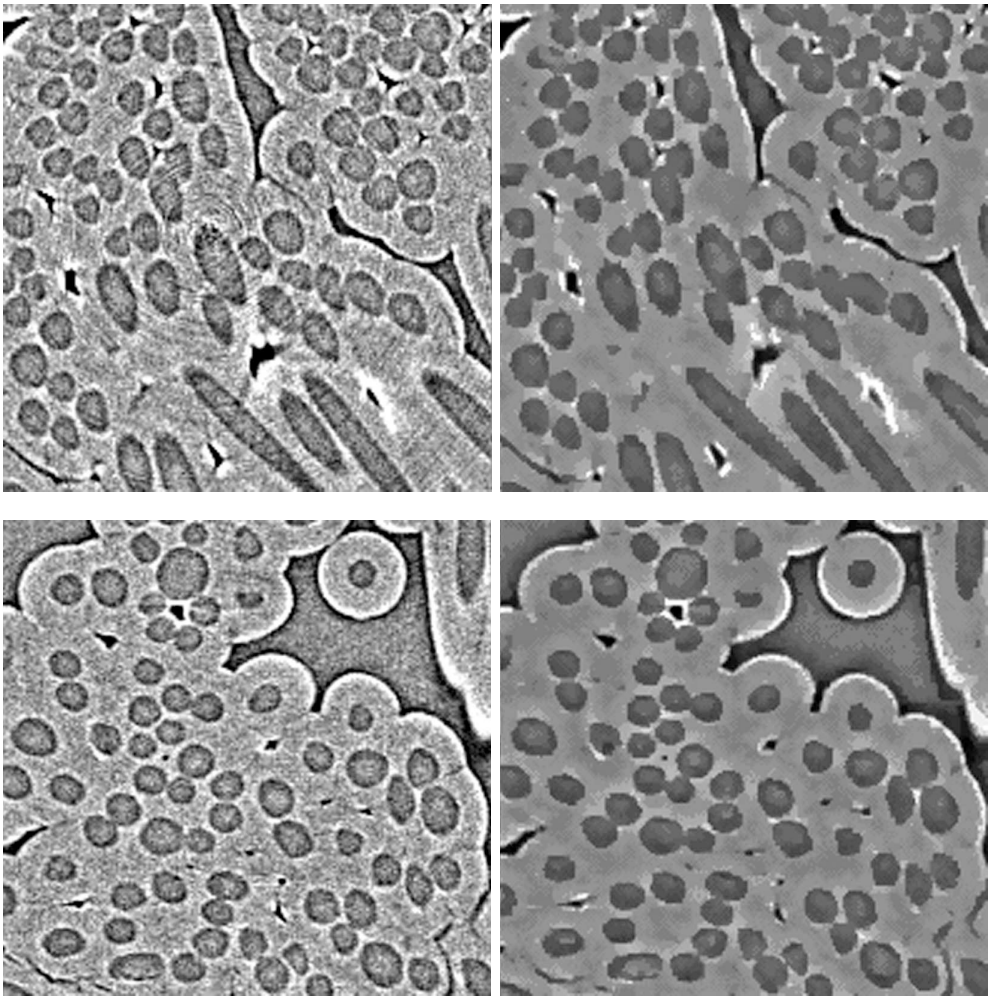


Figure 2.31. *Two slices of the tomography in Figure 2.27 before (left) and after (right) filtering with 3D anisotropic diffusion.*

Figure 2.31 shows the results of applying (2.13) with (2.10) and $D = 3$ to the tomography in Figure 2.27, and they are quite satisfactory. The relevant edges have been preserved while most of the noise has been wiped away.

Extraction of the fibres

However good the results in Figure 2.31 regarding noise reduction and mesoscale feature preservation, fibres cannot yet be separated on the basis of individual voxel gray levels. Many fibres show brighter nuclei inside darker boundaries, whose gray levels can also be encountered in the matrix. Our approach is masking all pixels within edges. For that purpose we used a differential profiler [157] along voxel rows. A differential profile is obtained by constructing an array of integers where every element represents the lowpass filtered gradient of gray level along a scan line. This is easily implemented using exclusively integer arithmetic. Let $I_{x,y,z}$ be the grayscale image. Let P_l be an integer vector to hold the profile of a scan line, and assume without loss of generality that the scan direction is along $+x$. Combining forward differencing

$$P_l = I_{l,y,z} - I_{l+1,y,z} \tag{2.14}$$

with lowpass filtering

$$P_l = \frac{1}{3}(P_{l-1} + P_l + P_{l+1}), \tag{2.15}$$

and forgetting about the dividing constant, we get

$$P_l = I_{l-1,y,z} - I_{l,y,z} + I_{l,y,z} - I_{l+1,y,z} + I_{l+1,y,z} - I_{l+2,y,z} = I_{l-1,y,z} - I_{l+2,y,z} \tag{2.16}$$

and thus the whole procedure to obtain the profile is reduced to a sequence of integer subtractions of gray levels. Near zero profile segments reflect parts of the image where no significant variations of gray level occur. Positive peaks reflect decreasing gray levels, such as when entering a fibre, and negative peaks increasing bright, such as when leaving a fibre. The narrower the peak, the faster the variation. The higher the peak, the larger the variation. Therefore, peak shape and size tell us everything we need to know about bright transitions along a given direction.

Figure 2.32 shows a gray level profile (above) and the corresponding differential profile (center) along a line of our tomography (below). The vertical arrows show the transitions in the image corresponding to the peaks above or below given thresholds (dashed lines) in the differential profile. A tentative gray level threshold is also plotted on the grey level profile, to show the lack of robustness of that approach due to the brighter centers of some fibres.

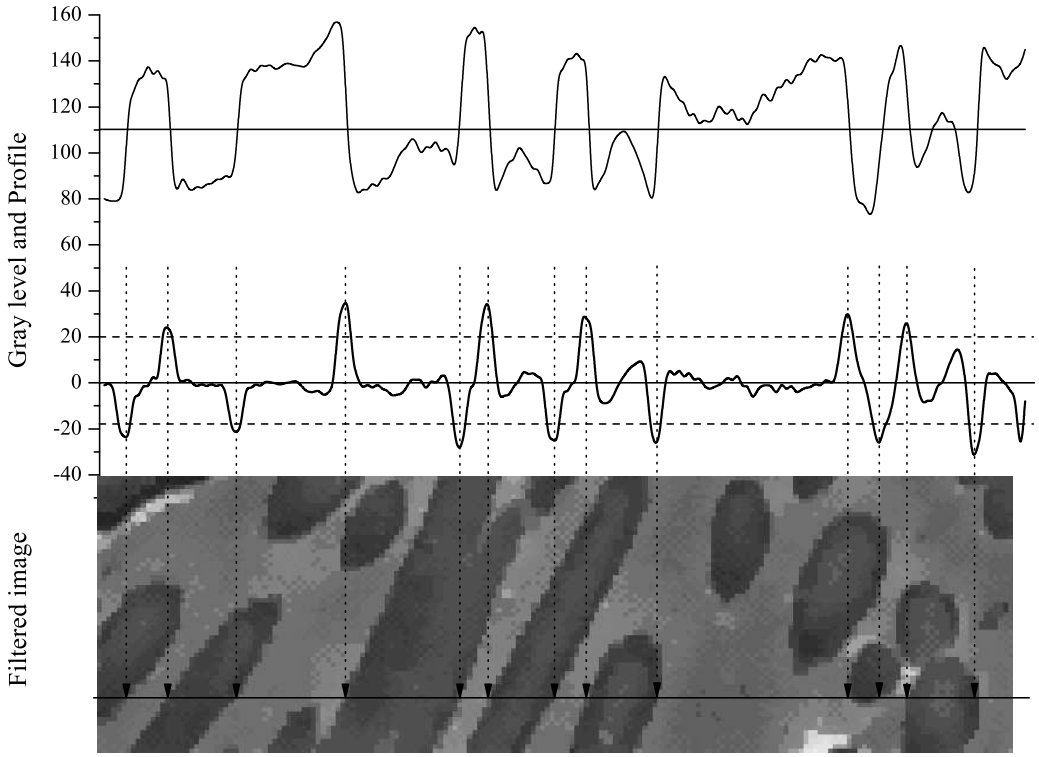


Figure 2.32. *Edge detection by profiling scan lines in slices of the tomography in Figure 2.27 after filtering with 3D anisotropic diffusion.*

To obtain a mask of fibre voxels, therefore, we profile the voxel rows along the three possible orientations —both directions— in the 3D image, such that all voxels along a differential profile from a positive peak to a negative peak are marked as foreground pixels, and the rest as background pixels. Some fibres are "broken", i.e. the dark boundary does not surround the entire fibre, and therefore some profiles may end up with spurious rectilinear spikes protruding from the fibre mask, due to the lack of a negative peak, or well similar rectilinear structures may be missing from the interior of a fibre due

to the lack of a positive peak, or a spurious negative peak inside a fibre. However, spatial correlation between the differently oriented slices in the 3D image solve part of these problems —interior fibre voxels not marked in the profiles in a given orientation are likely to be marked in the profiles along any of the transversal directions— and the a priori knowledge of the morphology of normal fibres is enough to get rid of the majority of the thin, rectilinear, spike-like artifacts protruding from "broken" fibres in a simple postprocessing stage.

After a mask of fibre voxels has thus been obtained, a labelling algorithm is used to refine the mask, and, later, to identify each fibre bundle. Refining the fibre mask consists in getting rid of some bright voxels that may be adhered to the outside face of fibre boundaries, as remnants of matrix material. This is achieved by labelling the foreground voxels in the mask with a single label, but only those corresponding to original gray levels below a given threshold. A second labelling of the mask will set as background voxels all foreground voxels belonging to connected components of foreground voxels unlabelled in the previous labelling. This ensures that the bright voxel hunt takes effect only in groups of voxels on the exterior of fibres, without affecting the bright cores of fibres. The result of this two stage cleaning is a depurated mask of fibre voxels where bright voxels are only permitted in the interior of fibres.

Once the mask of fibres has been obtained, it is possible to identify every fibre bundle by labelling all connected components in the mask, see Figure 2.33. It is also possible, by means of mathematical morphology, basically erosions and dilations, to isolate every fibre, such that each individual fibre can be identified and characterized. Note that the same procedure may be applied to cavities —profile peaks bounding the cavities are significantly taller and narrower due to the high contrast of their edges—, to extract and characterize them, see Figure 2.34.

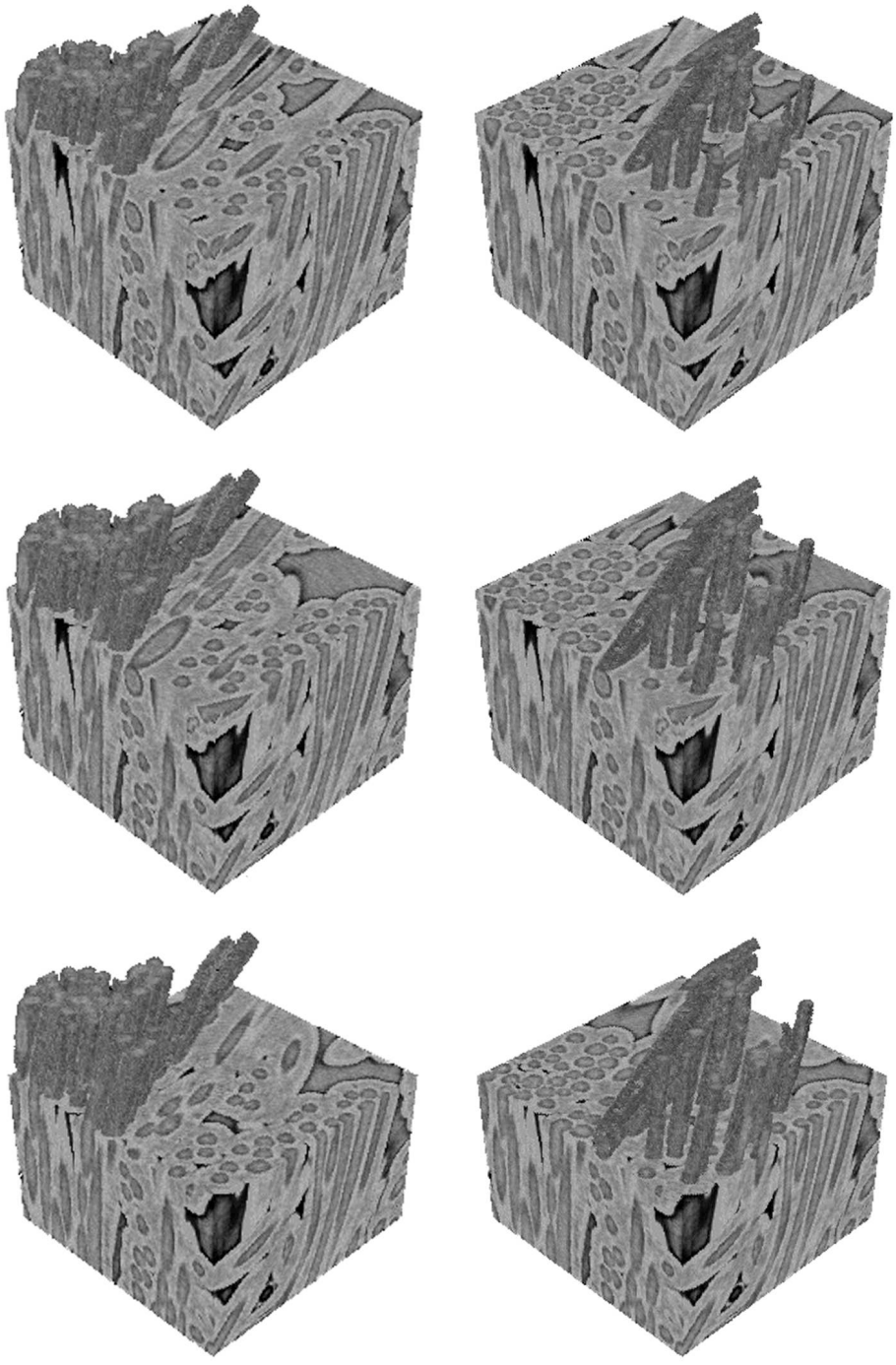


Figure 2.33. *Partial renderings of some fibre bundles using the original gray levels in the tomography.*

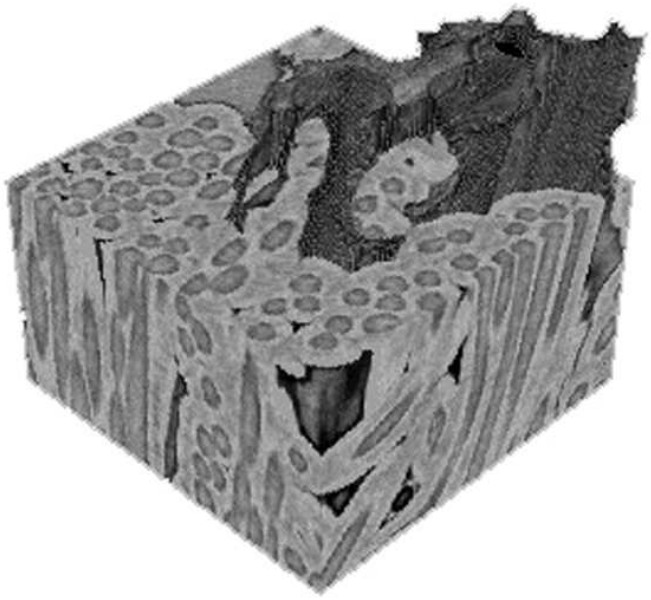


Figure 2.34. *Partial rendering of one of the cavities formed in the CVI process, using the original gray levels in the tomography.*

Benchmark results

To study the performance of hybrid labelling in the 3D case, I labelled the mask of fibres with the classical recursive technique, with its iterative version, with the two pass Union Find iterative technique, and with the hybrid technique, and recorded recursion and stack data, global equivalence table data, and time. For a total of 43 components, with an average size of 54 466 voxels, representing a volume fraction of 32%, the classical recursive technique required a total of 2 342 055 recursive function calls, with a recursion depth of 281 607. Hybrid labelling reduced these figures down to 206 900 (8.8%) and 15 481 (5.5%), respectively. These results were obtained when the scan direction was along z (see axes in Figure 2.27), the dominant orientation of the fibres in the sample. Table 2.7 summarizes the recursion call data for both techniques, and different scan orientations of the hybrid algorithm. The effect of the obvious anisotropy of the fibrous components is

clearly shown, even when many fibres in the sample do not follow the main orientation. However, even in the worst case (along x), the hybrid technique keeps recursion depth below 10% of the classical recursive technique. Hybrid labelling scanning along z required just 1/16 of the stack size used by the classical technique. The iterative version of the classical recursive technique needed a custom stack deep enough for 252 341 voxel locations (90% of the recursion depth of the recursive algorithm), i.e. a 2.9 Mb data structure.

Recursive calls	**Total**	**Average**	**Depth**
Classical	2 342 055	54 466	281 607
Iterative			252 341
(Stack depth)			89.6%
Hybrid along x	284 814	6 623	27 617
	12.2%	12.2%	9.8%
Hybrid along y	234 872	5 462	26 246
	10.0%	10.0%	9.3%
Hybrid along z	206 900	4 811	15 481
	8.8%	8.8%	5.5%

Table 2.7. *Summary of recursion and stack data in the labelling of all fibres in the microtomography in Figure 2.27.*

Regarding speed, the classical recursive technique required 147 ms to label the entire sample, versus the 91 ms (62%) required by the hybrid technique. Scan orientation had notable impact on the speed of the hybrid technique, as expected: Scanning along z used 84% of the time scanning along x. The iterative version of the classical recursive technique needed 167 ms per image, 114% of the recursive time. The two pass iterative technique took a long 498 ms to label the sample, and a global equivalence table able to handle 23 719 labels —for 43 components. The 5-fold increase in time is understandable taking into account that it encountered 1 285 286 merge points, each one needing a Union operation involving a variable-length search in the equivalence table. Table 2.8 summarizes these results. All performance data was obtained on a iPentium M 2.13 GHz, 1 Gb RAM, under OS MS WindowsXP, with the C/C++ compiler of the Borland C++Builder 6 package.

Speed	**Total (ms)**	
Recursive	146.9	
Iterative	167.3	113.9%
Hybrid along x	108.9	74.1%
Hybrid along y	99.7	67.9%
Hybrid along z	90.9	61.9%
Union-Find	497.8	339.0%

Table 2.8. *Performance data for the labelling of all fibres in the microtomography in Figure 2.27.*

2.5 Concluding remarks

I have described and studied the application to image processing and machine vision of a new technique for single pass labelling of connected components, wholly compatible with the classical recursive technique, such that any system using the latter can be upgraded just by replacing the recursive function. I also have presented test data, synthetic and real, supporting the claim that the use of this technique is advisable in a significant number of cases.

Hybrid labelling performs less consecutive recursive calls than classical recursive labelling in any case. This implies a significant reduction in the required stack size, therefore reducing the probability of a stack overflow interrupting an unsupervised critical application, or just allowing the processing of images whose size prevented the use of recursive labelling and its advantages with respect to iterative labelling. The decrease in the required stack size grows with increasing spatial structure in the mask of foreground pixels, where relatively compact objects are present. Direct conversion of the recursive technique to iterative is also a valid alternative to tackle stack overflow problems, but at the cost of managing a custom stack in the heap which has shown to require even more space than the images to be labelled, an overhead which also pays its toll in speed when compared to the proposed algorithm. I have also obtained upper and lower bounds for the recursion depth of the recursive and the hybrid algorithms and for the stack depth of the iterative version, which serve to determine the practical limits of applicability.

This improvement in robustness implies, in the particular, worst case of random images of uniformly distributed foreground pixels, an increased execution time below the percolation threshold. However, this occurs with relatively low execution times, and thus it is not that significant. Above the percolation threshold, with longer times, the trend is inverted and hybrid labelling needs just a third of the time needed by the classical recursive technique, such that, overall, hybrid labelling is clearly faster. The average performances of hybrid labelling and the two-pass iterative technique on uniform random images are virtually equal. However, if object characterization is required, or labelling just a given object is a must, hybrid labelling is

the choice. The two-pass technique can also be used to characterize objects during the labelling, but the required modifications carry a considerable overhead, thus hindering its performance. Versatility and ease of implementation are also subject matters when selecting an algorithm.

Uniform random images are the worst case scenario for hybrid labelling. Once some spatial structure appears, it does not other than improving. The synthetic Block set is a neutral approximation to the kind of imagery usually found in industrial machine vision and other image processing applications where a few, usually significant, and relatively compact objects are often expected. When dealing with the Block set, hybrid labelling behaves like iterative labelling, but reducing time consumption almost by a half, and to one fourth with respect to classical recursive labelling. Moreover, feasible image sizes before risking a stack overflow are increased several orders of magnitude.

Spatial structure favours hybrid and iterative labelling, but the former performs better because the latter is hindered by the need to handle a global equivalence table big enough to store as many labels as expected in the worst case. The size of this table can be reduced, thus decreasing the gap with hybrid labelling, but this decreases the robustness of the iterative technique, because the possibility of an image with an unpredicted large number of objects cannot usually be dismissed in real world applications.

The Ising set is somewhere in between, the objects in it do not show the same compactness as in the Block set, but a clear spatial structure can be noted, as a neutral approximation to remote sensing imagery and the like. On it, hybrid labelling already holds the winning hand, even if not by as much advantage as on the Block set —stack reduction is decreased to a "modest" 36 times. Tests with different image sizes show the same trends, suggesting that the above conclusions may hold for a wide range of image sizes.

The performance data also illustrates the fact that spatial structure drastically affects the behaviour of connected components labelling algorithms. Component labelling algorithms are not an exception, nor a special case, in image processing. Many non trivial general purpose (i.e. non specifically ap-

plication driven) image processing algorithms suffer of this dependence. Care should be taken when considering performance data irrespective of the spatial structure of the test images, as they may well be misleading for specific applications or general conclusions.

I have also presented hybrid versions of Wolff's algorithm and of the Swendsen-Wang algorithm, to exemplify the feasibility and advantages of implementing the new algorithm into standard techniques having a labelling algorithm at their core, either recursive or iterative, in both cases with an increase in speed. The hybrid Wolff's algorithm benefits also of the corresponding relaxation in the stack requirements without increasing the complexity of the algorithm, and the hybrid Swendsen-Wang algorithm has considerably simplified code and easier component characterization as additional advantages when compared to the original implementation.

Recursive techniques are specially suitable when object labelling is just an intermediate step for checking connectivity between opposite borders of an image, and not the goal itself. Then, hybrid labelling, while keeping the benefit of a reduced stack, significantly reduces time consumption with respect to recursive labelling. Traditional analysis of connectivity, involving object labelling with a two-pass algorithm and the subsequent scan of opposite borders of the image, has no choice here, as shown by the fact that hybrid labelling needed less than one tenth of the time needed by iterative labelling for any of the image configurations tested. Additionally, hybrid labelling permits the characterization of the spanning object in parallel with the detection.

Hybrid labelling can be easily modified to work on one dimensional image arrays, which is the typical storage format of frame grabbers, speeding it up more than a further ten percent, while relaxing even more the pressure on the stack, not by further decreasing the recursion depth, but by decreasing the room needed for each recursive call. The stack reduction is also true for the classical recursive technique and its iterative version, but the speeding up is far less noticeable, because they do not favour rows or columns.

I have also illustrated the advantages of using hybrid labelling in a couple

of real world machine vision applications: In a 2D, real time study case, the speed of the particular task was doubled and the minimum size for an object to cause a stack overflow was increased well beyond image size, thus rending unattainable such an inconvenient possibility. In a 3D study case, hybrid labelling showed the best performance by far, and a considerable gain in stack use, which was decreased to less than six percent of the classical recursive technique.

The 3D results are worth discussing, because, in principle, it is with 2D images where hybrid labelling presents a greater advantage: Half the directions are changed from recursive to iterative. The higher the number of dimensions, the lesser the improvement, because every additional dimension has to be dealt with using recursive calls, such that, in the limit, hybrid labelling would converge to pure recursive labelling[8]. Thus, in the 3D case, one direction is handled with iterative scanning whereas there are two directions handled with recursive calls. However, the improvements of the hybrid approach are clearly noticeable in the 3D study case. The fact that recursion is used for two directions may hide the all important fact that the number of recursive jumps to adjacent voxels along each of these two directions is also reduced, because for every recursive jump a whole burst is labelled, thus preventing any further recursive calls in the recursive directions to enter the voxels in that burst. The result is that recursive calls are not only eliminated along the direction chosen for iterative scanning, but also reduced in the other two directions. The rule is that hybrid labelling requires one and only one recursive call per burst, and this holds no matter the number of dimensions.

Under the denomination "hybrid", which denotes the mixed iterative/recursive approach, hybrid labelling does not incorporate *all* features of the classical recursive and iterative techniques. Seemingly, it only keeps the best features of each world. The iterative side of hybrid labelling is assured by the recursive side that no "merge" points will be encountered anywhere along its way. Preventing collisions between equivalence classes while being able to explore one of the spatial directions by means of closed iterative loops is

[8]Fortunately, time has not come yet for ∞D images to be of much practical interest.

advantage enough to outweight the fact that bursts remain one dimensional no matter the number of dimensions of the image.

The 3D study case also shows another feature of hybrid labelling: It permits to take advantage of anisotropy, such that when predominant features in the images are expected to favour a given direction, choosing it as the direction of iterative scanning within the algorithm speeds up the labelling and reduces even further stack use.

All in all, hybrid labelling shows that recursive object labelling is a valid alternative to the iterative two pass techniques, just by diminishing the drawback of the classical recursive technique: The need of an oversized stack due to the high number of consecutive recursive calls. Hybrid labelling is a smart alternative to the two-pass algorithms because it transfers the overhead of the global equivalence table to the system. Indeed, the system stack may be seen as a local, "per-component equivalence table", optimally managed by the processor through built-in processor instructions, and therefore the algorithm does not need to manage additional data structures, as it happens with the two pass iterative algorithm, or with the iterative version of the classical recursive algorithm. The system stack is an inherent characteristic of the hardware, therefore very fast and efficient, transparently accessed via recursive function calls through the built-in *push* and *pop* instructions, without any further consideration for the programmer than its limited size. Classical recursive labelling does it also, but less optimally than hybrid labelling, such as I have just shown.

Chapter 3

Stöhr Edge Encoding

'The ancient teachers of this science', said he,
'promised impossibilities and performed nothing.
The modern masters promise very little;
... but have indeed performed miracles'.

3.1 Introduction

One of the major fields in computer vision is object recognition. In a 1996 survey on the application fields for industrial image processing systems [173] (as cited in [258]), the more demanded topics were "examination of contour", "positioning", "measuring", and "object identification". Another major field of machine vision is texture analysis. Industrial machine vision systems are usually constrained by high speed requirements, in order to fit into high throughput production lines. This requirement, while alleviated by the exponential growth of computer performance, is contrasted with the growing complexity and computational overhead of a considerable number of image processing methods in the literature. As Pietikäinen and Ojala put it [198],

> "... we cannot help wondering if the research has focused too heavily on a few theoretically impressive (and consequently computationally complex) paradigms?".

I describe in the following a computationally simple method for encoding the edge of a binary shape or object of foreground pixels in a masked image, which I call Stöhr Edge Encoding (SEE), based on the addition of the first four terms of a 4-Stöhr sequence [235] to encode all the possible configurations of every corner of all pixels in the shape, and will show its utility for a number of typical computer vision problems. The method uses a grid of nodes corresponding to the vertices of every pixel in the image —thus reconstructing the original mesh from the equivalent image representation, see section 1.3. This grid has the same size as the image if the right and bottom borders can be discarded, as it is usually the case[1].

The encoding of an object edge, or all edges in an image is as follows: A given integer value (weight) is assigned to each vertex of every pixel in the object, the same assignment for all pixels, such that in a single scan the corresponding value of each vertex of each pixel is added to the corresponding node in the node grid. After the scan, every node in the grid holds a code describing the configuration of the node, i.e. the configuration of the 4-pixel neighborhood of the node.

The method uses as weights the first elements in a Stöhr sequence, but would also work with any subset of a B_4-Sequence [96]. Using the first members of a Stöhr sequence warrants that the results of all the possible sums where each element can occur at most once span an entire integer interval from 0 on, i.e. every resulting code for any possible node configuration (node type hereafter) belongs to the interval, and every integer in the interval is a meaningful and distinct node type code. This simplifies the computation of Stöhr Edge Code (SEC) histograms and boundary chain codes.

Once the nodes in the contour of an object hold the corresponding Stöhr Edge Codes, a chain code describing the object shape can be constructed, just by going through the perimeter of the object collecting the successive codes. The SEC of every node designates which is the following node in the perimeter of the object —once a turning direction has been decided— such

[1] Objects for inspection are usually not allowed to touch any border of the image, or an uncertainty about their real shape and size would arise.

that contour following is straightforward. Thus, the SEC can also be used to follow efficiently the contour of the object for the generation of a classical crack code.

Without further processing, a SEC chain is less than efficient as a chain code, because there is a high degree of redundancy —once the type of a given node is known, the types of the surrounding nodes are restricted to a subset of the possible configurations—, but this also warrants a high degree of robustness, such that transmission errors may be detected and corrected. Therefore it can be used as such when noisy channels or a high error rate are expected. If this is not the case, the *a priori* knowledge on the constraints on neighbouring SEC implicit in SEE allows the use of only two bits per code, thus generating a compact encoding scheme. Reconstruction from either the robust or the compact SEC chain is easy in both cases using specific reconstruction methods.

Some attempts have been made to use chain codes in pattern matching and object recognition [135,138,145,167,201]. I demonstrate the convenience of SEE chain codes with a simple test on pattern matching using Levenshtein distances with the classical crack code [209], the recently introduced vertex chain code [34], and the SEE chain code.

However, perhaps the primary interest of SEC chain codes is that their concept can be extended to 3D. Thus, I present a 3D chain code (a *real* 3D chain code, for 3D shapes, not curves), the first to be published up to my knowledge, and show how to encode any arbitrary 3D shape and how to reconstruct it from the corresponding chain.

The utility of SEE is not restricted to shape encoding through chain codes. If the SEE codes of an object or an entire image are accumulated on a histogram, the corresponding vector constitutes an image feature which can be used for object inspection and for texture classification.

I demonstrate the capacity for object recognition by training simple neural networks (multilayer perceptrons [104, 108, 174, 207, 210]) to differentiate between several objects with SEC histograms as the only input. Neural nets

are also trained to estimate the rotation angle of different objects. Clearly, SEC histograms used in this fashion are suitable for controlled environments where only known sets of rigid objects are expected. The restriction, however, comes accompanied by the extreme computational lightness of SEC histograms —five integer increments per foreground pixel—, which allows very high processing speeds in, on the other hand, not uncommon scenarios in industrial machine vision.

Last, but not least, I apply the SEE technique to texture classification, by using the SEC histograms of binarized texture samples from the Brodatz album [37] as input features to a k-nearest neighbor classifier [2, 66, 76], and to a SOFM [130–134]. The performance of SEE is compared to that of the well known uniform Local Binary Patterns (LBP) [189, 190], which use a similar approach to texture classification, by means of the distribution of a set of local configurations along the image.

To finish, I comment on the applicability of SEE to 3D images and specify the applications into the numerical model framing this report.

3.2 The algorithm

Pixels/voxels are considered parallelepipeds with edges and vertices. A grid of nodes is overlaid on the image such that all nodes are on the vertices of the pixels (voxels) in the image. Pixels are termed either foreground or background according to a given rule $\mathcal{A}$ (see section 2.1.1). An array of codes with the same size as the grid is set to all zeros. An integer weight is assigned to each vertex of every pixel in the image, such that in a single scan over the image the weight of each vertex of each foreground pixel is added to the code of the corresponding node. After the scan, every node in the mesh has a SEE code (SEC) describing the configuration of the surrounding —adjacent— pixels.

We use as weights the first four (eight in 3D) elements in the 4(-8)-Stöhr sequence, but it would also work with any subset of a $B_{4(8)}$-Sequence [96]. A B_2-Sequence, also called a Sidon sequence, is an infinite sequence of positive integers $1 = b_1 < b_2 < b_3 < \ldots$ such that all pairwise sums $b_i + b_j$ for $i \leq j$ are different. The definition is easily extended to B_n-Sequences. A h-Stöhr sequence [235] is defined as follows: Let $a_1 = 1$ and define a_{n+1} to be the least integer greater than a_n which cannot be written as the sum of at most h addends among the terms $a_1, a_2, \ldots, a_n$. Using the first four (eight) members of the 4(-8)-Stöhr sequence warrants that all the possible sums where each element can occur at most once span an entire integer interval from 0 onwards ($[0, 15]$ in the 2D case, $[0, 255]$ in 3D), i.e. every code belongs to the interval, and every integer in the interval is a meaningful and distinct node type code. This simplifies the subsequent use of the codes, either for histograms or for indexing look-up tables (LUT).

I illustrate the 2D case for clarity. The 3D case follows in a straightforward manner. The first four elements of the 4-Stöhr sequence are $\{1, 2, 4, 8\}$. The assignation of weights to each of the four vertices in a pixel determines the correspondence between each possible code and each local configuration. Every pixel is assigned the weights in the same order. See Figure 3.1 (left) for an example. The origin of coordinates is assumed to be the upper left corner of the image.

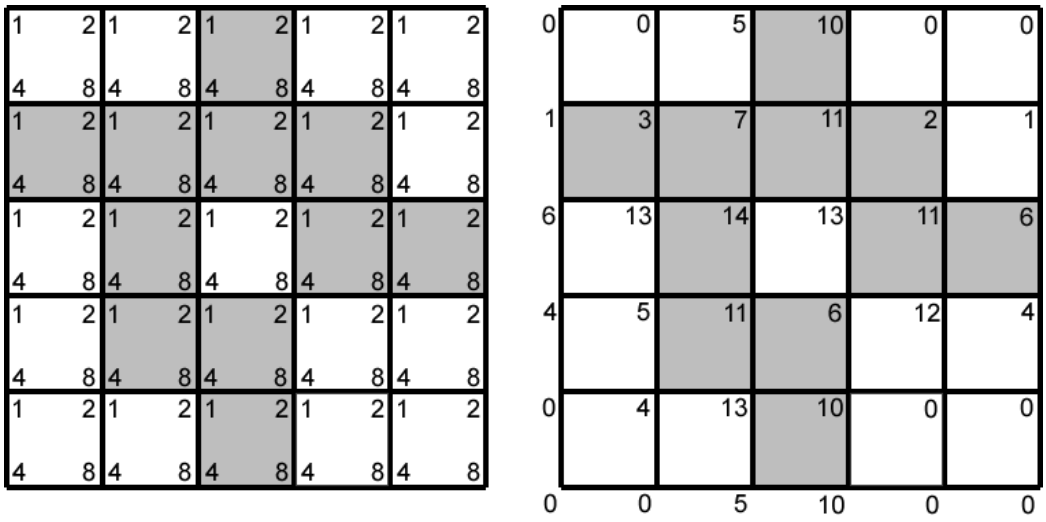


Figure 3.1. *Stöhr Edge Encoding. Node codes on the right are the result of adding up the corresponding weights of the vertices of foreground pixels on the left (cyclic boundaries have been assumed).*

Let $(b \twoheadrightarrow a)$ be an operator that increments integer a in b units. If $c_{x,y}$ denotes the code of node (x, y), on the upper left corner of pixel (x, y), a single scan over the image performing $(1 \twoheadrightarrow c_{x,y})$, $(2 \twoheadrightarrow c_{x+1,y})$, $(4 \twoheadrightarrow c_{x,y+1})$, and $(8 \twoheadrightarrow c_{x+1,y+1})$ for every foreground pixel (x, y) would produce the result in Figure 3.1 (right), where the resulting code of each node uniquely identifies its local configuration (see Figure 3.2 and Algorithm 10). The cost of the SEE scheme is just four integer increments per foreground pixel.

Algorithm 10 2D SEE

for all (x, y) $0 \rightarrow c_{x,y}$
for all $p_{x,y}$ obeying $\mathcal{A}$
$(1 \twoheadrightarrow c_{x,y})$
$(2 \twoheadrightarrow c_{x+1,y})$
$(4 \twoheadrightarrow c_{x,y+1})$
$(8 \twoheadrightarrow c_{x+1,y+1})$

The 3D case follows the same scheme, with the first eight elements of the 8-Stöhr sequence, to produce the 256 codes necessary for the 256 different

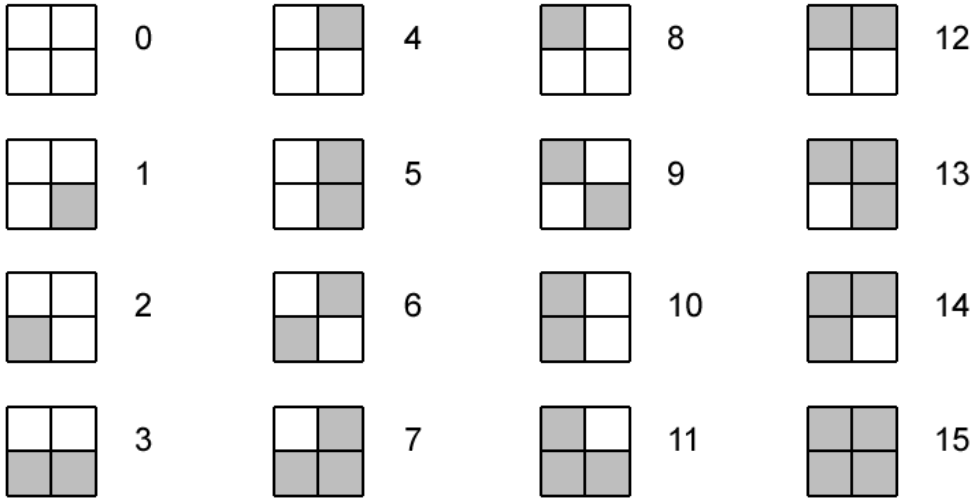


Figure 3.2. *All possible local configurations in a 2D image and the corresponding SEE codes for the weight assignment in Figure 3.1. Code 0 is a node embedded in background pixels, code 15 is a node embedded in an object. All the rest correspond to boundaries.*

local configurations in a 3D image.

3.3 Chain codes

Freeman [81] was the first to suggest the use of boundary chain codes to represent digitized shapes, in the early sixties. In later work [82], he presented some methods for processing chain codes and stated three requisites for any chain code, namely 1) information preservation, i.e. lossless coding, 2) compact storage, and 3) easy processing. He proposed the classical chain code where, from a starting pixel on the boundary of a digitized binary shape (a component in a digital image), each element in the code indicates the direction to the next pixel along the boundary. This can be one out of eight, if a 8-neighbourhood is used, or one out of four, if 4-neighbourhood, when it is commonly termed a *crack code* [209], in reference to following the "cracks" between pixels instead of jumping from pixel to pixel. To obtain the chain code a contour following algorithm is necessary, such as that in [71]. Since Freeman, in spite of extensive work on shape characterization and pattern recognition using chain codes [135, 138, 145, 167, 201], few new chain codes have been described [34, 172] apart from some structures loosely related to chain codes [27, 32, 33]. Bribiesca [34] recently proposed the vertex chain code, which counts the number of foreground pixels surrounding each vertex in the object contour. Considerable effort [46,50,127,278] has been dedicated to real time chain encoding of shapes, mainly related to video compression techniques [175].

In the following, I will describe two chain codes based in SEE (SCC), a *robust* or redundant version, and a *compact* version, will provide algorithms for obtaining both types of SCC from a shape —coding— and for obtaining the corresponding shape from a SCC —decoding. The prominent feature of SEE chain codes is that they do not need any additional contour following algorithm, as the SEC provide all the necessary information to trace the boundary of the shape. Thus, SEC can also be used for efficient computation of the crack code and the vertex code.

3.3.1 Robust code

Algorithms 11 and 12 show SEE chain code generation for 4- and 8-neighbourhoods. They assume the SEC of the image have been previously computed as in Algorithm 10 and are stored in $c_{x,y}$. c_i denotes bit i in c. Both turn clockwise and are expected to begin in a node $(x_{\mathrm{i}}, y_{\mathrm{i}})$ type 1, according to the encoding in Figure 3.1. If the image is scanned in search of the object in ascending x and y scan order, the first node with non zero SEC will always have SEC 1 (see Figure 3.2), and therefore it is not appended to the chain.

An auxiliary variable d is used to keep track of the last movement while surrounding the object according to the current SEC, using the following arbitrary assignation of integer values: $0 \rightarrow x^+$, $1 \rightarrow x^-$, $2 \rightarrow y^+$, $3 \rightarrow y^-$. This is necessary with 4-neighbourhood to convert nodes belonging to two different touching objects or two separate parts of the same object (SEC 6 and 9 in Figure 3.2) to the appropriate node type including only the pixels of the object of interest. Thus, for instance, if a node type 6 is reached from the left, the appropriate movement is downwards, and the correct node type for the chain code is 2. If it is reached from the right, the appropriate movement is upwards, and the correct node type is 4. Remember that the boundary is followed clockwise. With 8-neighbourhood, there is no need of node type corrections for the chain code. However, it is still necessary to know which was the last movement to decide the next move along the contour in nodes type 6 and 9. Thus, for instance, a node type 6 reached from the right should be followed downwards, not upwards. The rest of node types are treated the same for 4- and 8-neighbourhoods to decide the next movement along the contour. The contour is followed until the starting point is reached. Using binary masks generates compact and fast code, see Appendix D for more specific —but also more obscure— pseudocode for these and the following algorithms.

Node codes can be appended to the chain code *Code* as nibbles in an array of bytes, or well the first nibble in each byte can be used to store the code and the second a run-length code to indicate a number of equal consecutive

Algorithm 11 Robust 4-neighbourhood SEE Chain Code (R-SCC$_4$)

initialize *Code*
$d \leftarrow 0$
$x \leftarrow x_\mathrm{i} + 1$, $y \leftarrow y_\mathrm{i}$
do
  $c \leftarrow c_{x,y}$
  if $c = 6$ {**if** $d = 0$ $c \leftarrow 2$, **else if** $d = 1$ $c \leftarrow 4$}
  else if $c = 9$ {**if** $d = 2$ $c \leftarrow 8$, **else if** $d = 3$ $c \leftarrow 1$}
  append c to *Code*
  if (c_0 and $\overline{c_2}$) {$d \leftarrow 0$, **increase** x}
  else if (c_3 and $\overline{c_1}$) {$d \leftarrow 1$, **decrease** x}
  else if (c_1 and $\overline{c_0}$) {$d \leftarrow 2$, **increase** y}
  else if (c_2 and $\overline{c_3}$) {$d \leftarrow 3$, **decrease** y}
while $(x, y) \neq (x_\mathrm{i}, y_\mathrm{i})$

Algorithm 12 Robust 8-neighbourhood SEE Chain Code (R-SCC$_8$)

initialize *Code*
$d \leftarrow 0$
$x \leftarrow x_\mathrm{i} + 1$, $y \leftarrow y_\mathrm{i}$
do
  $c \leftarrow c_{x,y}$
  append c to *Code*
  if ($c = 6$ and $d = 0$) {$d \leftarrow 3$, **decrease** y}
  else if ($c = 9$ and $d = 3$) {$d \leftarrow 1$, **decrease** x}
  else if (c_0 and $\overline{c_2}$) {$d \leftarrow 0$, **increase** x}
  else if (c_3 and $\overline{c_1}$) {$d \leftarrow 1$, **decrease** x}
  else if (c_1 and $\overline{c_0}$) {$d \leftarrow 2$, **increase** y}
  else if (c_2 and $\overline{c_3}$) {$d \leftarrow 3$, **decrease** y}
while $(x, y) \neq (x_\mathrm{i}, y_\mathrm{i})$

nodes up to 16^2. This does not affect the way the chain code is obtained, and therefore is not explicit in Algorithms 11 and 12. Figure 3.3 provides an example.

[2]However, only a subset of node types can be followed by the same type in a SCC.

Algorithm 13 4-neighbourhood Crack Code using SEE

initialize *Code*
$d \leftarrow 0$
$x \leftarrow x_\mathrm{i}$, $y \leftarrow y_\mathrm{i}$
do
  $c \leftarrow c_{x,y}$
  if $c = 6$ {**if** $d = 1$ $c \leftarrow 2$, **else if** $d = 3$ $c \leftarrow 4$}
  else if $c = 9$ {**if** $d = 2$ $c \leftarrow 8$, **else if** $d = 0$ $c \leftarrow 1$}
  if (c_0 and $\overline{c_2}$) {$d \leftarrow 1$, **increase** x}
  else if (c_3 and $\overline{c_1}$) {$d \leftarrow 3$, **decrease** x}
  else if (c_1 and $\overline{c_0}$) {$d \leftarrow 2$, **increase** y}
  else if (c_2 and $\overline{c_3}$) {$d \leftarrow 0$, **decrease** y}
  append d to *Code*
while $(x, y) \neq (x_\mathrm{i}, y_\mathrm{i})$

Algorithm 14 8-neighbourhood Crack Code using SEE

initialize *Code*
$d \leftarrow 0$
$x \leftarrow x_\mathrm{i}$, $y \leftarrow y_\mathrm{i}$
do
  $c \leftarrow c_{x,y}$
  if ($c = 6$ and $p = 1$) {$d \leftarrow 0$, **decrease** y}
  else if ($c = 9$ and $p = 0$) {$d \leftarrow 3$, **decrease** x}
  else if (c_0 and $\overline{c_2}$) {$d \leftarrow 1$, **increase** x}
  else if (c_3 and $\overline{c_1}$) {$d \leftarrow 3$, **decrease** x}
  else if (c_1 and $\overline{c_0}$) {$d \leftarrow 2$, **increase** y}
  else if (c_2 and $\overline{c_3}$) {$d \leftarrow 0$, **decrease** y}
  append d to *Code*
while $(x, y) \neq (x_\mathrm{i}, y_\mathrm{i})$

R-SCC is termed robust because it includes redundancy. Once the SEC of a given node is known, the set of possible SEC for the adjacent nodes (including the next node along the boundary) is a subset of all possible SEC. If the whole number of bits is used to store each SEC along the boundary, the

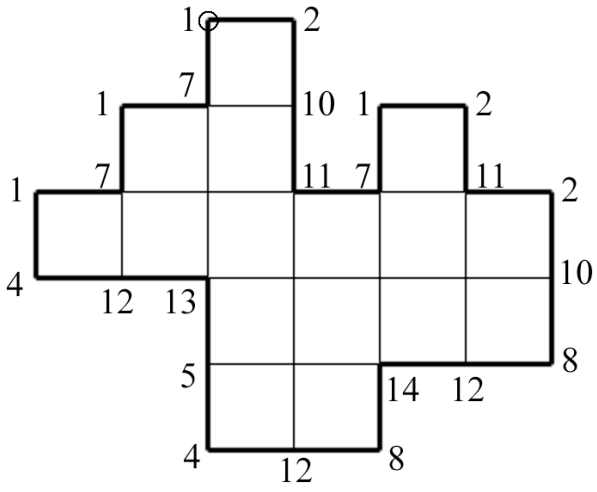


Figure 3.3. *A simple object, and the SEC of the boundary nodes.*

	Code	Length
Freeman	4 3 1 3 4 6 5 6 0 7 6 1 1	39 bits
Crack	1 2 2 1 0 1 2 1 2 2 3 3 2 3 3 0 0 3 3 0 1 0 1 0	48 bits
Vertex	1 1 2 3 2 1 2 1 3 2 1 2 1 3 1 1 3 3 2 1 1 3 1 3	48 bits
R-SCC	2 10 11 7 1 2 11 2 10 8 12 14 8 12 4 5 13 12 4 1 7 1 7	92 bits
SCC	0 2 3 3 0 0 3 0 2 0 2 3 0 3 0 1 3 2 0 0 3 0 3	46 bits

Table 3.1. *Chain codes of the shape in Figure 3.3.*

SEE chain code is thus incorporating redundancy. Half of the length of the R-SCC (when run-length is not used) is dedicated to this redundancy. This is a waste of storage in noise-free systems, but increases robustness in noisy (error-prone) environments. Thus, the R-SCC in Table 3.1 is 92 bits long, versus the 48 bit long crack and vertex codes, or the 39 bit long Freeman (classical) chain code.

Algorithms 13 and 14 show how SEE can be used to obtain the crack code. Instead of appending the SEC to the chain, it is the value of d (redefined to match the usual convention in crack codes) that is appended to the code. Not shown but fairly similar would be the algorithms to construct the vertex chain code, just by appending to the code, instead of d, the bitwise number of 1's in c.

Algorithm 15 Shape reconstruction from R-SCC *Code*

$d \leftarrow 0$
$x \leftarrow x_\mathrm{i}$, $y \leftarrow y_\mathrm{i}$
do
  extract c from $Code$
  if d= 0
    increase x
    if $c = 2$ $d \leftarrow 2$
    else if $c = 3$ $p_{x,y} \leftarrow 1$
    else if $c = 6$ $d \leftarrow 3$
    else if $c = 7$ $\{d \leftarrow 3, p_{x,y-1} \leftarrow 1\}$
    else Error
  else if $d = 1$
    decrease x
    if $c = 4$ $d \leftarrow 3$
    else if $c = 6$ $\{d \leftarrow 2, p_{x-1,y} \leftarrow 1\}$
    else if $c = 12$ $p_{x-1,y-1} \leftarrow 1$
    else if $c = 14$ $\{d \leftarrow 2, p_{x-1,y} \leftarrow 1\}$
    else Error
  else if $d = 2$
    increase y
    if $c = 8$ $d \leftarrow 1$
    else if $c = 9$ $\{d \leftarrow 0, p_{x,y} \leftarrow 1\}$
    else if $c = 10$ $p_{x-1,y} \leftarrow 1$
    else if $c = 11$ $\{d \leftarrow 0, p_{x,y} \leftarrow 1\}$
    else Error
  else
    decrease y
    if $c = 1$ $d \leftarrow 0$
    else if $c = 5$ $p_{x,y-1} \leftarrow 1$
    else if $c = 9$ $d \leftarrow 1$
    else if $c = 13$ $\{d \leftarrow 1, p_{x-1,y-1} \leftarrow 1\}$
    else Error
while $Code \neq \emptyset$

Algorithm 15 shows how to reconstruct a contour from a robust SEE chain code (R-SCC). The last movement (kept in d) determines the four

possible node types of the next code element. If it is not in the corresponding subset, there is an error in the code. This may either halt the process of reconstruction or well try a weak correction by replacing the defective element with the SEC in the legal subset closest to the defective element and having the following element in its list of allowed successors.

A costly, but more effective, error correction strategy is marking all defective elements, and perform an exhaustive ordered search for all defective elements in the corresponding legal subsets under the constraint that at the end of the reconstruction the end point of the contour must coincide with the initial point. The cost of the correction depends on the number of defective elements in the chain, but the search space for each defective element is limited to four candidates, and the probability that a wrong correction may result in matching initial and ending points is very low.

The redundancy based error detection capability of the R-SECC could be achieved with a crack code repeating each element of the chain twice. This would equate the length of the robust SEE chain code, but would differ in the performance of the correction strategy. A defective element would imply a mismatched pair. A decision should be taken about which of the two elements in the pair could be the defective one. This decreases the search space per defective element to two legal candidates (versus the four legal candidates in the R-SECC). However, redundancy in this duplicated crack code is in one direction only, either forwards or backwards, and a wrong correction can not be detected by the following element in the chain. A defective R-SECC element is detected by the previous element in the chain, but the wrong correction would cause a mismatch with the following element in the chain. Redundancy is bidirectional, all elements are linked to the previous and the following element in the chain. Therefore, the effective search space has the same size in both strategies and the robustness is greater in R-SECC due to the "distributed" redundancy.

3.3.2 Compact code

With SEE it is possible to construct a chain code the same length than the crack code or the vertex code, just by eliminating the redundancy in the R-SCC. Algorithms 16 and 17 show how. We will drop the prefix 'R' to denote the compact SEE chain code.

Algorithm 16 Compact 4-neighbourhood SEE Chain Code (SCC_4)

initialize *Code*
$d \leftarrow 0$
$x \leftarrow x_i + 1$, $y \leftarrow y_i$
do
  $c \leftarrow c_{x,y}$
  if $c = 6$ {**if** $d = 0$ $c \leftarrow 2$, **else if** $d = 1$ $c \leftarrow 4$}
  else if $c = 9$ {**if** $d = 2$ $c \leftarrow 8$, **else if** $d = 3$ $c \leftarrow 1$}
  if $d = 0$ $n \leftarrow (c_0 c_2)$
  else if $d = 1$ $n \leftarrow (c_1 c_3)$
  else if $d = 2$ $n \leftarrow (c_0 c_1)$
  else if $d = 3$ $n \leftarrow (c_2 c_3)$
  append n to *Code*
  if (c_0 and $\overline{c_2}$) {$d \leftarrow 0$, **increase** x}
  else if (c_3 and $\overline{c_1}$) {$d \leftarrow 1$, **decrease** x}
  else if (c_1 and $\overline{c_0}$) {$d \leftarrow 2$, **increase** y}
  else if (c_2 and $\overline{c_3}$) {$d \leftarrow 3$, **decrease** y}
while $(x, y) \neq (x_i, y_i)$

Chain elements are two bit long, and are formed using only the information in the SEC that is new with respect to the previous node. What part of a SEC is new in a chain depends on the direction we are approaching the node. Thus, see Figure 3.2, when we approach a node horizontally from the left (x^-, i.e. $d = 0$ in Algorithms 16 and 17), the only new information the SEC is providing with respect to the SEC of the previous node regards the pixels to the right of the current node, and (with the weight assignation of Figures 3.1 and 3.2) is stored in the bits 1 and 3 of the SEC. All the same, when we approach a node from down upwards (y^+, $d = 3$), the innovation in

Algorithm 17 Compact 8-neighbourhood SEE Chain Code (SCC_8)

initialize *Code*
$d \leftarrow 0$
$x \leftarrow x_\mathrm{i} + 1$, $y \leftarrow y_\mathrm{i}$
do
  $c \leftarrow c_{x,y}$
  if $d = 0$ $n \leftarrow (c_0 c_2)$
  else if $d = 1$ $n \leftarrow (c_1 c_3)$
  else if $d = 2$ $n \leftarrow (c_0 c_1)$
  else if $d = 3$ $n \leftarrow (c_2 c_3)$
  append n to *Code*
  if ($c = 6$ and $p = 0$) {$d \leftarrow 3$, **decrease** y}
  else if ($c = 9$ and $p = 3$) {$d \leftarrow 1$, **decrease** x}
  else if (c_0 and $\overline{c_2}$) {$d \leftarrow 0$, **increase** x}
  else if (c_3 and $\overline{c_1}$) {$d \leftarrow 1$, **decrease** x}
  else if (c_1 and $\overline{c_0}$) {$d \leftarrow 2$, **increase** y}
  else if (c_2 and $\overline{c_3}$) {$d \leftarrow 3$, **decrease** y}
while $(x, y) \neq (x_\mathrm{i}, y_\mathrm{i})$

the SEC is in the pixels above the current node, i.e. bits 3 and 2 of the current SEC. The contour following part is the same as in the previous algorithms. Appendix D provides explicit pseudocode for the bitwise operations.

Table 3.1 shows the SCC of the shape in Figure 3.3 together with the R-SCC and some other chain codes in the literature. Note that the 2 bit difference in length of the SCC with respect to the crack code or the vertex code is due to the fact that SCC assumes the contour always begins in a node of type 1 (i.e. objects are always found in normal scan order), thus the first node is never included in the chain.

Compactness in compact SCC comes at no cost in reconstruction complexity with respect to robust R-SCC. The only loss is thus the error detection capability in exchange for storage room or bandwidth. Algorithm 18 shows pseudocode for the reconstruction of the corresponding object from a compact SCC. Once the last movement is known, the meaning of the two bits in

Algorithm 18 Shape reconstruction from SCC *Code*

$d \leftarrow 0$
$x \leftarrow x_\mathrm{i}$, $y \leftarrow y_\mathrm{i}$
do
  extract c from $Code$
  if $d = 0$
    increase x
    if $c = 0$ $d \leftarrow 2$
    else if $c = 1$ $p_{x,y} \leftarrow 1$
    else if $c = 2$ $d \leftarrow 3$
    else if $c = 3$ $\{d \leftarrow 3, p_{x,y-1} \leftarrow 1\}$
  else if $d = 1$
    decrease x
    if $c = 0$ $d \leftarrow 3$
    else if $c = 1$ $\{d \leftarrow 2, p_{x-1,y} \leftarrow 1\}$
    else if $c = 2$ $p_{x-1,y-1} \leftarrow 1$
    else if $c = 3$ $\{d \leftarrow 2, p_{x-1,y} \leftarrow 1\}$
  else if $d = 2$
    increase y
    if $c = 0$ $d \leftarrow 1$
    else if $c = 1$ $\{d \leftarrow 0, p_{x,y} \leftarrow 1\}$
    else if $c = 2$ $p_{x-1,y} \leftarrow 1$
    else if $c = 3$ $\{d \leftarrow 0, p_{x,y} \leftarrow 1\}$
  else
    decrease y
    if $c = 0$ $d \leftarrow 0$
    else if $c = 1$ $p_{x,y-1} \leftarrow 1$
    else if $c = 2$ $d \leftarrow 1$
    else if $c = 3$ $\{d \leftarrow 1, p_{x-1,y-1} \leftarrow 1\}$
while $Code \neq \emptyset$

the chain element is unequivocally determined.

3.3.3 Template matching with SEE chain code

To illustrate the possibilities of the SEE based chain code in computer vision, I performed a simple test for object recognition using Levenshtein [143] and generalized string edit distances [183]. Given two strings X and Y made of symbols from a finite alphabet Σ, $X, Y \in \Sigma^*$, the Levenshtein distance, also known as string edit distance, and the generalized string distance are measures of similarity between X and Y based on the minimum number of elementary edit operations needed to transform X into Y.

The set of elementary edit operations is usually restricted to insertion of a symbol, deletion of a symbol, and substitution of one symbol for another (transposition and substring manipulation are also considered by some authors), denoted by pairs $(a \rightarrow b) \neq (\varepsilon \rightarrow \varepsilon)$, with $a, b \in \Sigma$ and ε the null string. An insertion is $(a \rightarrow \varepsilon)$, a deletion is $(\varepsilon \rightarrow a)$, both with $a \neq \varepsilon$, and a substitution is $(a \rightarrow b)$ with $a \neq b$. An edit transformation of X into Y is a sequence S of elementary edit operations s_i that transforms X into Y. The Levenshtein distance is the minimum number of edit operations that achieve this. The generalized edit distance $\delta(X, Y)$ considers different weights for each possible edit operation, $\gamma[(a \rightarrow b)]$, $\gamma(S) = \sum \gamma(s_i)$, such that $\delta(X, Y) = \min\{\gamma(S)\}$, the minimum weighted sum of all possible edit transformations taking X to Y. Edit distances fulfill the triangle inequality, and δ is a metric in Σ^* if γ is subjected to: $\gamma[(a \rightarrow a)] = 0$, $\gamma[(a \rightarrow b)] > 0 \; \forall \; a \neq b$, and $\gamma[(a \rightarrow b)] = \gamma[(b \rightarrow a)] \; \forall \; a, b \in \Sigma \cup \varepsilon$.

String edit distances have been used for matching of protein and DNA sequences [218], for text retrieval and correction [273], speech recognition [128], and other pattern matching applications, including computer vision applications [43, 44, 51, 83, 166]. See also the references in [183]. An $O(|X| \times |Y|)$ algorithm using dynamic programming for the computation of the (generalized) edit distance has been independently discovered by several authors [184, 217, 221, 256, 257] even if in the pattern recognition literature it is usually attributed to Wagner and Fischer [257]. It uses the recursive relation $\delta(X_{1\ldots i}, Y_{1\ldots j}) = \min\{\delta(X_{1\ldots i-1}, Y_{1\ldots j}) + \gamma[(X_i \rightarrow \varepsilon)], \delta(X_{1\ldots i-1}, Y_{1\ldots j-1}) + \gamma[(X_i \rightarrow Y_j)], \delta(X_{1\ldots i}, Y_{1\ldots j}) + \gamma[(\varepsilon \rightarrow Y_j)]\}$ to iteratively construct a distance

matrix whose $(|X|, |Y|)$ element is the edit distance from X to Y. Since then, new algorithms have been developed to reduce the worst case or the average case complexity [48,49,64,86,140,181,251], of use in time critical applications, such as data base retrieval. However, the tests that follow do not have tight time constraints nor involve really long sequences, so the original dynamic programming $O(|X| \times |Y|)$ algorithm is used.

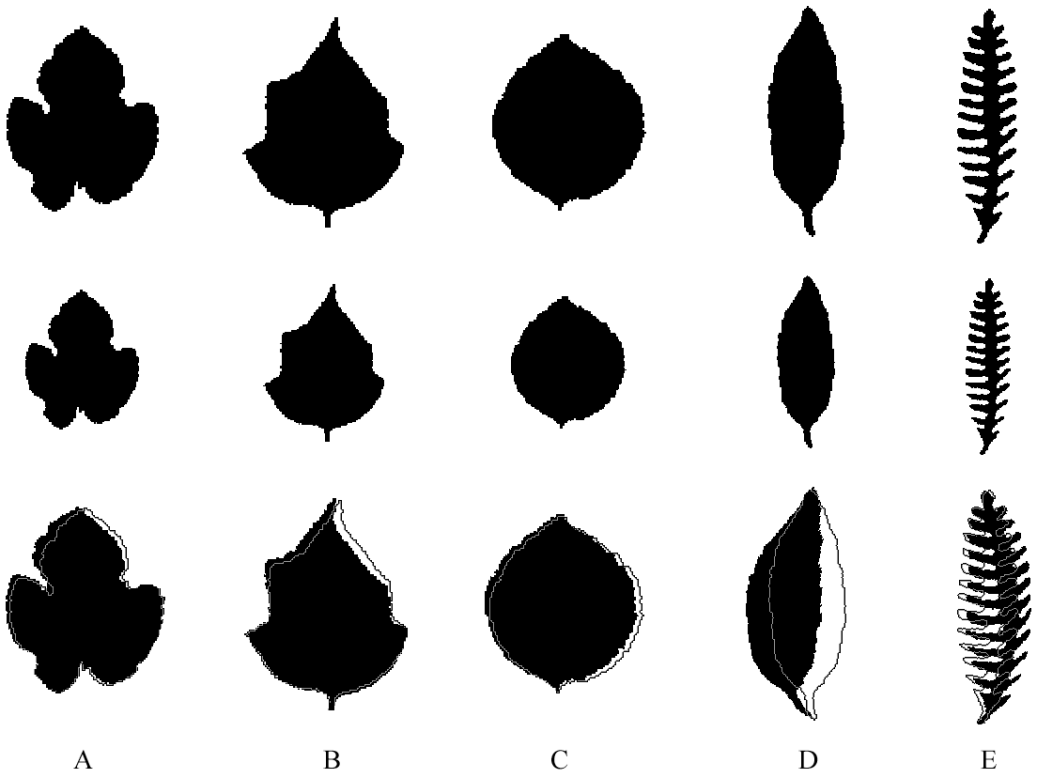


Figure 3.4. *Digitized silhouettes of five plant and tree leaves: A) Fig tree, B) Poinsettia, C) Birch tree, D) Lemon tree, E) Fern. Upper row: Originals from [34]. Middle row: Original shapes downscaled to 75%. Lower row: Original shapes slightly warped. The original shapes are outlined on the warped images.*

Digital shapes can be represented by chain codes, which are sequences of symbols from small alphabets ($|\Sigma_{\text{Crack}}| = 4$, $|\Sigma_{\text{VCC}}| = 3$, and $|\Sigma_{\text{SCC}}| = 4$), thus allowing the use of string edit distances as measures of similarity for pattern matching and object recognition purposes [43, 44, 51]. Figure 3.4 shows the digitized silhouettes of five plant and tree leaves used by Bribiesca in his

work on the vertex chain code [34]. The middle row shows the original silhouettes downsampled to 75% of their original size, and the lower row shows slightly (and not so slightly) nonlinearly transformed (warped) versions. I computed the crack chain code (CCC), the vertex chain code (VCC), and the compact SEE chain code (SCC) of all shapes in Figure 3.4, and computed all pairwise Levenshtein and generalized edit distances between them. The weights selected for the generalized string distance try to reflect the magnitude of the direction change caused by each possible symbol substitution, unitary cost for 90° turns, double cost for 180° turns, while deletions and insertions were assigned unitary cost. For the Levenshtein distance all costs are set to one.

The tables that follow show some results. Comparisons are made on the individual performance of each coding scheme for pattern matching. For direct matching, all distances are expressed as percentage of the maximum distance in the table for each different codification scheme, to ensure that results are independent of code specificities, such as number of symbols in the alphabet. Therefore, 0% represents the least similar pair in the table for each code, whereas 100% indicates two exactly matching sequences. Intermediate values for each coding scheme represent intermediate degrees of similarity between total match and biggest dissimilarity.

Table 3.2 shows the results of pairwise matching between the original shapes. There is of course a perfect match with each shape with itself for all coding schemes. There is a general agreement among codes on the similarities between the different leaves, with a few exceptions: VCC finds the Fig leaf (A) more similar to the Poinsettia (B), whereas CCC and SCC find it to be closer to the Birch leaf (C). The three of them clearly consider[3] the Fern leaf (E) the least similar to any other, specially the Lemon leaf (D), but while CCC and VCC find a slightly higher similarity with the Fig (A) and the Poinsettia (B) leaves, respectively, SCC just finds it very different from any of the other four leaves. The three agree on the Poinsettia (B) being closest to the Birch leaf (C), the Birch leaf (C) being closest to the Poinsettia (B),

[3]Shape comparison with chain codes is based exclusively on local information about the contours.

CCC	**A**	**B**	**C**	**D**	**E**
A	**100**%	45%	48%	41%	6%
B	45%	**100**%	60%	53%	3%
C	48%	60%	**100**%	60%	5%
D	41%	53%	60%	**100**%	0%
E	6%	3%	5%	0%	**100**%
VCC	**A**	**B**	**C**	**D**	**E**
A	**100**%	49%	45%	28%	5%
B	49%	**100**%	53%	38%	6%
C	45%	53%	**100**%	46%	4%
D	28%	38%	46%	**100**%	0%
E	5%	6%	4%	0%	**100**%
SCC	**A**	**B**	**C**	**D**	**E**
A	**100**%	45%	47%	33%	1%
B	45%	**100**%	54%	44%	2%
C	47%	54%	**100**%	51%	2%
D	33%	44%	51%	**100**%	0%
E	1%	2%	2%	0%	**100**%

Table 3.2. *Direct matching with Levenshtein distance between leaf shapes in upper row of Figure 3.4. Values indicate relative match, from 100% for exact match (zero distance) down to 0% for worst match (maximum distance in each table).*

and the Lemon leaf (D) to the Birch leaf (C). In the disagreements between VCC and CCC, SCC seems to take an intermediate position, for instance in the decisions about second-closest pairs.

Table 3.3 shows the same results using the generalized edit distance. The three encoding schemes show little change with respect to cost assignation. The change is not enough to alter the general pattern of similarity, which remains about the same. This was verified in all the tests that follow. Therefore, to avoid redundancy, only the results with the Levenshtein distance are shown hereafter.

Table 3.4 shows the results of matching the small downscaled shapes to the original shapes. The table shows both direct matching using the distances (normalized within table as in the previous tables) and through a voting scheme, where the candidate templates $T_i \in \{\mathrm{A}, \mathrm{B}, \mathrm{C}, \mathrm{D}, \mathrm{E}\}$ receive

CCC	**A**	**B**	**C**	**D**	**E**
A	**100**%	43%	51%	45%	2%
B	43%	**100**%	62%	56%	0%
C	51%	62%	**100**%	63%	5%
D	45%	56%	63%	**100**%	2%
E	2%	0%	5%	2%	**100**%
VCC	**A**	**B**	**C**	**D**	**E**
A	**100**%	47%	44%	28%	5%
B	47%	**100**%	52%	38%	6%
C	44%	52%	**100**%	46%	4%
D	28%	38%	46%	**100**%	0%
E	5%	6%	4%	0%	**100**%
SCC	**A**	**B**	**C**	**D**	**E**
A	**100**%	44%	47%	33%	1%
B	44%	**100**%	54%	44%	2%
C	47%	54%	**100**%	51%	2%
D	33%	44%	51%	**100**%	0%
E	1%	2%	2%	0%	**100**%

Table 3.3. *Direct matching with generalized edit distance between leaf shapes in upper row of Figure 3.4. Values indicate relative match, from 100% for exact match (zero distance) down to 0% for worst match (maximum distance in each table).*

a number of votes v_i according to the Levenshtein distance to the pattern being matched, $P \in \{\mathrm{a}, \mathrm{b}, \mathrm{c}, \mathrm{d}, \mathrm{e}\}$, $v_i = \max\{\delta(P, T_j)\}_j - \delta(P, T_i)$. Votes in the table are expressed in percentage over total votes received by all candidate templates for each pattern.

The three coding schemes assign the correct template to each pattern, but some differences are shown. In direct matching, CCC shows the highest winner scores, while VCC shows the lower ones, and SCC is in the middle, closer to CCC than to VCC. However, the high winners may hide the fact that distances to second and third best matches are also higher in CCC, thus hinting at a lesser sensitivity, or broader response curve, which may affect the performance of the classifier. The clearest example in Table 3.4 occurs when matching the downscaled Fern leaf (e), where CCC shows the strongest direct response of the three encoding schemes, but the winner (E) gets 2.33

Direct						Votation				
A	**B**	**C**	**D**	**E**	CCC	**A**	**B**	**C**	**D**	**E**
65%	46%	57%	58%	12%	**a**	**30**%	19%	26%	26%	0%
37%	**68**%	62%	64%	7%	**b**	15%	**30**%	27%	28%	0%
36%	47%	**71**%	61%	6%	**c**	16%	21%	**34**%	29%	0%
33%	44%	57%	**72**%	0%	**d**	16%	22%	28%	**35**%	0%
13%	15%	15%	12%	**35**%	**e**	2%	10%	10%	0%	**78**%
A	**B**	**C**	**D**	**E**	VCC	**A**	**B**	**C**	**D**	**E**
55%	44%	55%	54%	19%	**a**	**27**%	19%	27%	27%	0%
32%	**59**%	53%	55%	15%	**b**	12%	**31**%	27%	29%	0%
23%	34%	**63**%	52%	5%	**c**	12%	19%	**38**%	31%	0%
20%	31%	46%	**64**%	0%	**d**	12%	19%	29%	**40**%	0%
22%	21%	18%	18%	**29**%	**e**	22%	17%	2%	0%	**58**%
A	**B**	**C**	**D**	**E**	SCC	**A**	**B**	**C**	**D**	**E**
62%	42%	52%	49%	13%	**a**	**32**%	19%	25%	24%	0%
34%	**65**%	53%	54%	10%	**b**	15%	**33**%	26%	26%	0%
30%	40%	**69**%	52%	6%	**c**	15%	20%	**38**%	28%	0%
27%	38%	50%	**69**%	0%	**d**	15%	20%	27%	**38**%	0%
9%	9%	10%	11%	**29**%	**e**	0%	0%	2%	10%	**88**%

Table 3.4. *Matching with Levenshtein distance the downscaled shapes (small letters) to the original leaf shapes (capital letters) in Figure 3.4. Left: Direct matching, where values indicate relative match, from 100% for exact match (zero distance) down to 0% for worst match (maximum distance in each table). Right: Votation, where values indicate percentage of votes for each original shape as candidate for matching the corresponding downscaled shapes. Bold: Winners.*

times the response of the second closest (B), while with SCC the ratio grows to 2.64 times. The same behaviour can be seen throughout the whole table.

This is clearly revealed by the voting scheme, where SCC clearly shows the best performance, still exhibiting a narrower response to the best match. VCC shows the worst performance with very wide responsivity, such that the downscaled Fig leaf (a), is matched with almost equal strength to the Fig leaf, the Birch leaf, and the Lemon leaf (A, C, and D), whereas CCC shows a similar behaviour, if not so markedly, and SCC keeps a healthy 7 points between the winner, the Fig leaf (A), and the second closest, the Birch leaf (C). This also happens with the matching of the downscaled Poinsettia (b).

Direct						Votation				
A	**B**	**C**	**D**	**E**	CCC	**A**	**B**	**C**	**D**	**E**
89%	39%	46%	40%	1%	***A***	**42**%	18%	21%	19%	0%
40%	**86**%	54%	50%	0%	***B***	17%	**37**%	24%	22%	0%
47%	60%	**91**%	61%	5%	***C***	18%	23%	**36**%	23%	0%
38%	49%	54%	**75**%	3%	***D***	17%	23%	25%	**36**%	0%
2%	3%	6%	6%	**67**%	***E***	0%	0%	5%	5%	**91**%
A	**B**	**C**	**D**	**E**	VCC	**A**	**B**	**C**	**D**	**E**
75%	39%	34%	25%	13%	***A***	**51**%	21%	17%	10%	0%
41%	**71**%	40%	33%	16%	***B***	20%	**45**%	20%	14%	0%
43%	47%	**80**%	45%	13%	***C***	18%	21%	**41**%	20%	0%
19%	24%	31%	**46**%	14%	***D***	8%	17%	27%	**49**%	0%
5%	5%	0%	3%	**36**%	***E***	10%	10%	0%	6%	**73**%
A	**B**	**C**	**D**	**E**	SCC	**A**	**B**	**C**	**D**	**E**
81%	39%	38%	30%	6%	***A***	**45**%	20%	20%	14%	0%
38%	**78**%	44%	40%	10%	***B***	17%	**42**%	22%	19%	0%
43%	49%	**84**%	49%	9%	***C***	18%	21%	**40**%	21%	0%
25%	33%	40%	**58**%	14%	***D***	11%	19%	26%	**44**%	0%
3%	2%	2%	0%	**47**%	***E***	6%	4%	4%	0%	**87**%

Table 3.5. *Matching with Levenshtein distance the warped shapes (slanted letters) to the original leaf shapes (capital letters) in Figure 3.4. Left: Direct matching, where values indicate relative match, from 100% for exact match (zero distance) down to 0% for worst match (maximum distance in each table). Right: Votation, where values indicate percentage of votes for each original shape as candidate for matching the corresponding warped shapes. Bold: Winners.*

Table 3.5 shows the same results for the matching of the warped leaves with the original leaves. The behaviour is very similar. The three encoding schemes are able to find the correct template both by direct matching and by vote, CCC shows the strongest direct responses but also a broader responsivity than SCC, while VCC is the worst performer, if not bad. Again with the voting scheme SCC reveals the power of its narrower sensitivity — the warped Fern leaf (E) being the exception—, but now even VCC is able to grow from its poor direct matching performance showing a better voting behaviour than CCC due to a narrower sensitivity. With the warped images there are no border cases, all second-best matches are reasonably far from the best matches with the three encoding schemes.

Direct						Votation				
A	***B***	***C***	***D***	***E***	CCC	***A***	***B***	***C***	***D***	***E***
57%	39%	53%	47%	11%	**a**	**30**%	18%	27%	24%	0%
32%	57%	**58**%	52%	8%	**b**	14%	29%	**30**%	27%	0%
29%	38%	**65**%	47%	5%	**c**	15%	20%	**38**%	27%	0%
26%	35%	52%	**54**%	0%	**d**	16%	21%	31%	**32**%	0%
12%	14%	14%	14%	**28**%	**e**	0%	12%	9%	11%	**68**%
A	***B***	***C***	***D***	***E***	VCC	***A***	***B***	***C***	***D***	***E***
45%	39%	**52**%	40%	17%	**a**	26%	20%	**32**%	22%	0%
28%	44%	**48**%	39%	14%	**b**	14%	29%	**33**%	24%	0%
17%	26%	**55**%	38%	4%	**c**	11%	18%	**42**%	28%	0%
14%	23%	**42**%	38%	0%	**d**	12%	20%	**36**%	33%	0%
26%	26%	23%	24%	**28**%	**e**	22%	25%	0%	12%	**41**%
A	***B***	***C***	***D***	***E***	SCC	***A***	***B***	***C***	***D***	***E***
55%	39%	50%	41%	14%	**a**	**31**%	19%	28%	21%	0%
32%	**53**%	50%	42%	10%	**b**	16%	**31**%	29%	24%	0%
28%	34%	**63**%	44%	6%	**c**	15%	20%	**39**%	26%	0%
24%	32%	**48**%	48%	0%	**d**	16%	21%	**32**%	31%	0%
14%	18%	15%	22%	**26**%	**e**	0%	14%	4%	33%	**49**%

Table 3.6. *Matching with Levenshtein distance the downscaled shapes (small letters) to the warped leaf shapes (slanted letters) in Figure 3.4. Left: Direct matching, where values indicate relative match, from 100% for exact match (zero distance) down to 0% for worst match (maximum distance in each table). Right: Votation, where values indicate percentage of votes for each warped shape as candidate for matching the corresponding downscaled shapes. Bold: Winners.*

Table 3.6 shows a more difficult scenario. The downscaled leaf silhouettes are matched to the warped templates. This poses the hardest challenge, as the results show. None of the encoding schemes is able to find the correct winner in all cases. CCC and SCC show the best performance, just a slight difference causes one error —SCC incorrectly assigned the small Lemon leaf (e) to the warped Birch leaf (C) for less than one point, and CCC incorrectly assigned the small Poinsettia leaf (b) to the warped Birch leaf (C) for one point, and was on the verge of making the same mistake as SECC with the Lemon leaf (e)—, whereas VCC was wrong with three of the patterns, the Fig leaf, the Poinsettia leaf, and the Lemon leaf (a, b, and d), all incorrectly assigned to the warped Birch leaf (C). The usual confidence of SCC on its

Direct						Votation				
a	**b**	**c**	**d**	**e**	CCC	**a**	**b**	**c**	**d**	**e**
53%	25%	23%	19%	3%	***A***	**46**%	20%	18%	15%	0%
33%	**53**%	32%	29%	6%	***B***	22%	**38**%	21%	19%	0%
48%	54%	**62**%	48%	6%	***C***	23%	25%	**30**%	22%	0%
42%	47%	43%	**49**%	6%	***D***	23%	26%	23%	**28**%	0%
2%	0%	4%	10%	**21**%	***E***	6%	0%	11%	27%	**56**%
a	**b**	**c**	**d**	**e**	VCC	**a**	**b**	**c**	**d**	**e**
45%	28%	17%	14%	26%	***A***	**51**%	23%	6%	0%	20%
39%	**44**%	26%	23%	26%	***B***	38%	**49**%	7%	0%	7%
52%	48%	**55**%	42%	23%	***C***	27%	24%	**31**%	18%	0%
40%	39%	38%	38%	24%	***D***	**27**%	25%	24%	23%	0%
17%	14%	4%	0%	**28**%	***E***	27%	22%	7%	0%	**45**%
a	**b**	**c**	**d**	**e**	SCC	**a**	**b**	**c**	**d**	**e**
49%	24%	20%	15%	5%	***A***	**50**%	22%	17%	12%	0%
32%	**48**%	27%	25%	9%	***B***	24%	**40**%	19%	17%	0%
45%	44%	**59**%	42%	6%	***C***	23%	23%	**32**%	22%	0%
34%	36%	37%	**42**%	14%	***D***	22%	23%	25%	**30**%	0%
4%	0%	0%	6%	**18**%	***E***	14%	1%	0%	22%	**63**%

Table 3.7. *Matching with Levenshtein distance the warped shapes (slanted letters) to the downscaled leaf shapes (small letters) in Figure 3.4. Left: Direct matching, where values indicate relative match, from 100% for exact match (zero distance) down to 0% for worst match (maximum distance in each table). Right: Votation, where values indicate percentage of votes for each downscaled shape as candidate for matching the corresponding warped shapes. Bold: Winners.*

responses, revealed by its narrower responsivity, is not such in this case, where it shows response curves as wide as CCC. However, overall it showed the best performance, very close to that of CCC, also in this case. Again VCC was third in the ranking.

Last, table 3.7 shows the reverse, the warped silhouettes matched to the downscaled images. This time two of the three encoding schemes are able to solve the riddle without errors. VCC is again the worst performer, including one error where the correct template (d) is not even second best, but fourth —out of five— for a pattern (warped Lemon leaf, *D*) where SCC yields the correct template without much hesitation —five points of difference to the

second best match—, and CCC also gives the correct result, even if with less confidence —two points above to the second best. SCC is again the best performer, with narrower sensitivity and good overall accuracy.

Therefore, Stöhr Edge Encoding offers a new chain code that does not need a contour following algorithm other than the SEC codes themselves, and in rotation invariant environments gives at least similar performance for template matching with string edit distances as the classical crack code (4-neighbourhood Freeman chain code) and the recent vertex chain code, if not better.

3.3.4 A 3D chain code

SEE works in any number of dimensions, including, naturally, three dimensional images. In 3D, nodes are surrounded by eight voxels, and the number of different possible local configurations is 256. Assigning the first eight elements of the 8-Stöhr sequence, $\{1, 2, 4, 8, 16, 32, 64, 128\}$ to the eight vertices of each voxel does the trick, see Figure 3.5. As the interested reader will have already noted, SEE codes provide a binary codification of the local environment of each node, and thus SEE codes can also be viewed as small subimages containing small neighbourhoods, only if in a scan order generally differing from the usual raster order. Each bit in a SEC indicates the state of a given pixel in the neighbourhood of the corresponding node in the original image. 2D SEE chain coding just concatenates the (innovative part of the) SEC codes according to a preestablished contour following scheme, with the advantage that the SEE codes themselves provide the information about the direction to follow.

There is no reason to not be able to extend this idea to 3D SEC and images. Well, a reason may not exist, but a practical issue certainly arises: 2D contours can be unambiguously followed once a turning direction has been chosen, but 3D surfaces enclosing 3D shapes are not that straightforwardly scanned such that all pixels in the boundary are visited once and only once according to a predictable, repeatable scheme. This is where hybrid labelling (Chapter 2) reenters the stage, under the shape of hybrid shape boundary traversal. Hybrid boundary traversal (a backscan, a forward scan, followed by a second forward scan inspecting neighbour nodes in the transverse directions) provides a boundary following scheme permitting to scan all boundary nodes in a 3D shape, such that all (the innovative parts of) their SEE codes can be concatenated into a chain code that will ultimately hold the whole information about the shape in a compact way, as much as it happens with the usual 2D chain codes.

Up to the knowledge of the author, no chain code for 3D shapes has been published yet. Certainly, the traditional Freeman chain code has been straightforwardly extended to allow displacements out of the plane, thus

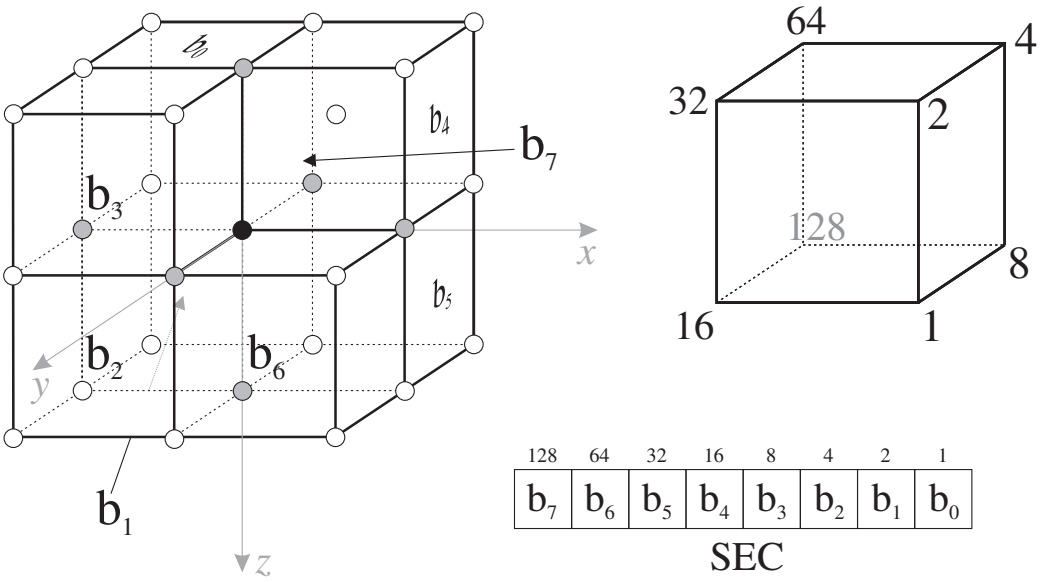


Figure 3.5. *SEE in 3D: (left) A node (black dot) surrounded by 8 voxels and 6 adjacent nodes (grayed dots). Each foreground voxel is represented by a 1 in the indicated bit (b_i) of the SEE codeword (SEC); (right) Assignment of weights to voxel vertices, and below the correspondence between bits in the SEC, weights in the voxel vertices, and voxels surrounding the node. For convenience, the central node is referred as node (x, y, z), and the voxel labelled as b_6 is referred as voxel (x, y, z). From this, a relationship between image coordinates and nodes can be easily established.*

permitting the codification of *curves* in 3D space, just by increasing the number of codes to represent the new directions, but the principles underlying the method are intrinsically the same as that of the 2D chain code. Only Bibriesca, in his presentation of the vertex chain code [34], introduced a single sentence suggesting further work to develop a chain code for 3D shapes, but no news on any development in this sense have since appeared. However, a compact representation for arbitrary 3D shapes similar to that of 2D chain codes may prove useful in many different applications, such as archiving of features of interest in medical images for assessment in time series —tumors in tomographic images, for instance, where the evolution of shape and size could then be monitored independently of the position in each successive tomography without the need to store or analyse the whole tomographies each time.

It is possible to generate a very simple 3D chain code just by applying the 3D hybrid labelling algorithm described in Chapter 2 to the outer voxels of an arbitrary shape: Hybrid labelling is able to move in any of 6 possible directions from a pixel to any of the adjacent pixels. These can be encoded with three bits, say 1 for x^- (backscan, in the terminology of hybrid labelling), 2 for x^+ (forward scan), 3 for y^-, 4 for y^+, 5 for z^-, and 6 for z^+ (all of them within the second forward scan). Suppose we scan all boundary voxels of an arbitrary 3D shape with hybrid labelling, while storing into a chain the 3 bits that encode the direction of each movement. Now let us try to reconstruct the shape with the stored chain. We enter the first burst of voxels in the surface of the shape, and scan forward until the burst of 2's in the chain is finished. Then a 4 follows. This means we have to move a step in y^+. However, from which voxel in the burst? Clearly, we have lost some crucial information. We need to keep track in the encoding also of the movement along the burst during the second forward scan, so we can know later, during the reconstruction, from which position in the burst are we expected to take any of the traverse directions indicated in the chain. In three bits there is still room for two more codes, 0 and 7, so this is not going to cause adverse effects on the size of each code in the chain. We may use code 7 to indicate each step forward along the burst in the second forward scan. Let us go back to the end of the very first burst during the reconstruction. After the burst of 2's, we find two 7's, and then the code 4. This means we have to count two voxels from the beginning of the burst, and then jump in y^+. Now it works. However, whereas the size of each code in the chain is not affected, the total length of the chain certainly is. Now we need two codes per boundary voxel, one to enter them and the other to keep track of the position of potential turns in the second forward scan. Also note that to be able to scan the whole surface of any arbitrary 3D shape we have to use 8-neighbourhood connectivity: Just imagine a voxel sitting on top of a flat surface of voxels. To be able to jump from the upper voxel to the surface below visiting only *boundary* nodes, we need a diagonal jump, or we will have to inevitably go through the voxel right below the upper voxel, which is not a boundary voxel. This increases even more the length of the chain, because the bursts in the forward scan become two voxels longer. Therefore, voxel

hybrid scanning produces the first real 3D chain code described up to date, but it somehow causes certain intellectual discomfort.

How to preserve the topology of the surface without having to use all this duplicity of codes? The answer is Stöhr Edge Encoding.

Consider the SEE weight assignation of Figure 3.5. All SEC different from 0 and 255 (FF_h) are nodes on a shape boundary, i.e. they belong to facets separating foreground voxels (voxels obeying $\mathcal{A}$, see 2.1.1) from background voxels (voxels not obeying $\mathcal{A}$). Our aim is to scan all the boundary nodes in a shape according to a predefined order, such that all nodes in the boundary of any arbitrary 3D shape are scanned once and only once. During the scan, we will append the (innovative part of the) SEC of each node to a chain code, very much in the way of the compact 2D SCC. The chain code thus constructed can later be decoded to reconstruct the boundary of the original shape, just by rebuilding the local neighbourhood of the boundary nodes using the SEE information contained in the chain.

For this purpose, the same scanning scheme is used in the coding and decoding of the chain, hybrid boundary traversal (HBT), which is just an adaptation of the hybrid labelling algorithm to scan nodes instead of pixels/voxels, using boundary bond connectivity. By boundary bond connectivity I refer to the following connectivity criterion: Two adjacent nodes are connected if the edge linking them belongs to both at least a background voxel and a foreground voxel (i.e. a boundary edge). Thus we are allowed to go from a node to one of the adjacent nodes if in the set of the four voxels which they share there is at least one foreground and one background voxel. This information is contained in the SEC of any of the nodes under inspection, the source node and the potential destination node, and it is all the information hybrid boundary traversal needs.

Algorithm 19 shows pseudocode for constructing the 3D SEE chain code of any arbitrary shape using hybrid boundary traversal. The assumptions are that the 3D image has already been processed to discriminate the voxels obeying $\mathcal{A}$ from those not, and then the SEC of all nodes in the image have been computed (see Algorithm 10).

Algorithm 19 3D SEE (Hybrid boundary traversal) chain code

initialize $Code$
scan all x, y, z **until** $c_{x,y,z} \neq 0$
if $c_{x,y,z} \neq 0$ Encode(x, y, z)

Encode(x, y, z)
    $m \leftarrow x$
    $b \leftarrow c_{x,y,z}$
    while $0 < b_3b_2b_1b_0 < 0\mathrm{F_h}$ {**decrease** x, $b \leftarrow c_{x,y,z} \leftarrow 0$, $b_3b_2b_1b_0 \Rightarrow Code$}
    $b \leftarrow c_{m,y,z} \leftarrow 0$
    while $0 < b_7b_6b_5b_4 < 0\mathrm{F_h}$ {**increase** m, $b \leftarrow c_{m,y,z} \leftarrow 0$, $b_7b_6b_5b_4 \Rightarrow Code$}
    do
        $b \leftarrow c_{x,y-1,z}$
        if $0 < b_7b_6b_3b_2 < 0\mathrm{F_h}$ {$b_5b_4b_1b_0 \Rightarrow Code$, Encode$(x, y-1, z)$}
        $b \leftarrow c_{x,y+1,z}$
        if $0 < b_5b_4b_1b_0 < 0\mathrm{F_h}$ {$b_7b_6b_3b_2 \Rightarrow Code$, Encode$(x, y+1, z)$}
        $b \leftarrow c_{x,y,z-1}$
        if $0 < b_6b_5b_2b_1 < 0\mathrm{F_h}$ {$b_7b_4b_3b_0 \Rightarrow Code$, Encode$(x, y, z-1)$}
        $b \leftarrow c_{x,y,z+1}$
        if $0 < b_7b_4b_3b_0 < 0\mathrm{F_h}$ {$b_6b_5b_2b_1 \Rightarrow Code$, Encode$(x, y, z+1)$}
        increase x
    while $x \leq m$
end of Encode

A comparison with Algorithm 4 (section 2.2) will easily reveal how closely the hybrid boundary transversal algorithm underlying the encoding scheme is related to hybrid labelling. Instead of labelling voxels, now the SEC of the visited nodes are set to zero, after their innovative part has been extracted to be added to the chain. Also the mere inspection of the voxels has been replaced by the condition of boundary bond connectivity, which requires at least one foreground and one background voxel in the four voxels sharing the edge under inspection. Note that nibbles (half a byte) to be added to the chain are denoted as streams of four bits selected from a byte (a whole 3D SEC), where the subscripts indicate the position of the bits in the original byte. The expression $b_5b_4b_1b_0 \Rightarrow Code$ denotes bits 0, 1, 4, and 5 to be

added to the chain ($Code$), the l.s.b. on the right. The SEC of node (x, y, z) is denoted by $c_{x,y,z}$.

Hybrid boundary traversal in Algorithm 19 can also be used for fast extraction of the outer surface of any arbitrary 3D shape, just by deleting all references to $Code$.

Algorithm 20 shows pseudocode for the reconstruction of the shape from the corresponding 3D SEE chain code, using hybrid boundary traversal. The structure of the algorithm is the same of Algorithm 19. It goes through all boundary nodes using hybrid boundary traversal, with the peculiarity that it builds the SEC of the traversed nodes on the fly, using information from the previous nodes and the current nibble in the chain. At the same time, it uses the SEC of the current node to find its way along the boundary surface according to hybrid boundary traversal. The notation is the same as above, except that now nibbles are extracted from the chain ($c \Leftarrow Code$), and the SEC are built from streams of eight bits, coming from the current chain nibble and the SEC of a neighbouring node (denoted b).

All SEC are initialized to zero, such that later they can be used to avoid revisiting nodes. Thus the condition of entry for reconstruction with hybrid boundary traversal is that the candidate node to be entered has SEC zero and the edge linking it to the current node is a boundary edge, i.e. it belongs to at least a foreground and a background voxel. Every time a new node is visited (and only new nodes are visited) the four pixels corresponding to the innovation in its SEC (the current nibble in the chain) are set, according to the information in the nibble. Which pixels are these, depends on which direction are we entering the node from. The edge linking the previous node to the new node defines the direction, the pixels to be set are those sharing the next edge along that direction. The correspondence between the pixels and the nibble bits is given by the SEE weight assignment (see Figure 3.5 for the assignment corresponding to Algorithm 20).

Note that hybrid boundary traversal in spite of its recursive nature maintains all the advantages of hybrid labelling regarding stack use (see end of section 2.4.2 for performance data), with the advantage that only the surface

Algorithm 20 Reconstruction from a 3D SCC (Hybrid boundary traversal)

set all $c_{x,y,z}$ to 0
$p_{x_0,y_0,z_0} \leftarrow 1$, $c_{x_0,y_0,z_0} \leftarrow 40_\mathrm{h}$
Decode(x_0, y_0, z_0)

Decode(x, y, z)
  $m \leftarrow x$
  $b \leftarrow c_{x,y,z}$
  while $0 < b_3b_2b_1b_0 < 0\mathrm{F}_\mathrm{h}$
    decrease x
    $c \Leftarrow Code$, $c_{x,y,z} \leftarrow b_3b_2b_1b_0c_3c_2c_1c_0$
    $p_{x-1,y-1,z-1} \leftarrow c_0, p_{x-1,y-1,z} \leftarrow c_1, p_{x-1,y,z} \leftarrow c_2, p_{x-1,y,z-1} \leftarrow c_3$
  $b \leftarrow c_{m,y,z}$
  while $0 < b_7b_6b_5b_4 < 0\mathrm{F}_\mathrm{h}$
    increase m
    $c \Leftarrow Code$, $c_{m,y,z} \leftarrow c_3c_2c_1c_0b_7b_6b_5b_4$
    $p_{m,y-1,z-1} \leftarrow c_0, p_{m,y-1,z} \leftarrow c_1, p_{m,y,z} \leftarrow c_2, p_{m,y,z-1} \leftarrow c_3$
  do
    $b \leftarrow c_{x,y,z}$
    if $(c_{z,y-1,z} = 0)$ and $(0 < b_5b_4b_1b_0 < 0\mathrm{F}_\mathrm{h})$
      $c \Leftarrow Code$, $c_{x,y-1,z} \leftarrow b_4b_5c_3c_2b_0b_1c_1c_0$
      $p_{x-1,y-2,z-1} \leftarrow c_0, p_{x-1,y-2,z} \leftarrow c_1, p_{x,y-2,z-1} \leftarrow c_2, p_{x,y-2,z} \leftarrow c_3$
      Decode$(x, y-1, z)$
    if $(c_{z,y+1,z} = 0)$ and $(0 < b_7b_6b_3b_2 < 0\mathrm{F}_\mathrm{h})$
      $c \Leftarrow Code$, $c_{x,y+1,z} \leftarrow c_3c_2b_6b_7c_1c_0b_2b_3$
      $p_{x-1,y+1,z} \leftarrow c_0, p_{x-1,y+1,z-1} \leftarrow c_1, p_{x,y+1,z} \leftarrow c_2, p_{x,y+1,z-1} \leftarrow c_3$
      Decode$(x, y+1, z)$
    if $(c_{z,y,z-1} = 0)$ and $(0 < b_7b_4b_3b_0 < 0\mathrm{F}_\mathrm{h})$
      $c \Leftarrow Code$, $c_{x,y,z-1} \leftarrow c_3b_7b_4c_2c_1b_3b_0c_0$
      $p_{x-1,y-1,z-2} \leftarrow c_0, p_{x-1,y,z-2} \leftarrow c_1, p_{x,y-1,z-2} \leftarrow c_2, p_{x,y,z-2} \leftarrow c_3$
      Decode$(x, y, z-1)$
    if $(c_{z,y,z+1} = 0)$ and $(0 < b_6b_5b_2b_1 < 0\mathrm{F}_\mathrm{h})$
      $c \Leftarrow Code$, $c_{x,y,z+1} \leftarrow b_6c_3c_2b_5b_2c_1c_0b_1$
      $p_{x-1,y-1,z+1} \leftarrow c_0, p_{x-1,y,z+1} \leftarrow c_1, p_{x,y-1,z+1} \leftarrow c_2, p_{x,y,z+1} \leftarrow c_3$
      Decode$(x, y, z+1)$
    increase x
  while $x \leq m$
end of Decode

of the 3D shape is scanned, not the entire volume, therefore further reducing stack use and time. Note also that the auxiliary additional local variables with respect to hybrid labelling (i.e. not m) in Algorithms 19 (b) and 20 (b and c) are there just for readability, and they can be either suppressed or given a global scope, because they do not have to keep their values for each instance of the recursion, and therefore they do not increase stack use.

Also note that this encoding scheme is general, in the sense that it can be extended to any arbitrary number of dimensions, given that both SEE and hybrid boundary traversal can be naturally extended too.

Some chains

To illustrate the description of the method, the HBT codes of a few very simple and —for obvious reasons— small 3D shapes follow. The HBT chain code of a $4 \times 4 \times 4$ cube is (hex):

```
44 40 46 66 04 66 60 46 66 00 22 20 43 33 0C CC 0C CC 08 CC
C0 CC C0 33 30 43 33 0C CC 0C CC 00 88 80 89 99 08 99 90 89
                  99 00 11 10 33 33 33 33 30
```

It is made up of 97 nibbles, i.e. a total length of 388 bits (49 bytes). The same cube without one of the voxels in the middle of its upper face yields the chain:

```
44 40 42 66 04 46 60 46 66 00 22 20 43 33 0C CC 0C CC 08 CC
C0 CC C0 33 30 43 33 0C CC 0C CC 00 88 80 89 99 08 99 90 89
               99 00 11 10 33 33 33 33 3F FF F0
```

made up of 101 nibbles, i.e. 404 bits or 51 bytes. If instead of deleting a voxel we add one on top of the upper face, we get the chain:

```
00 0F 06 60 8E 66 00 44 40 8C CC 0C CC 03 33 00 22 20 86 66
03 33 30 CC C0 CC C0 33 30 43 33 0C CC 0C CC 00 88 80 89 99
               08 99 90 89 99 00 11 10 33 33 30
```

with the same length as the previous one (404 bits). This is explained by the fact that the length of the HBT chain in nibbles is equal to the number of free voxel faces plus one (four times this in bits). A $4 \times 4 \times 4$ cube has $6 \times 16 = 96$ free voxel faces. Deleting one of the voxels from the surface of the cube adds 4 free faces (the lateral faces of the "hole", the old face on the surface already accounted for the free face in the bottom of the hole), making a total of 100 free voxel faces. Whereas placing a new voxel on the surface of the original cube adds 4 free faces (the lateral faces of the "bump", this time it is the free face on top of the bump which was accounted for the face now under the new voxel), i.e. again 100 free faces.

This is a relevant (even if not unexpected at all) characteristic of HBT chains: The length of the chain depends on the surface area of the shape being encoded, not the volume. This is important because, whereas a $4 \times 4 \times 4$ cube can be held into a $4 \times 4 \times 4$ binary matrix, thus occupying just 64 bits, the corresponding chain weights a formidable 388 bits. And this would not speak in favour of HBT chains were it not for the fact that a $100 \times 100 \times 100$ cube fills a $100 \times 100 \times 100$ binary matrix (1 Mbit), whereas the corresponding HBT chain is 240 004 bits long, i.e. 24%, and a $1000 \times 1000 \times 1000$ cube needs 1 Gbit where the corresponding HBT chain uses 24 Mbit, less than 2.5%. In general, the ratio for a $L \times L \times L$ cube is $4 \times (6L^2 + 1)/L^3 = O(6/L)$ which is favourable to the HBT chain for $L > 6$, and decreases with L.

If compared to the obvious alternative, storing a list of the coordinates of the boundary voxels, we get that the number of free voxels in a $L \times L \times L$ cube is $6L^2 - 12L + 8$, while the number of free voxel faces is $6L^2$. For L as low as 10, we get that we would need to store 488 pairs of coordinates, which would make 976 bytes if only one byte was used to store each coordinate, whereas HBT requires 601 nibbles (301 bytes). Things only get better when L grows, because then one byte does not suffice to store each coordinate.

What about the straightforward scheme introduced above, using hybrid labelling on the boundary voxels? For a $10 \times 10 \times 10$ cube, the direct scheme requires 1139 3-bit codes, while HBT uses 601 4-bit codes. This is 3 417 bits versus 2 404 bits, in favour of HBT, as expected. The direct scheme uses

more than two 3-bit codes per boundary voxel, $O(2 \times 3 \times 6L^2)$ in regular cubes, while HBT uses one 4-bit code per boundary voxel, $O(4 \times 6L^2)$.

Therefore, HBT is more efficient in what regards chain length. Moreover, if the shape being encoded is not specially rugged, HBT tends to generate long constant sequences allowing strong lossless compression of the chain. As a simple example, the HBT chain for a $10 \times 10 \times 10$ cube is 601 nibbles long (301 bytes), but it only takes 134 bytes if a run length encoding scheme is used, where chain nibbles are embedded in bytes with the upper four bits indicating the amount of redundancy (up to 15 repetitions):

```
84 00 04 86 00 04 86 00 04 86 00 04 86 00 04 86 00 04 86 00
04 86 00 04 86 00 04 86 00 04 86 10 82 00 04 83 00 8C 00 8C
00 08 8C 00 8C 00 83 00 04 83 00 8C 00 8C 00 08 8C 00 8C 00
83 00 04 83 00 8C 00 8C 00 08 8C 00 8C 00 83 00 04 83 00 8C
00 8C 00 08 8C 00 8C 00 83 00 04 83 00 8C 00 8C 00 00 88 00
08 89 00 08 89 00 08 89 00 08 89 00 08 89 00 08 89 00 08 89
      00 08 89 00 08 89 10 81 00 F3 F3 F3 F3 F3 03 00
```

3.4 Very fast machine vision for controlled environments

When it comes to object inspection in machine vision under given strict, but not unusual, constraints, SEE provides a better tool than chain codes. These constraints are fixed views of rigid objects from a known finite set whose features are preserved after binarization, but this includes, for instance, inline backlit inspection of machined parts, a common machine vision application.

If the SEE codes (SEC) are accumulated into a histogram, the resulting vector can be used for object recognition, defect detection, and pose estimation. Obviously, the SEC histogram is not unique. Different objects may have equal SEC histograms[4]. However, if enough image resolution is used, such that a sufficient dynamic margin in the SEC is assured, and the set of objects to be handled by the application is finite and known, ambiguities can be resolved beforehand and do not pose a serious objection to the use of SEC histograms.

The advantages of SEC histograms as feature vectors for shape-based object inspection are 1) very fast and 2) very simple 3) integer computation. Algorithm 21 shows how to obtain the SEC histogram of an image $p_{x,y}$ at the same time that obtains and stores the SEC of all nodes. It is a subtle modification of Algorithm 10 that takes advantage of the fact that in the computation of the SEE codes, nodes in the upper left corner of pixels (first corner in scan order) are not disturbed by the forthcoming computation with the pixels not yet scanned. The SEC histogram, HSEC, is a vector in $\mathbb{N}^{16}$ ($\mathbb{R}^{16}$ when normalized). SEC 0 denotes a node embedded in the background and thus HSEC_0 measures the extent of background in the image. SEC 15 denotes a node embedded in foreground pixels, and therefore HSEC_{15}

[4]The proof is trivial, SEC histograms do not preserve information about location. The local configurations of two nodes in the contour of the object can be exchanged, thus changing the shape of the object without affecting the SEC histogram. Note, however, that SEC codes are redundant, and the local configuration of a node cannot be changed without affecting any of its neighbouring nodes. This is avoided by exchanging the location of two blocks of more than four pixels in the contour of an object with the same configurations along their borders.

measures the extent of foreground in the image. SEC $\in \{1, \ldots 14\}$ denote contour nodes with different configurations.

Algorithm 21 2D SEC and histogram HSEC

for all (x, y) $0 \rightarrow c_{x,y}$
for all (x, y)
    if $p_{x,y}$obeys $\mathcal{A}$
        $(1 \nrightarrow c_{x,y})$
        $(2 \nrightarrow c_{x+1,y})$
        $(4 \nrightarrow c_{x,y+1})$
        $(8 \nrightarrow c_{x+1,y+1})$
    Increase $\text{HSEC}_{c_{x,y}}$

If only the SEC histogram is needed, there is no need to store the SEC of the image, as the histogram can be computed on the fly, see Algorithm 22. The drawback is that pixels are visited several times. This can be solved by using a temporary array with the size of a row to store partial codes until the next row is scanned, in a memory-speed trade off.

Algorithm 22 2D SEC histogram HSEC without storage of SEC

for all (x, y)
    $0 \rightarrow c$
    if $p_{x,y}$ obeys $\mathcal{A}$ $(1 \nrightarrow c)$
    if $p_{x-1,y}$ obeys $\mathcal{A}$ $(2 \nrightarrow c)$
    if $p_{x,y-1}$ obeys $\mathcal{A}$ $(4 \nrightarrow c)$
    if $p_{x-1,y-1}$ obeys $\mathcal{A}$ $(8 \nrightarrow c)$
    Increase HSEC_c

3.4.1 Some tests

To demonstrate the potential of SEC histograms, I performed several tests with the leaf images in Figure 3.4. First, I trained a multilayer perceptron (MLP) to distinguish the five original leaves (upper row) with different rotation angle $\alpha \in [0°, 45°]$ in steps of 0.5°. The choice of multilayer perceptrons is not restrictive, SEC histograms can be used with any method using feature vectors, for instance k-NN. Anyhow, if the MLP is able to learn to recognize the five different objects for any rotation angle α with the SEC histograms as only input, then the SEC histograms contain enough information for the discrimination.

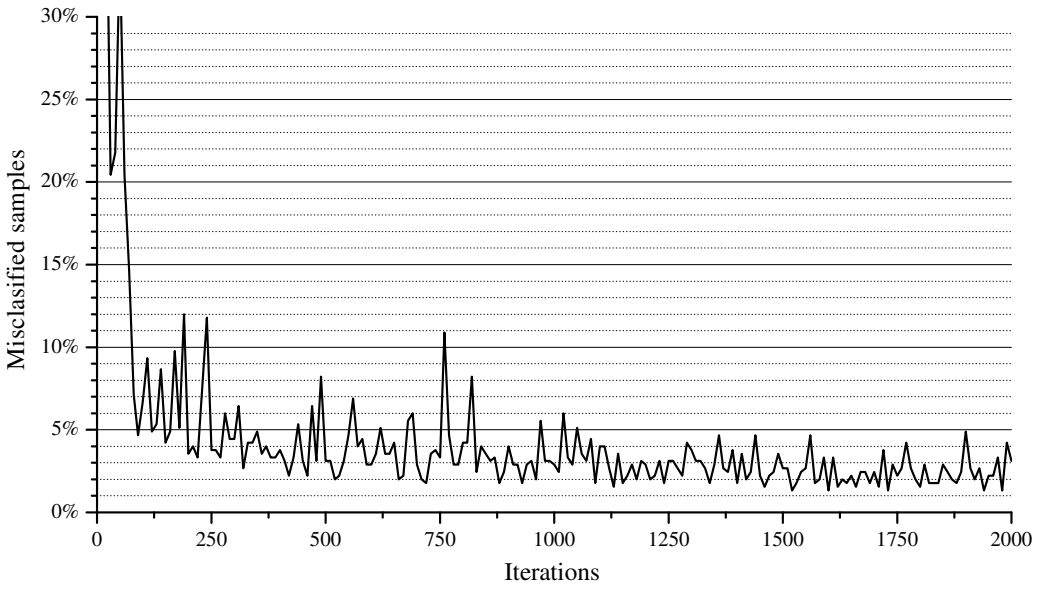


Figure 3.6. *Typical learning curve showing the evolution of the percentage of misclassified samples. The curve shown corresponds to a MLP with 12 hidden nodes trained with momentum 0.2 and learning rate 0.8.*

I trained several two layer MLP with 8 to 12 hidden nodes, 12 input nodes (corresponding to the 12 contour SEC, excluding SEC 6 and 9[5]), and 5 sigmoidal outputs, each one corresponding to each of the leaf types in the upper row of Figure 3.4. The class assigned to the input is that of the higher

[5]All pixels in the leaf images have neighbours in their 4-neighbourhoods, i.e. there have no nodes with configuration type 6 or 9.

output. For the training process I used backpropagation with momentum, in order to attenuate the effect of local minima.

	A	**B**	**C**	**D**	**E**
A	89	1	0	0	0
B	4	86	0	0	0
C	0	0	90	0	0
D	0	0	0	90	0
E	0	0	0	0	90

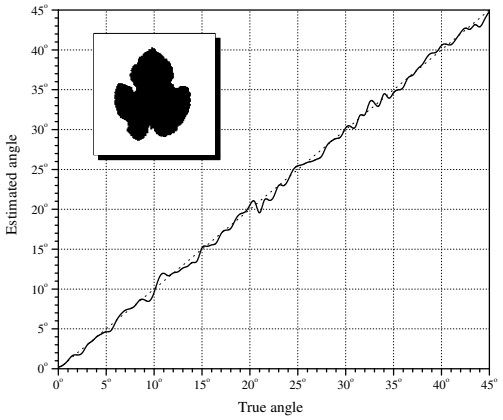
Table 3.8. *Typical confusion matrix in the MLP learning of five leaf shapes in 90 different rotations with their SEC histograms as input.*

The results were fairly similar, with slight variations due to the stochastic nature of the learning process. After 600 to 1 500 iterations in which the SEC histograms of the 90 different rotation angles α for each of the five leaf images were shown to the MLP, see Figure 3.6 for a typical learning curve and Table 3.8 for a typical confusion matrix, all MLP were able to correctly classify 93.4% of the 450 images, with average accuracy 98.7%. The errors were concentrated in the Poinsettia leaf (B), where the MLP used to mistook for a Fig leaf (A) 4 of the 90 differently rotated Poinsettia leaf images (6.67%), and the Fig leaf (A), where the MLP mistook for a Poinsettia leaf (A) 1 to 2 of the 90 different rotated Fig leaf images (1.11% to 2.22%). The Birch (C), Lemon (D), and Fern (E) leaves were correctly classified for all rotation angles α in all tests.

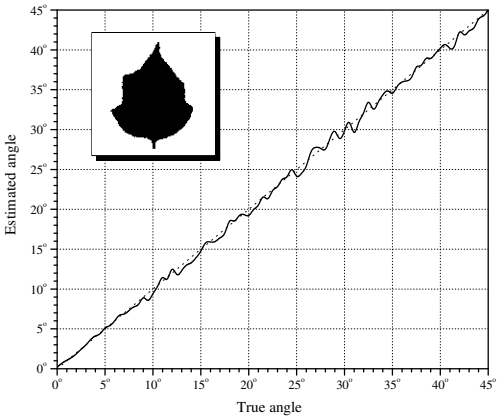
	Error
Fig leaf	$0.13^\circ \pm 0.70^\circ$
Poinsettia leaf	$0.14^\circ \pm 0.72^\circ$
Birch leaf	$0.15^\circ \pm 0.84^\circ$
Lemon leaf	$0.21^\circ \pm 0.72^\circ$
Fern leaf	$0.07^\circ \pm 0.95^\circ$

Table 3.9. *Results of MLP learning of the rotation angle of leaf shapes,* $\alpha \in [0^\circ, 45^\circ]$ *in steps of* 0.5°*, using SEC histograms.*

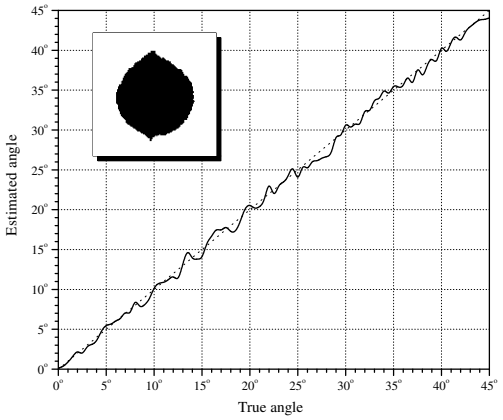
I also trained an MLP to estimate the angle α for each leaf image separately. I used the same 12 input nodes corresponding to the SEC histogram,



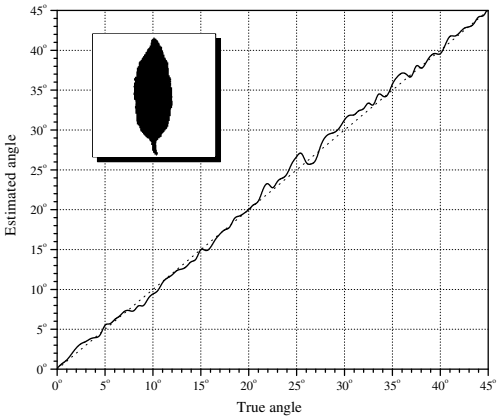
(a) Fig leaf, A.



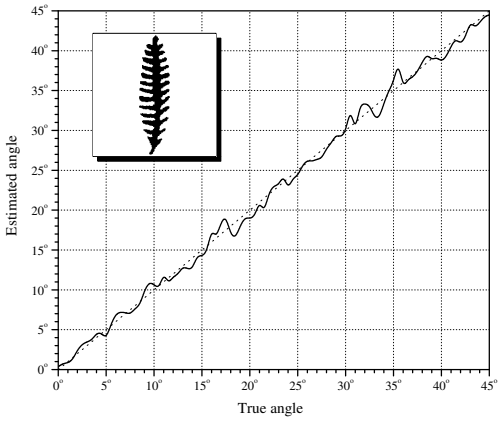
(b) Poinsettia leaf, B.



(c) Birch leaf, C.



(d) Lemon leaf, D.

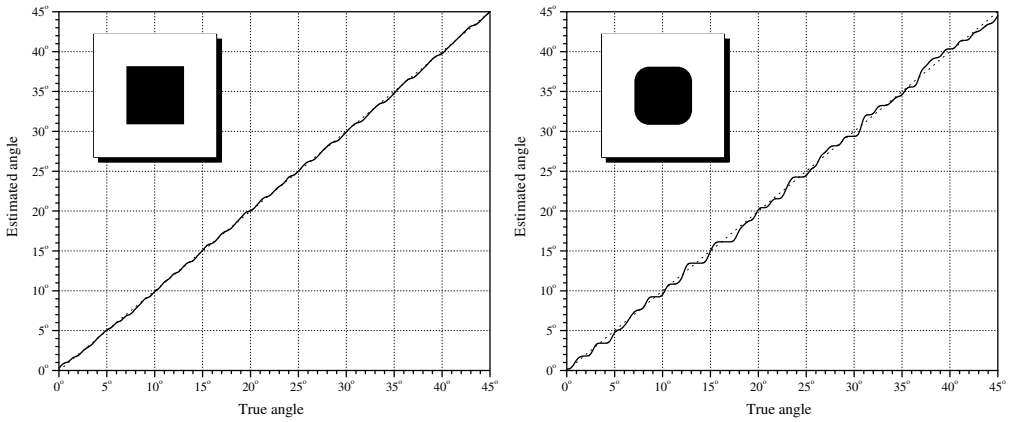


(e) Fern leaf, E.

Figure 3.7. *Results of MLP learning of the rotation angle of leaf shapes,* $\alpha \in [0°, 45°]$ *in steps of* $0.5°$*, using SEC histograms.*

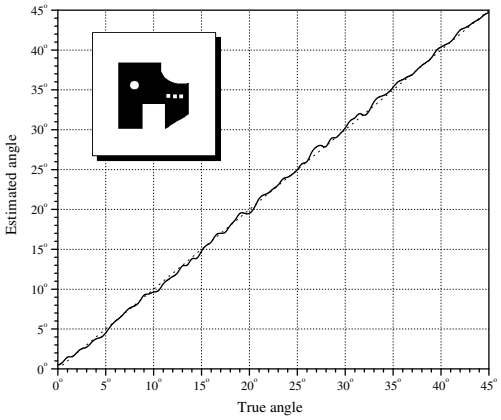
3 hidden nodes, and a single linear output node to estimate α. The results are summarized in Figure 3.7 and Table 3.9.

The leaf images in Figure 3.4 are not specially "easy" shapes when compared to typical machined parts in industrial machine vision environments. Figure 3.8 and Table 3.10 show the results of the same test with two simple geometrical forms and a synthetic machined part ("Piece").



(a) Square.

(b) Rounded square.



(c) Piece.

Figure 3.8. *Results of MLP learning of the rotation angle of some shapes,* $\alpha \in [0°, 45°]$ *in steps of* $0.5°$*, using SEC histograms.*

Note that all images (shadowed regions in the graphs) are 256×256, i.e. resolution is very low. Thus, for instance, the square shape in Figure 3.8(a) is 120 pixels long. This means that $0.5°$ accuracy in the determina-

	Error
Square	$0.03° \pm 0.22°$
Rounded square	$0.08° \pm 0.55°$
Piece	$0.06° \pm 0.40°$

Table 3.10. *Results of MLP learning of the rotation angle of some shapes,* $\alpha \in [0°, 45°]$ *in steps of* $0.5°$*, using SEC histograms.*

tion of the rotation angle around the centre of the square corresponds to a maximum displacement of $120/\sqrt{2} \cdot \sin(0.5°) = 0.74$ pixel.

The accuracy of SEC histograms can be compared to that of the standard method to obtain the rotation angle of a binary shape, the angle of the principal eigenvector of the central moment matrix,

$$\alpha = \frac{1}{2} \arctan\left(2 \frac{\mu_{11}}{\mu_{20} - \mu_{02}}\right). \tag{3.1}$$

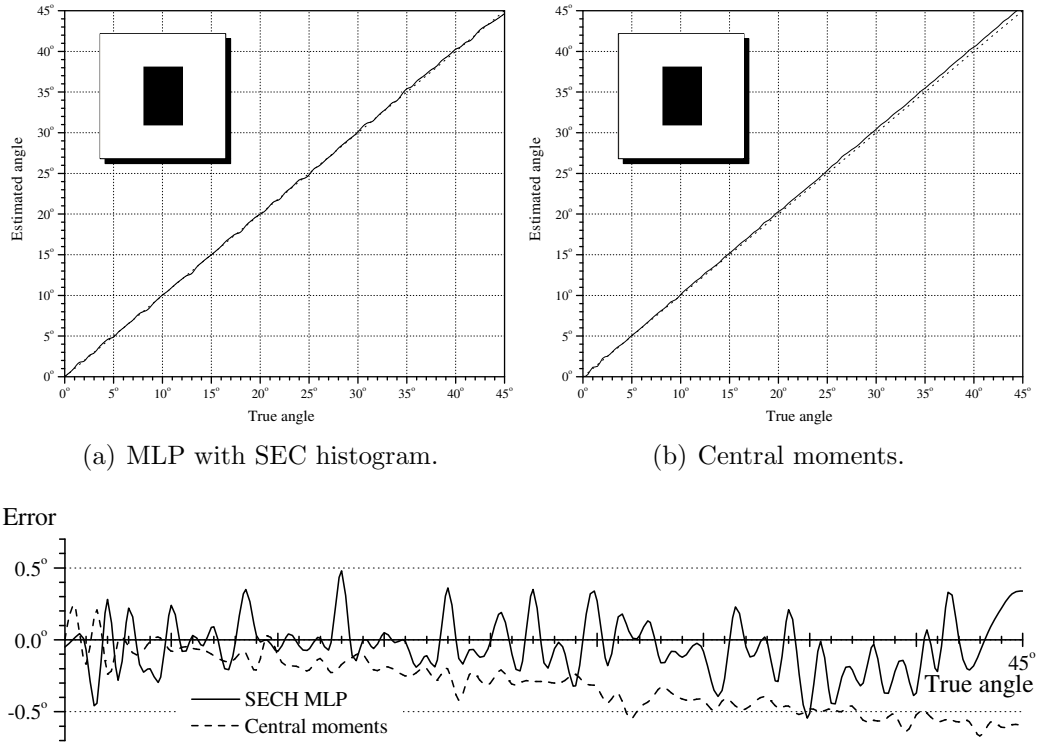
However, moments cannot be applied to shapes without a clear principal axis, such as the square or the rounded square in Figure 3.8. I compared the SEC histogram results with the moment method on a 120×80 rectangle. Figure 3.9 and Table 3.11 summarize the results.

	Error
SECH MLP	$0.04° \pm 0.20°$
Central moments	$0.31° \pm 0.20°$

Table 3.11. *SEC histogram MLP vs. central moments for the estimation of the rotation angle of a* 120×80 *rectangle,* $\alpha \in [0°, 45°]$ *in steps of* $0.5°$*.*

Additionally, Table 3.12 compares the results of the moment method with those of the SEC histogram MLP for the leaf shapes in Figure 3.4. For a fair comparison, moment results have been corrected according to the moment based estimation of rotation angle for the original images.

Moments show higher average error, but lower standard deviation. The variance in the SEC histogram MLP is due to the MLP itself, and its ability to model the underlying function. The higher average error in the moment



(a) MLP with SEC histogram.

(b) Central moments.

(c) Error curves.

Figure 3.9. *SEC histogram MLP vs. central moments for the estimation of the rotation angle of a* 120×80 *rectangle,* $\alpha \in [0°, 45°]$ *in steps of* $0.5°$*.*

Error	Moments	SECH MLP
Fig leaf	$0.30° \pm 0.19°$	$0.13° \pm 0.70°$
Poinsettia leaf	$0.28° \pm 0.17°$	$0.14° \pm 0.72°$
Birch leaf	$0.14° \pm 0.72°$	$0.15° \pm 0.84°$
Lemon leaf	$0.64° \pm 0.19°$	$0.21° \pm 0.72°$
Fern leaf	$0.31° \pm 0.19°$	$0.07° \pm 0.95°$

Table 3.12. *Comparison of central moments and SEC histogram MLP for the estimation of the rotation angle of the leaf shapes in Figure 3.4,* $\alpha \in [0°, 45°]$ *in steps of* $0.5°$*.*

method is due to the low accuracy induced by the low resolution of the images. The SEC histogram MLP is more immune to this, because as long as there are differences in the images between different angles, the problem is reduced to being able to partition the input space in accordance to the training

labels. The moment method, however, is based on geometry, and digitization inaccuracies due to the discrete nature of the image are unavoidably transferred to the final result.

Last, I measured similarities between all leaf images in Figure 3.4. This goes beyond the intended scope for the application of SEC histograms, but the results serve to show the potential of SEC histograms when the constraints mentioned above are relaxed. Tables 3.13 show the pairwise correlation (product moment) between all leaf images in Figure 3.4. Tables 3.14 show all the pairwise Kullback-Leibler distances [137]. Note that the Kullback-Leibler distance $\mathrm{KL}(p,k) = \sum_i p_i \log(p_i/k_i)$ is not symmetric (it is not a proper metric).

ρ	**A**	**B**	**C**	**D**	**E**
a	**98.6**%	97.7%	97.4%	88.7%	59.7%
b	97.8%	**98.7**%	97.6%	86.2%	61.4%
c	98.0%	97.9%	**98.5**%	81.6%	69.0%
d	80.0%	79.4%	74.5%	**99.7**%	12.8%
e	61.2%	61.0%	67.3%	15.6%	**100.0**%

ρ	**A**	**B**	**C**	**D**	**E**
A	**99.6**%	99.1%	99.6%	82.5%	66.6%
B	98.7%	**99.3**%	99.2%	81.4%	67.6%
C	98.1%	98.2%	**99.0**%	79.9%	66.5%
D	75.3%	75.0%	70.9%	**95.1**%	21.8%
E	59.1%	59.4%	65.3%	14.2%	**99.6**%

ρ	***A***	***B***	***C***	***D***	***E***
a	99.0%	98.3%	**99.3**%	78.7%	65.3%
b	98.0%	**99.4**%	99.3%	77.8%	66.3%
c	97.3%	97.8%	**99.2**%	76.0%	65.2%
d	83.9%	82.3%	78.5%	**96.0**%	19.9%
e	56.5%	59.0%	66.3%	10.0%	**99.5**%

Table 3.13. *Matching the downscaled shapes (small letters), the warped shapes (slanted letters), and the original leaf shapes (capital letters) in Figure 3.4 with product moment correlation between SEC histograms. Bold: Row winners.*

With correlation the only error arises when matching the downscaled

shapes to the warped shapes, where the downscaled Fig leaf (a) is wrongly assigned the warped Birch leaf (C), or, the other way around, where the warped Birch leaf (C) is wrongly assigned the downscaled Fig leaf (a). Otherwise, the results are quite convincing, as they do not only find the correct template, but also show a behaviour coherent with the visual inspection of the shapes, such that the three first shapes (Fig, Poinsettia and Birch leaves) are quite similar, and the Fern leaf is the most dissimilar to any other.

KL	**A**	**B**	**C**	**D**	**E**
a	0.035	0.041	**0.028**	0.142	0.204
b	0.056	**0.035**	0.036	0.162	0.184
c	0.060	0.055	**0.037**	0.195	0.140
d	0.215	0.211	0.234	**0.008**	0.617
e	0.264	0.255	0.211	0.588	**0.000**

KL	**A**	**B**	**C**	**D**	**E**
A	0.014	0.019	**0.007**	0.198	0.200
B	0.036	0.022	**0.018**	0.199	0.170
C	0.038	0.035	**0.018**	0.212	0.188
D	0.258	0.239	0.237	**0.100**	0.328
E	0.283	0.268	0.227	0.579	**0.002**

KL	***A***	***B***	***C***	***D***	***E***
a	**0.008**	0.020	0.016	0.271	0.209
b	0.013	**0.004**	0.009	0.261	0.179
c	0.018	0.019	**0.011**	0.280	0.197
d	0.119	0.113	0.119	**0.078**	0.341
e	0.188	0.171	0.135	0.637	**0.003**

KL	**a**	**b**	**c**	**d**	**e**
A	**0.008**	0.013	0.017	0.140	0.227
B	0.022	**0.004**	0.020	0.141	0.200
C	0.016	**0.009**	0.011	0.151	0.158
D	0.199	0.197	0.214	**0.078**	0.642
E	0.179	0.157	0.178	0.395	**0.003**

Table 3.14. *Matching the downscaled (small letters), warped (slanted letters), and original (capital letters) leaf shapes in Figure 3.4 with Kullback-Leibler pairwise distances between SEC histograms. Bold: Row winners.*

Kullback-Leibler, on the other hand, shows also a behaviour fairly coherent but fails to find the right winner in more cases than the simple product moment correlation. It is true that the failures are not by big margins, and that the second closest match is always the right guess in all errors, but clearly the product moment correlation outperforms the Kullback-Leibler distance in this case. Somehow the behaviour of the Kullback-Leibler distance and the correlation are complementary, Kullback-Leibler shows better results where the correlation fails, and vice versa. This is probably due to the normalization implicit in the Kullback-Leibler distance (histograms are treated as probability distributions with unit area), which may help when matching shapes at different scales, but not when matching shapes at the same scale. However, no matter the specific method used for the matching, the potential of SEC histograms is clearly demonstrated.

To recapitulate, the aim of this section was not the design of a machine vision system, but to show that an MLP is able to learn to distinguish among several objects, and able to estimate the rotation of different objects with SEC histograms as only input. Therefore, to demonstrate that SEC histograms as feature vectors contain enough information for this kind of discrimination, and to show their potential for the fulfillment of common tasks in industrial machine vision applications in constrained environments (production lines of rigid objects), such as shape control, rotation control, inspection of presence/absence of features such as holes, drills, or notches, or automated classification. With the advantage that SEC histograms can be obtained with very low computational effort, in just a single raster scan of the image or region of interest within the image, requiring just a few integer increments. The method used to map the feature space of SEC histograms into an output space of knowledge, decision, or action, can be any of the many in the literature able to cope with a feature space with 12 to 15 dimensions, and will determine the total overhead of the system. If the mapping method is not computationally very heavy, such as, for instance, a simple product moment correlation, or an MLP trained off-line with a few hidden nodes, very high frame rates can be achieved in systems where the binarization process is very simple, such as backlit inspection.

Another relevant characteristic of SEC histograms is that they can be computed on the fly, thus reducing storage requirements to a minimum — two image lines at most, or even the equivalent to a single line plus one pixel. This, together with their computational simplicity, renders them specially appealing for dedicated hardware implementations for very fast machine vision.

3.5 Texture analysis

SEE can be used as a textural feature vector by applying a convenient binarization method to the original image and obtaining the SEC histogram. Used in that way, SEC histograms resemble the Local Binary Patterns (LBP) of Ojala and Pietikäinen [189], with a few notable differences. LBP are obtained by constructing a binary word with the pixels adjacent to the pixel of interest, which are assigned the binary value 1 if they have a graylevel higher than the central pixel and value 0 if lower. Thus, LBP implicitly include a local binarization method, where 3×3 windows are binarized by thresholding with the graylevel of the central pixel. After the binarization, the local configuration is encoded in a 8-bit binary array by unwinding the pixels surrounding the central pixel. This produces 256 different codes, but the authors suggest accumulating them according to a criterion of uniformity, where binary arrays with none or one bit transition are considered uniform, and assigned a uniform code equal to the number of consecutive ones in the original LBP, therefore in the range $[0, 8]$, and all other are considered non uniform LBP and are assigned the code 9. This produces the uniform $\mathrm{LBP}_8^{\mathrm{u}}$ distribution, which has been shown to be a useful tool for texture classification and segmentation [151, 190, 198]. The LBP concept has been later extended to include a set of rotational invariant texture features by increasing the neighbourhood size using interpolation, at the cost of increased computational overhead [191].

SEC histograms, on the other hand, do not include implicitly any given binarization method, and therefore anyone can be used depending on the characteristics of the original images or the application. For purposes of comparison, in the following I will use the same local binarization method implicit in the LBP. After binarization, the SEE codes represent the spatial configuration of 2×2 windows, instead the 3×3 windows of LBP, and no "uniformization" procedure is carried on, such that all the different possible configurations are preserved in the SEC histogram. This is possible without a notable increase in the feature space dimension —16 possible SEC vs. 10 possible $\mathrm{LBP}_8^{\mathrm{u}}$— due to the smaller window size implicit in SEE.

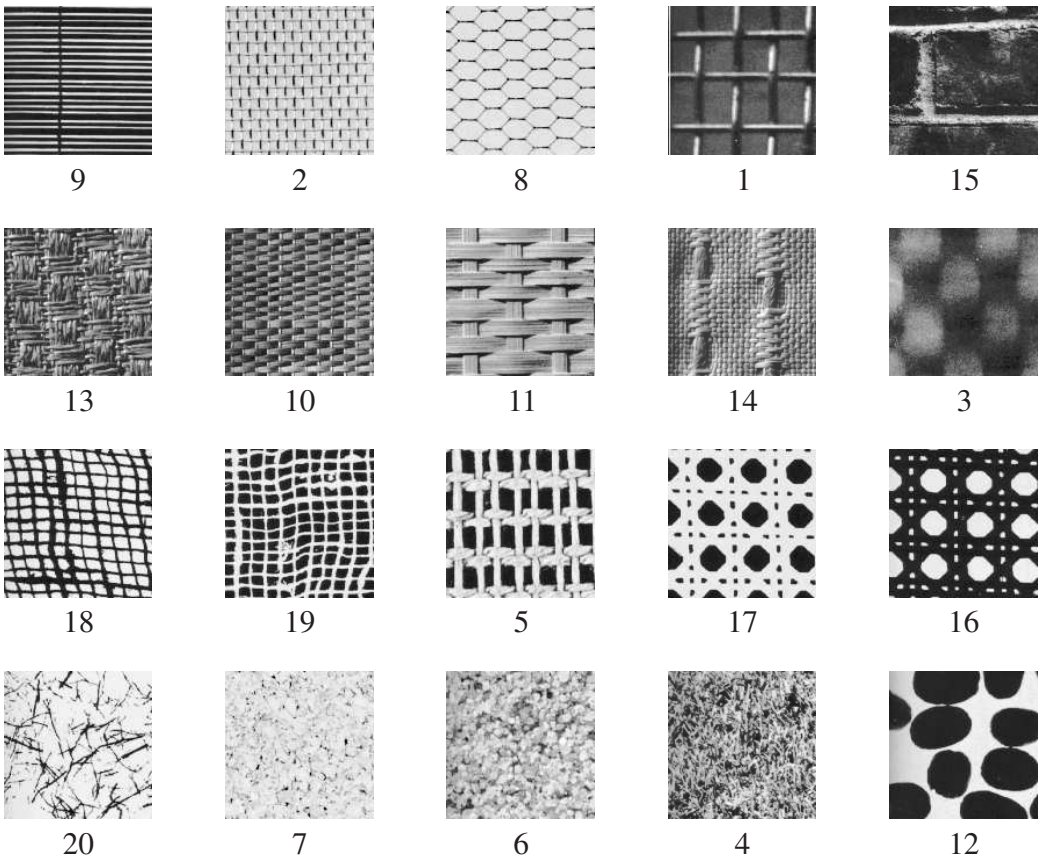


Figure 3.10. *A representative of each of the 20 Brodatz textures used in the tests.*

I performed some tests with samples from the Brodatz album [37], to compare the performance of SECH and $\mathrm{LBP}_8^{\mathrm{u}}$ for texture analysis, namely Nearest Neighbour (k-NN) classification [2, 66, 76] of texture samples. I also mapped the SECH of Brodatz samples into a SOFM 2D [130–134]. Figure 3.10 shows 20 128×128 samples from 20 different Brodatz textures. The Brodatz album was composed in 1966 for artistic purposes, but since then it has been used as a source of benchmark texture samples by many authors. The photographs in the Brodatz album depict very different natural materials under diverse illumination conditions. Only some of the photographs in the Brodatz album can be used as sources of textures, and even these present a high heterogeneity within the frame due to both the targets and nonuniform illumination. I chose 20 different Brodatz images, digitized at 640×640,

and extracted from each of them 25 non overlapping 128×128 samples, to build a test set with 500 samples belonging to 20 different texture classes. The chosen textures cover a wide range of materials while keeping a certain degree of similarity among them. A representative of each class can be seen in Figure 3.10, where they have been somehow arranged with a criterion of similarity, and Figure 3.11 shows some representatives of some of the classes to show the degree of variability within classes, due to the differences in illumination and the heterogeneity of the materials. The samples were fed into the classifiers as such, with no preprocessing to equalize the graylevel within classes, nor rotations or translations to match spatial differences.

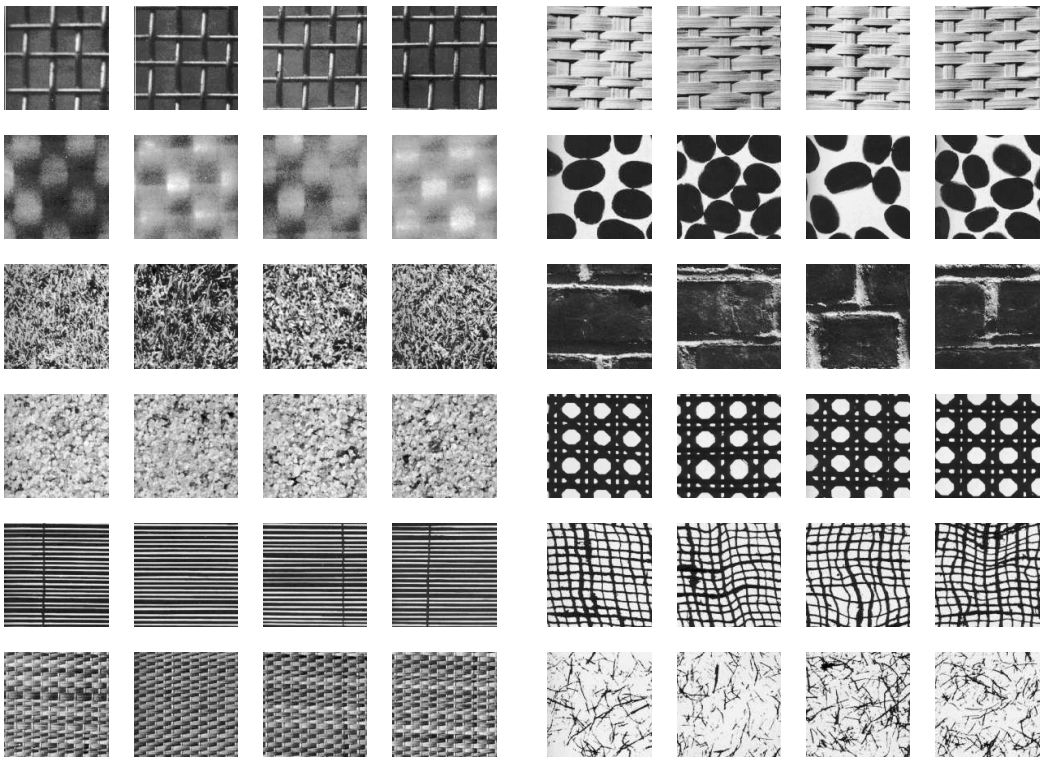


Figure 3.11. *Four representatives of some of the textures used in the tests, showing the within class variations due to non uniform illumination and the heterogeneity of the materials.*

Tables 3.15 to 3.18 show the results of classifying the entire sample set (500 samples from 20 different classes) with 1-NN and voting 3-NN, using SECH and LBP, with a *leave-one-out* scheme [139]. In a leave-one-out clas-

sification experiment, each sample is used in turn as test set and all the rest as training set. This scheme is particularly convenient in this case, because the number of samples available is low, such that dividing them among a training set and a test set (*holdout* scheme) would yield too small sets, and we are dealing with a lazy classifier, such that there is no cost associated to entirely retraining the classifier for each sample in the set. Note also that leave-one-out yields a pessimistic error bound —in fact, an upper bound for the Bayes error [85]— when compared to resubstitution, as the probability of misclassification is increased because the sample set of the test class has one sample less in the training set than the rest of classes.

%	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1	92	0	4	0	0	4	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	0	96	0	0	0	0	0	4	0	0	0	0	0	0	0	0	0	0	0	0
3	4	0	84	0	0	0	0	0	0	8	0	4	0	0	0	0	0	0	0	0
4	0	0	0	92	4	0	0	0	0	0	0	0	4	0	0	0	0	0	0	0
5	0	0	0	0	80	8	0	0	0	0	0	0	0	8	0	0	4	0	0	0
6	0	0	0	0	0	96	0	0	0	0	0	0	0	0	0	0	0	0	0	4
7	0	0	0	0	0	0	100	0	0	0	0	0	0	0	0	0	0	0	0	0
8	0	4	0	0	0	0	8	88	0	0	0	0	0	0	0	0	0	0	0	0
9	8	0	0	0	0	4	0	0	84	0	0	0	0	0	4	0	0	0	0	0
10	0	0	0	0	0	0	0	0	0	100	0	0	0	0	0	0	0	0	0	0
11	0	0	0	0	0	0	0	0	0	0	96	0	0	4	0	0	0	0	0	0
12	0	0	0	4	0	0	0	0	0	0	0	76	0	0	8	4	0	0	8	0
13	0	0	0	0	0	0	0	0	0	0	0	0	96	4	0	0	0	0	0	0
14	0	0	0	0	8	0	0	0	0	0	0	0	8	84	0	0	0	0	0	0
15	20	0	0	0	0	0	0	0	0	4	0	4	0	0	72	0	0	0	0	0
16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	80	12	4	4	0
17	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	4	96	0	0	0
18	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	4	0	68	28	0
19	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	4	0	20	76	0
20	0	0	0	4	4	4	0	0	0	0	0	0	0	0	0	0	0	0	0	88

Table 3.15. *Confusion matrix in the classification of the Brodatz set ($N = 500$, $C = 20$) using 1-NN with SECH. Overall accuracy: 87.2%, κ: 86.5%.*

The classification results with 1-NN are very different for SECH and LBP. SECH yields an overall accuracy above 87% whereas LBP does not reach the 50% mark. κ values are not better for LBP, 46.7% versus 86.5%. The bigger difficulties (accuracy below 80%) for SECH are with textures 18 and 19 (the

%	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1	32	4	0	0	16	0	20	0	0	0	0	4	0	0	0	0	0	12	0	12
2	0	60	0	0	20	0	0	0	4	0	0	0	0	0	8	4	4	0	0	0
3	0	0	88	0	0	0	0	0	0	4	0	4	0	0	0	0	0	4	0	0
4	0	0	0	68	0	20	0	0	0	0	0	0	4	8	0	0	0	0	0	0
5	8	20	0	0	28	0	0	8	0	0	0	8	0	0	20	4	4	0	0	0
6	0	0	0	20	0	44	0	0	0	0	0	0	16	20	0	0	0	0	0	0
7	20	0	0	0	0	0	56	0	0	0	0	4	0	0	0	0	0	8	0	12
8	4	0	0	0	8	0	0	40	0	0	0	8	0	0	16	4	12	0	0	8
9	0	8	0	0	0	0	0	0	88	0	0	4	0	0	0	0	0	0	0	0
10	0	0	0	0	0	0	0	0	0	96	4	0	0	0	0	0	0	0	0	0
11	0	0	4	0	0	0	0	0	0	4	80	0	0	8	0	0	0	0	4	0
12	8	4	0	0	8	0	4	8	4	0	0	24	0	0	12	8	4	4	0	12
13	0	0	0	0	0	28	0	0	0	0	0	0	48	12	0	0	0	4	8	0
14	0	0	0	12	0	24	0	0	0	0	16	0	8	40	0	0	0	0	0	0
15	0	8	0	0	8	0	0	16	0	0	0	8	0	0	8	12	20	4	4	12
16	0	4	0	0	4	0	0	4	0	0	0	8	0	0	16	48	16	0	0	0
17	0	4	0	0	4	0	0	8	0	0	0	8	0	0	32	20	20	0	0	4
18	16	0	4	0	0	0	0	0	0	0	4	0	4	0	4	0	0	40	28	0
19	4	4	4	0	0	0	0	0	0	0	4	0	16	0	0	0	0	40	28	0
20	4	0	0	0	4	0	8	4	0	0	0	8	0	0	16	0	4	0	0	52

Table 3.16. *Confusion matrix in the classification of the Brodatz set ($N = 500$, $C = 20$) using 1-NN with LBP. Overall accuracy: 49.4%, κ: 46.7%.*

direct and inverse versions of the loose tissue in the third row of Figure 3.10, see several samples in Figure 3.11), 15 and 1 (the brick wall and the thick wire fence), and 12 (the backlit coffee beans, also with a high internal variability, see Figure 3.11). On the other hand, all samples of textures 7 and 10 are correctly classified (see Figure 3.11 for a glimpse of the within class variability of texture 10 due to nonuniform illumination), and textures 2, 6, 11 (another class with large within class variability due to the illumination), 13, and 17 are classified with accuracy above 95%.

To comment the results with LBP we have to use different thresholds, because 80% of the texture classes did not reach the 80% mark, and only one surpassed 95% accuracy. LBP presents the bigger difficulties (accuracy below 30%) with textures 5, 12, 15, 17, and 19. These include one of the texture classes, 17, where SECH scored above 95%. The best results (above

75%) of LBP are with textures 3, 9, 10, and 11. These include two of the textures, 10 and 11, with good results also for SECH, even if SECH yields better figures. Textures 3 (with a huge within class variability, see Figure 3.11) and 9 are the only two where LBP outperformed SECH, both by 4% (88% vs. 84%). The performance of SECH was clearly superior to that of LBP. SECH's worst and second worst results were 68% and 72%, whereas LBP's were 8% and 20%. On the other hand, SECH scored full accuracy in two textures, whereas LBP's best and second best were 96% and 88%.

%	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1	88	0	0	0	0	8	0	0	0	0	0	0	0	0	4	0	0	0	0	0
2	0	88	0	0	0	4	0	4	0	0	0	0	0	0	0	0	0	0	0	4
3	4	0	64	0	0	0	0	0	0	16	4	4	0	0	8	0	0	0	0	0
4	0	0	0	92	4	0	0	0	0	0	0	0	4	0	0	0	0	0	0	0
5	0	0	0	0	64	16	0	0	0	0	0	0	0	8	0	0	8	0	0	4
6	0	0	0	0	0	100	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	100	0	0	0	0	0	0	0	0	0	0	0	0	0
8	0	4	0	0	0	0	8	88	0	0	0	0	0	0	0	0	0	0	0	0
9	4	4	0	0	0	4	0	0	84	0	0	0	0	0	4	0	0	0	0	0
10	0	0	0	0	0	0	0	0	0	100	0	0	0	0	0	0	0	0	0	0
11	0	0	0	0	0	0	0	0	0	0	88	0	0	12	0	0	0	0	0	0
12	0	0	0	4	0	0	0	0	0	0	0	72	0	0	0	4	0	4	16	0
13	0	0	0	0	0	0	0	0	0	0	0	0	96	4	0	0	0	0	0	0
14	0	0	0	0	4	0	0	0	0	0	0	0	12	84	0	0	0	0	0	0
15	12	0	0	0	0	0	0	0	0	4	0	4	0	4	76	0	0	0	0	0
16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	80	12	4	4	0
17	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	4	96	0	0	0
18	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	60	40	0
19	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	16	84	0
20	0	0	0	4	8	4	0	0	0	0	0	0	0	0	0	0	0	0	0	84

Table 3.17. *Confusion matrix in the classification of the Brodatz set ($N = 500$, $C = 20$) using 3-NN with SECH. Overall accuracy: 84.4%, κ: 83.6%.*

Using a voting 3-NN classifier, where each test sample is assigned the class of the majority of its three nearest neighbours, or that of the nearest neighbour in case of tie, things do not change much. SECH loses some advantage, 2.8% in overall accuracy down to 84.4%, and 2.9% in κ down to 83.6%, whereas LBP improves a little bit its results, 1.8% in overall accuracy up to 51.2% and 1.9% in κ up to 48.6%.

%	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1	32	4	0	0	8	0	24	0	0	0	0	0	0	0	0	0	0	8	12	12
2	0	48	0	0	36	0	0	0	4	0	0	4	0	0	0	4	4	0	0	0
3	0	0	80	0	0	0	0	0	0	12	0	4	0	0	0	0	0	0	4	0
4	0	0	0	48	0	28	0	0	0	0	0	0	8	16	0	0	0	0	0	0
5	8	20	0	0	32	0	0	8	0	0	0	4	0	0	20	4	4	0	0	0
6	0	0	0	20	0	40	0	0	0	0	0	0	20	20	0	0	0	0	0	0
7	8	0	0	0	0	0	76	0	0	0	0	4	0	0	0	0	0	8	0	4
8	4	0	0	0	8	0	0	44	0	0	0	4	0	0	4	0	24	0	0	12
9	0	4	0	0	0	0	0	0	92	0	0	4	0	0	0	0	0	0	0	0
10	0	0	0	0	0	0	0	0	0	84	16	0	0	0	0	0	0	0	0	0
11	0	0	0	0	0	0	0	0	0	8	76	0	0	8	0	0	0	4	4	0
12	8	4	0	0	8	0	4	8	4	0	0	8	0	0	12	20	4	4	0	16
13	0	0	0	0	0	24	0	0	0	0	0	0	56	12	0	0	0	0	8	0
14	0	0	0	8	0	12	0	0	0	0	16	0	12	52	0	0	0	0	0	0
15	0	4	0	0	12	0	0	16	0	0	0	8	0	0	8	4	28	4	4	12
16	0	4	0	0	0	0	0	4	0	0	0	4	0	0	8	52	28	0	0	0
17	0	4	0	0	4	0	0	4	0	0	0	0	0	0	12	20	48	0	0	8
18	8	0	4	0	0	0	4	0	0	0	4	0	4	0	4	0	0	32	40	0
19	4	4	4	0	0	0	0	0	0	0	4	0	8	0	0	0	0	16	60	0
20	4	0	0	0	0	0	12	4	0	0	0	4	0	0	16	0	4	0	0	56

Table 3.18. *Confusion matrix in the classification of the Brodatz set ($N = 500$, $C = 20$) using 3-NN with LBP. Overall accuracy: 51.2%, κ: 48.6%.*

Figure 3.12 shows a mapping of the SECH feature vectors into a $\mathbb{R}^2$ map using a discrete SOFM. The ellipses represent the principal components of the feature vectors of each class mapped in $\mathbb{R}^2$. The ellipses in the 2D map show the variance of each texture class after transformation, oriented along the principal axis. Due to the topology preservation characteristic of the SOM, this is a convenient way of visualizing the relations among texture classes according to SECH, even when we will probably be loosing some information, because we cannot assure that the SECH vectors are contained into a 2D manifold in the feature space. Anyway, we see a good correlation between the map in Figure 3.12 and the k-NN results in Tables 3.15 and 3.17.

Again, I have not intended the design of an optimum classifier for texture discrimination in digital images, I have just shown that SECH vectors can be

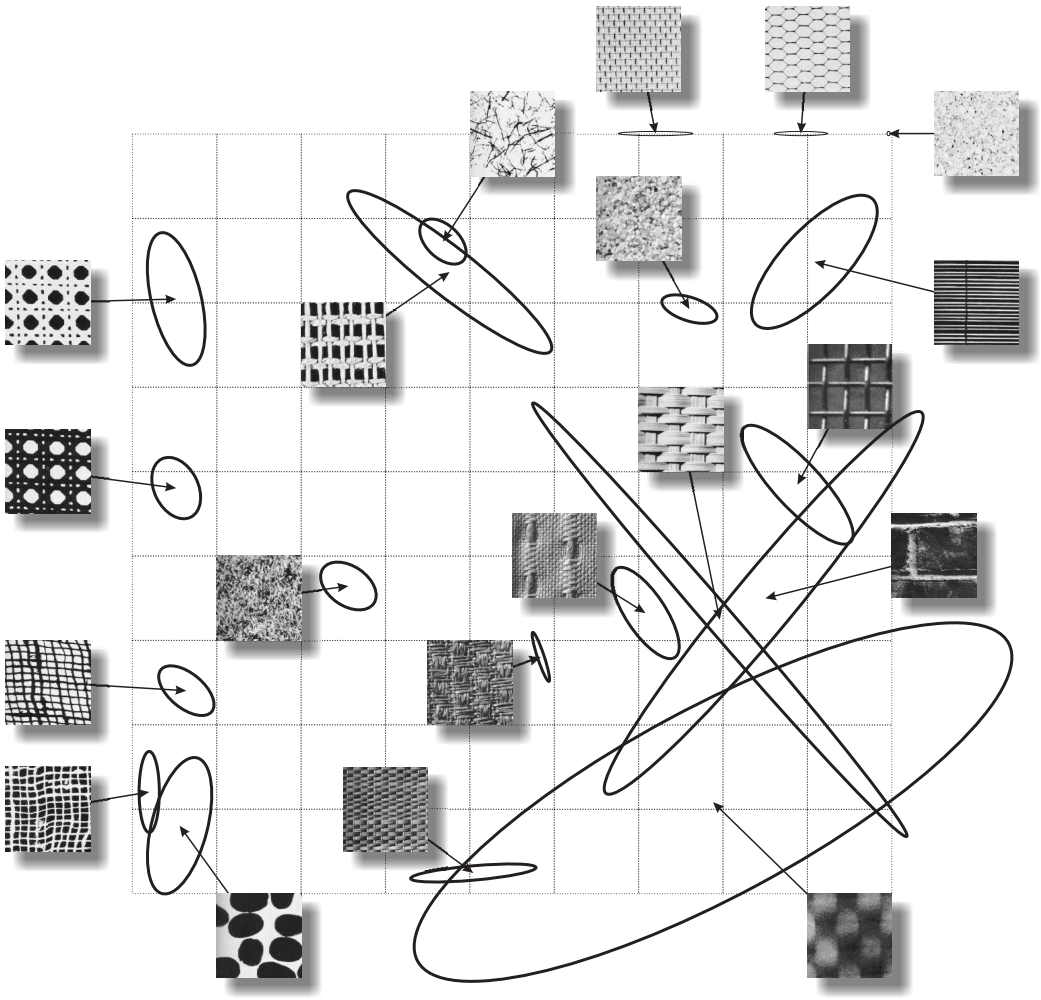


Figure 3.12. *Nonlinear mapping of the SECH feature vectors of the 20 Brodatz textures into a* $\mathbb{R}^2$ *map using a SOFM.*

useful for texture analysis and classification, through a classification example and the visualization of a reduced dimension mapping of the SECH feature space for a benchmark dataset, which demonstrate a coherent behaviour of the SECH vectors as simple and fast texture descriptors, with a better behaviour in the case at hand than the well known uniform local binary patterns.

3.6 SEE in the numerical method

We have just seen a number of applications to image processing and computer vision of the SEE local configuration scheme. To finish, we will see how SEE is of use in the numerical model for the analysis of the effective response of nonhomogenous media framing this report.

Recall the equivalent image representation (EIR) of the inclusion/matrix model of the nonhomogeneous media under study (see 1.3). We have to solve the potential for all nodes on boundaries inclusion/matrix (all object boundary nodes in the IER), and then perform numerical integration of the potential to obtain the effective field in each inclusion (see 1.2.2). SEE allows clean and efficient implementations of these computations.

3.6.1 Interpolation and random walks

No matter what method is used to solve the potential, the set of s_ν coefficients in (1.5), six in 3D systems (four in 2D), has to be determined for each node in the mesh. The determination of these coefficients requires the inspection of all eight (four) neighbour elementary cells of each node to determine their permittivity, as in (1.6). This could be done once per iteration, with the corresponding overhead, or only once before beginning the iterations if a six (four) dimensional array is used to store the coefficients of each of the L^3 (L^2) nodes in the mesh, which would take a considerable amount of memory.

SEE provides an efficient solution by encoding the local configuration of the elementary cells surrounding each node with very low cost (see 3.2), such that later, during the iterative computations, the code associated to each node will determine the coefficients from a predefined finite set of possibilities stored in a look-up table (LUT).

Thus, for instance, in the 2D case the coefficients $s_{\mathrm{x}^+}(c)$, $s_{\mathrm{x}^-}(c)$, $s_{\mathrm{y}^+}(c)$, $s_{\mathrm{y}^-}(c)$ for each possible SEE code $c \in [0, 15]$ are computed as in (1.6) and stored in four arrays, indexed by the corresponding code, thus configuring a

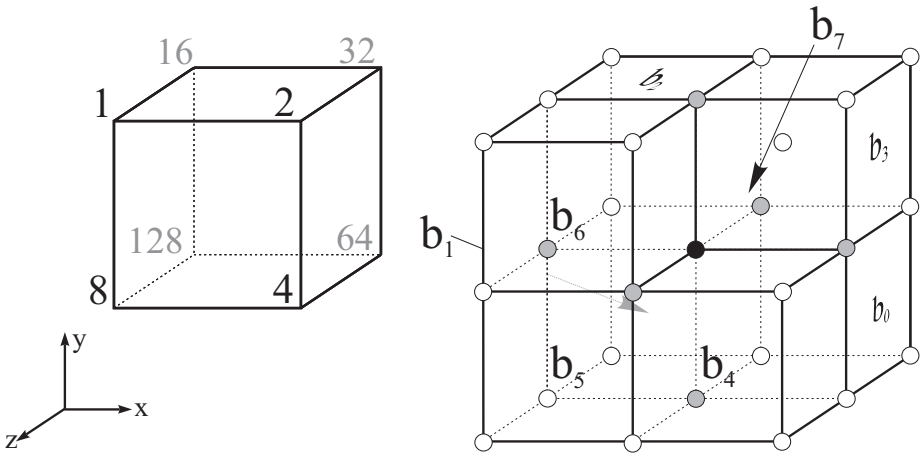


Figure 3.13. *3D SEE: (left) A voxel with weights assigned to each vertex; (right) A node (black dot) surrounded by 8 voxels and 6 neighbour nodes (grayed dots). Each foreground voxel is represented by a 1 in the indicated bit (b_i) of the SEE codeword.*

LUT of coefficients. Then, expressions such as $s_{\mathrm{x}^+}(c_{x,y})$ will automatically provide the corresponding coefficients for node (x, y).

Therefore a 16×4 LUT stores the s_ν coefficients of (1.5) using SEE codes, saving both memory and computation overhead. But the LUT has to be built, i.e. the s_ν coefficients still have to be computed for every possible local configuration, in the fashion of (1.6). As there are 16 such different configurations in 2D they can be precomputed one by one and *hard coded* into the program. But this is entirely out of question in the 3D case, with 256 different configurations.

Conveniently, SEE codes do not only provide an efficient mean of identifying node configurations, but also keep an internal binary encoding of the configuration of the nodes, which greatly facilitates the construction of the coefficients in (1.5). Once a given assignment of weights has been decided for the corners of the EIR pixels or voxels, we have seen that each SEE codeword can be viewed as a binary codification of the presence/absence of foreground pixels or voxels in the neighbourhood of the node, see Figure 3.13. The local constraints of conservation of the normal component of $\vec{D}$ and of the tangential component of $\vec{E}$ are reduced to formulas depending only on

the permittivity of the cells surrounding each node, as in (1.6), which corresponds to code 3 in Figure 3.2. Every node has as many coefficients as edges linking it to neighbouring nodes, six in the 3D case.

Let $n_{\mathrm{n}}(c)$ be the number of foreground voxels surrounding a type c node, where $c \in \{0, \cdots, 255\}$ is a SEE code, and let $n_v(c)$ be the number of foreground voxels surrounding the edge linking the node to one of its neighbours $v \in \{\mathrm{x}^+, \mathrm{x}^-, \mathrm{y}^+, \mathrm{y}^-, \mathrm{z}^+, \mathrm{z}^-\}$. Then the corresponding coefficient $s_v(c)$ can be computed as

$$s_v(c) = \frac{1}{3} \frac{n_v(c)\epsilon_{\mathrm{i}} + [4 - n_v(c)]\epsilon_{\mathrm{m}}}{n_{\mathrm{n}}(c)\epsilon_{\mathrm{i}} + [8 - n_{\mathrm{n}}(c)]\epsilon_{\mathrm{m}}} \tag{3.2}$$

where $n_{\mathrm{n}}(c)$ is the total number of 1's in the binary representation of c, and $n_v(c)$ is the number of 1's in four specific bits of c. Which four bits, depends on what coefficient we are interested in. For instance, in the SEE assignment of Figure 3.13, bits b_4, b_5, b_6, and b_7 of the SEE code c would provide $n_{\mathrm{z}^+}(c)$ for $s_{\mathrm{z}^+}(c)$, while bits b_0, b_3, b_4, and b_7 would provide $n_{\mathrm{x}^+}(c)$ for $s_{\mathrm{x}^+}(c)$.

Counting bits in an octet is fast and easy[6]. The LUT of coefficients in (1.5) can thus be constructed for the 256 SEE codes just by counting bits in the SEE codewords and applying (3.2) to get the six coefficients for each local configuration.

3.6.2 Numerical integration

The same reasoning can be used to solve the integrals in the right hand side of (1.8) in 1.2.2. To evaluate the contribution of a node to each of the components of the corresponding integral, we have to determine to how many free facets of occupied cells (EIR boundary voxels), normal to each direction, belongs the node. This depends on its local configuration, and again can be efficiently determined counting bits in the corresponding SEE codeword.

Thus, for instance, resorting again to the SEE assignment of Figure 3.13, if l_{e} is the length of an elementary cell, a node of type c with known potential

[6]Think, for instance, on Kernighan's brilliant C method to count the 1's in c: `int n; for(n=0;c;n++,c&=--c);` If only the 1's at certain positions are needed, masking c with the appropriate mask will do the job.

U would contribute $+\frac{1}{4}l_e^2U$ to the y component of the potential integral in (1.8) for each pair of bits $(b_0, b_3), (b_4, b_7), (b_5, b_6)$, and (b_1, b_2) in c satisfying $(b, \bar{b})$. And it would contribute $-\frac{1}{4}l_e^2U$ for each of those pairs satisfying $(\bar{b}, b)$. Two similar sets of pairs of bits can be straightforwardly derived for the other two components of the potential integral.

Appendix A

Analytical formulation for the computation of the effective permittivity of a composite

If your wish is to become really a man of science
and not merely a petty experimentalist,
I should advise you to apply to every branch of natural philosophy,
including mathematics.

We consider inclusions at mesoscopic scale, what allows to analyze the microstructure of the material preserving the intensive magnitudes of the constituents, as these can be defined only for sizes above a given minimum [154]. A minimum size of 10 Å has been reported for the validity of the Maxwell Garnett approach in the optical regime for embedded nanocrystallites [266]. We model the macroscopic system as a periodic array of unit cells enclosing a volume V_{C}, which represents the minimum volume that preserves the macroscopic homogeneity according to the EMA.

The effective response of the unit cell can be obtained by equating the macroscopic electric displacement to the ergodic mean of the local fields in the cell. The macroscopic field in the unit cell is

$$\vec{E}_0 = \left\langle \vec{E}(\vec{r}) \right\rangle_{V_{\mathrm{C}}} = \frac{1}{V_{\mathrm{C}}} \int_{V_{\mathrm{C}}} \vec{E}(\vec{r})\, d^3 r. \tag{A.1}$$

And the mean value of the field in each connected domain d, with volume V_d, is

$$\vec{E}_d = \frac{1}{V_d} \int_{V_d} \vec{E}(\vec{r})\, d^3r. \tag{A.2}$$

Averaging the local displacement field over the unit cell,

$$\begin{aligned} \overleftrightarrow{\epsilon}_{\text{eff}}\ \vec{E}_0 = \vec{D}_0 = \left\langle \vec{D}(\vec{r}) \right\rangle_{V_\text{C}} = \left\langle \epsilon(\vec{r})\vec{E}(\vec{r}) \right\rangle_{V_\text{C}} = \\ = \frac{1}{V_\text{C}} \int_{V_\text{C}} \epsilon(\vec{r})\vec{E}(\vec{r})\, d^3r = \sum_d p_d \epsilon_d \vec{E}_d \end{aligned} \tag{A.3}$$

where the summation is extended over all d domains, each with known permittivity ϵ_d and occupying a volume fraction $p_d = V_d/V_\text{C}$. Analogously, (A.1) and (A.2) lead to

$$\overleftrightarrow{\epsilon}_{\text{eff}}\ \vec{E}_0 = \overleftrightarrow{\epsilon}_{\text{eff}}\ \frac{1}{V_\text{C}} \int_{V_\text{C}} \vec{E}(\vec{r})\, d^3r = \overleftrightarrow{\epsilon}_{\text{eff}} \sum_d p_d \vec{E}_d. \tag{A.4}$$

Therefore

$$\overleftrightarrow{\epsilon}_{\text{eff}}\ \vec{E}_0 = \sum_d p_d\ \overleftrightarrow{\epsilon}_{\text{eff}}\ \vec{E}_d = \sum_d p_d \epsilon_d \vec{E}_d. \tag{A.5}$$

Each of the constituents is treated as isotropic, but the effective complex permittivity $\overleftrightarrow{\epsilon}_{\text{eff}}$ is, in general, a tensor. If the local $\vec{E}_d$ fields are aligned mutually and with $\vec{E}_0 = E_0\hat{e}$, i. e. $\vec{E}_d = E_d\hat{e}$, (A.5) leads to

$$\epsilon_{\text{eff}} = \frac{\sum_d p_d \epsilon_d E_d}{\sum_d p_d E_d} \tag{A.6}$$

from which some of the mixing rules can be derived. For instance, if the local fields are approximately equal to the effective macroscopic field, i.e. highly homogeneous unit cell or stratified inclusions parallel to the macroscopic field, the upper Wiener bound follows, $\epsilon_{\text{eff}} = \sum p_d \epsilon_d$. In the case of stratified inclusions in series with the macroscopic field, given that $\epsilon_{\text{eff}} E_0 = \epsilon_i E_i = \epsilon_j E_j\ \forall\, i, j$, we obtain the lower Wiener bound, $\epsilon_{\text{eff}} = (\sum p_d \epsilon_d^{-1})^{-1}$.

Under our assumptions $\vec{\nabla} \times \vec{E} \cong 0$, and we can use the Divergence Theorem to prove that

$$\int_V \vec{\nabla} U(\vec{r})\ d^3r = \oiint_S U(\vec{r})\ d\vec{S}. \tag{A.7}$$

We will prove the validity of (A.7). The Divergence Theorem [179] relates the volume integral of the divergence of a vector field and its surface integral over the boundary of the volume:

$$\int_V (\vec{\nabla} \cdot \vec{F}) d^3r = \oiint_{\delta V} \vec{F} \cdot d\vec{S}. \tag{A.8}$$

If the vector field $\vec{F} = v\,\vec{c}$, with $\vec{c} \neq 0$ a constant vector, then

$$\oiint_S \vec{F} \cdot d\vec{S} = \vec{c} \cdot \oiint_S v\; d\vec{S} \tag{A.9}$$

and, knowing that

$$\vec{\nabla} \cdot (v\,\vec{c}) = (\vec{\nabla} v) \cdot \vec{c} + v(\vec{\nabla} \cdot \vec{c}), \tag{A.10}$$

for any scalar field v and vector $\vec{c}$, we get

$$\begin{aligned} \int_V (\vec{\nabla} \cdot \vec{F})\; d^3r &= \int_V \vec{\nabla} \cdot (\vec{c}\, v)\; d^3r = \\ &= \int_V (\vec{\nabla} v \cdot \vec{c} + v \vec{\nabla} \cdot \vec{c})\; d^3r = \vec{c} \cdot \int_V \vec{\nabla} v\; d^3r. \end{aligned} \tag{A.11}$$

Equating (A.9) and (A.11) according to (A.8), we get

$$\vec{c} \cdot \oiint_S v\; d\vec{S} = \vec{c} \cdot \int_V \vec{\nabla} v\; d^3r \tag{A.12}$$

so

$$\vec{c} \cdot \left(\oiint_S v\; d\vec{S} - \int_V \vec{\nabla} v\; d^3r \right) = 0. \tag{A.13}$$

But $\vec{c} \neq 0$, and $\vec{c} \cdot \vec{f}(v)$ must vary with v, so that $\vec{c} \cdot \vec{f}(v)$ cannot always be 0. Therefore:

$$\oiint_S v\; d\vec{S} = \int_V \vec{\nabla} v\; d^3r. \tag{A.14}$$

Therefore, we can write (A.5) using the potential $U(\vec{r})$ at the domain boundaries as

$$\begin{aligned} \overleftrightarrow{\epsilon}_{\text{eff}}\; \vec{E}_0 = \overleftrightarrow{\epsilon}_{\text{eff}}\; \frac{1}{V_{\text{C}}} \oiint_{S_{\text{C}}} U(\vec{r})\, d\vec{S} = \\ = \sum_d p_d \epsilon_d \frac{1}{V_d} \oiint_{S_d} U(\vec{r})\, d\vec{S} = \sum_d \frac{1}{V_{\text{C}}} \epsilon_d \oiint_{S_d} U(\vec{r})\, d\vec{S} \end{aligned} \tag{A.15}$$

where the integral on the left is extended to the boundary of the unit cell, whereas those on the right are extended to the boundary of each connected domain d. This equation can be solved for the three orthonormal spatial directions to obtain the nine components of the effective permittivity.

If we consider N inclusions with the same permittivity ϵ_{i}, each occupying a volume enclosed by the surface S_n, embedded in a host matrix with permittivity ϵ_{m}, subjected to an unitary field in the direction $\hat{e}$, (A.15) becomes:

$$\overleftrightarrow{\epsilon}_{\mathrm{eff}}\ \hat{e} = \frac{1}{V_{\mathrm{C}}}\left[\epsilon_{\mathrm{i}}\sum_{n}^{N}\oiint_{S_n} U(\vec{r})\,d\vec{S} + \right. \tag{A.16}$$
$$\left. +\epsilon_{\mathrm{m}}\left(\oiint_{S_{\mathrm{C}}} U(\vec{r})\,d\vec{S} - \sum_{n}^{N}\oiint_{S_n} U(\vec{r})\,d\vec{S}\right)\right].$$

From (A.1) and (A.7) we have

$$\hat{e} = \frac{1}{V_{\mathrm{C}}}\oiint_{S_{\mathrm{C}}} U(\vec{r})\ d\vec{S} \tag{A.17}$$

and thus (A.16) is reduced to

$$\overleftrightarrow{\epsilon}_{\mathrm{eff}}\ \hat{e} = \epsilon_{\mathrm{m}}\hat{e} + \frac{\epsilon_{\mathrm{i}} - \epsilon_{\mathrm{m}}}{V_{\mathrm{C}}}\sum_{n}^{N}\oiint_{S_n} U(\vec{r})\,d\vec{S}. \tag{A.18}$$

Particularizing the macroscopic unit field $\hat{e}$ for the three orthonormal directions in space, we get from (A.18) nine equations for the nine components of $\overleftrightarrow{\epsilon}_{\mathrm{eff}}$. All we need is the potential at the boundaries of the N inclusions.

Our problem is reduced to the determination of the detailed potential at the boundary of the inclusions within the unit cell. This problem lacks general analytical solution. However, obtaining an arbitrarily good approximation for any given configuration is possible through numerical methods. Several models exist for the numerical estimation of specific configurations. The Helmholzt problem for the potential within the unit cell does not specify the fringing potentials at the transverse boundaries. Many of the methods in the literature subject the cell to a potential U_0 and impose a reflecting boundary potential ($\partial U/\partial n = 0$) on the free boundaries. This forces a tangential

electric field at the boundaries, which is only valid for some symmetrical configurations, depends on the choice of the unit cell, and excludes the possibility of fringing fields. The fringing fields depend on the spatial configuration of the cell, and should not be carelessly disregarded.

We tessellate the system with unit cells enclosed in parallelepipedic elements. In such a tessellation the effective permittivity tensor must be translation invariant. A plane wave propagating in the system satisfies the Bloch–Floquet [26, 78] conditions for the electric and magnetic fields, and under the LWA this leads to periodic Born–Von Karman fields at the boundaries [36].

Appendix B

Monte Carlo solution of elliptic PDE

We will treat the 2D case. Higher dimensional cases follow straightforwardly.

Let

$$af_{xx} + 2bf_{xy} + cf_{yy} + df_x + ef_y + F = 0 \tag{B.1}$$

be a partial differential equation (PDE) defined over a region R with boundary ∂R. The factors a, b, c, d, e, and F and the unknown function f are time-independent functions of (x, y), and $b^2 - ac < 0$ in R, i.e. (B.1) is elliptical.

We want to solve (B.1) subject to the boundary condition

$$f(x, y) = \phi(x, y) \quad \text{if} (x, y) \in \partial R \tag{B.2}$$

and possibly some local constraints.

The region R is divided into a regular mesh of step size h. Each interior point P_0 of the mesh has four neighbours P_1, P_2, P_3, and P_4. Let W_i be the random walk starting at P_0 and ending at the boundary point Q_i, constructed by moving away from P_0 following random directions generated by a random number generator until an absorbing point Q_i is hit. Absorbing points are the boundary points with fixed solution in (B.2). The probabilities governing the random walk depend on the local constrains on f.

Let

$$r(P) = 2\left[a(P) - b(P) + c(P)\right] + h\left[e(P) + d(P)\right] \tag{B.3}$$

for any point $P = (x, y)$ inside R, and let

$$Z(W_i) = \phi(Q_i) + h \sum_{P_j \in W_i} \frac{F(P_j)}{r(P_j)} \tag{B.4}$$

be the *primary estimator* of $f(P_0)$. The Monte Carlo solution of the elliptic equation (B.1) at point P_0 subject to (B.2) consists in generating a sequence of N independent random walks starting at P_0 and ending at different absorbing points, W_n. Then $Z(W_n)$ is evaluated for all n, and $f(P_0)$ is approximated by the *secondary estimator*

$$\theta = \frac{1}{N} \sum_{n=1}^{N} Z(W_n). \tag{B.5}$$

Proof that θ is a good approximation of f can be found in [212].

Appendix C

Exodus Monte Carlo

Estimating the potential at the boundaries of the inclusions in the unit cell can be a costly procedure. Just think of a body centered cubic (BCC) unit cell (see Appendix E) with cubic inclusions occupying a volume fraction $p = 10\%$, covered with a 100^3 mesh. Nearly 17 000 nodes (1.64%) are situated on inclusion boundaries. If Monte Carlo random walk is used, it implies 17 000 nodes sourcing N random walkers each. N determines the precision of the estimations and should be high enough. If we choose $N = 10^6$ walkers, we would need 1.7×10^{10} random walks for each direction, i.e. 5.1×10^{10} random walks to get the three equations like (1.8). Each random walk requires following a random walker from node to node according to the probabilities (1.5) until it reaches a fixed potential node. This may take a long walk, depending on the spatial configuration of the unit cell and the distance from the source node to the fixed potential cell facets. This renders a straightforward implementation of the Monte Carlo solution too costly for more than a very few boundary nodes.

The alternative is substituting the N sequential random walks sourcing from each boundary node by a single run of the Exodus method, first proposed by Emery and Carson [73]. Exodus is a probabilistic Monte Carlo method not subjected to randomness, and thus has the additional advantage of not needing a pseudo-random number generator, producing implementation independent solutions at least as accurate as those provided by finite differences and finite elements methods [213].

The idea is very simple: Instead of tracking a single random walker at a time, the whole population of N random walkers moves at once. From an initial state in which all of the N walkers are concentrated at the node of interest, a proportion of walkers equal to the corresponding probability of transition moves in each iteration from the current node to each neighbour. This requires scanning the whole mesh in each iteration, distributing the walkers in each node between its neighbours according to the probabilities s_ν in (1.5), except for the fixed potential nodes k, which act as sinks. Walkers entering fixed potential nodes get trapped.

The procedure goes on until a preestablished proportion $\eta \to 1$ of the initial N walkers has been trapped into fixed potential nodes. Thus we get the n_k in (1.7) for a given source node i in the same number of iterations that would need a single walk of length $\eta E\{l_{\max}\}$, with $l_{\max}$ the longest of N random walks sourcing from node i. The number of iterations to obtain the potential at node i decreases by a factor of $\frac{\eta}{N}\frac{E\{l_{\max}\}}{\mu_l}$, where $N \gg \eta < 1$ and $E\{l_{\max}\} > \mu_l$, the average length of one random walk sourcing from node i. Note that the dependence of the speed of Exodus on N is indirect, through the effect of sample size on the expected value of $l_{\max}$.

The impressive reduction in the number of iterations per node is counterbalanced by the fact that now each iteration requires scanning the whole mesh to redistribute the entire population of walkers, while with typical Monte Carlo each iteration requires the generation of a pseudorandom number to move only one walker from a node to one of its neighbours. Therefore, Exodus is an efficient alternative to traditional random walk if N is greater than L^3, the number of nodes in the mesh, at least one order of magnitude more than the expected value of the longest walk is greater than the average walk. Provided that N depends on L, because it has to assure average significant values of n_k in (1.7) for all $2L^2$ nodes in the fixed potential faces, the above assumption ultimately depends on the size of the mesh, L^3, and on the spatial arrangement of the inclusions, which govern the random walks.

Appendix D

Explicit algorithms for the computation of chain codes with SEE

But in the detail which he gave you of them
he could not sum up the hours and months of misery
which I endured wasting in impotent passions.

Algorithm 23 Robust 4-neighbourhood SEE Chain Code (R-SCC$_4$)

---

initialize *Code*
$d \leftarrow 0$
$x \leftarrow x_\mathrm{i} + 1$, $y \leftarrow y_\mathrm{i}$
do
  $c \leftarrow c_{x,y}$
  if $c = 6$ {**if** $d = 0$ $c \leftarrow 2$, **else if** $d = 1$ $c \leftarrow 4$}
  else if $c = 9$ {**if** $d = 2$ $c \leftarrow 8$, **else if** $d = 3$ $c \leftarrow 1$}
  append c to *Code*
  if $(c \wedge 05\mathrm{h}) = 01\mathrm{h}$ {$d \leftarrow 0$, **increase** x}
  else if $(c \wedge 0\mathrm{Ah}) = 08\mathrm{h}$ {$d \leftarrow 1$, **decrease** x}
  else if $(c \wedge 03\mathrm{h}) = 02\mathrm{h}$ {$d \leftarrow 2$, **increase** y}
  else if $(c \wedge 0\mathrm{Ch}) = 04\mathrm{h}$ {$d \leftarrow 3$, **decrease** y}
while $(x, y) \neq (x_\mathrm{i}, y_\mathrm{i})$

Algorithm 24 Robust 8-neighbourhood SEE Chain Code (R-SCC$_8$)

initialize $Code$
$d \leftarrow 0$
$x \leftarrow x_\mathrm{i} + 1$, $y \leftarrow y_\mathrm{i}$
do
  append $c_{x,y}$ to $Code$
  if ($c_{x,y} = 6$ and $p = 0$) {$d \leftarrow 3$, **decrease** y}
  else if ($c_{x,y} = 9$ and $p = 3$) {$d \leftarrow 1$, **decrease** x}
  else if ($c_{x,y} \wedge 05\mathrm{h}) = 01\mathrm{h}$ {$d \leftarrow 0$, **increase** x}
  else if ($c_{x,y} \wedge 0\mathrm{Ah}) = 08\mathrm{h}$ {$d \leftarrow 1$, **decrease** x}
  else if ($c_{x,y} \wedge 03\mathrm{h}) = 02\mathrm{h}$ {$d \leftarrow 2$, **increase** y}
  else if ($c_{x,y} \wedge 0\mathrm{Ch}) = 04\mathrm{h}$ {$d \leftarrow 3$, **decrease** y}
while $(x, y) \neq (x_\mathrm{i}, y_\mathrm{i})$

Algorithm 25 4-neighbourhood Crack Code using SEE

initialize $Code$
$d \leftarrow 0$
$x \leftarrow x_\mathrm{i}$, $y \leftarrow y_\mathrm{i}$
do
  $c \leftarrow c_{x,y}$
  if $c = 6$ {**if** $d = 1$ $c \leftarrow 2$, **else if** $d = 3$ $c \leftarrow 4$}
  else if $c = 9$ {**if** $d = 2$ $c \leftarrow 8$, **else if** $d = 0$ $c \leftarrow 1$}
  if ($c \wedge 05\mathrm{h}) = 01\mathrm{h}$ {$d \leftarrow 1$, **increase** x}
  else if ($c \wedge 0\mathrm{Ah}) = 08\mathrm{h}$ {$d \leftarrow 3$, **decrease** x}
  else if ($c \wedge 03\mathrm{h}) = 02\mathrm{h}$ {$d \leftarrow 2$, **increase** y}
  else if ($c \wedge 0\mathrm{Ch}) = 04\mathrm{h}$ {$d \leftarrow 0$, **decrease** y}
  append d to $Code$
while $(x, y) \neq (x_\mathrm{i}, y_\mathrm{i})$

Algorithm 26 8-neighbourhood Crack Code using SEE

initialize *Code*
$d \leftarrow 0$
$x \leftarrow x_\mathrm{i}$, $y \leftarrow y_\mathrm{i}$
do
  if $(c_{x,y} = 6$ and $p = 1)$ $\{d \leftarrow 0$, **decrease** $y\}$
  else if $(c_{x,y} = 9$ and $p = 0)$ $\{d \leftarrow 3$, **decrease** $x\}$
  else if $(c_{x,y} \wedge 05\mathrm{h}) = 01\mathrm{h}$ $\{d \leftarrow 1$, **increase** $x\}$
  else if $(c_{x,y} \wedge 0\mathrm{Ah}) = 08\mathrm{h}$ $\{d \leftarrow 3$, **decrease** $x\}$
  else if $(c_{x,y} \wedge 03\mathrm{h}) = 02\mathrm{h}$ $\{d \leftarrow 2$, **increase** $y\}$
  else if $(c_{x,y} \wedge 0\mathrm{Ch}) = 04\mathrm{h}$ $\{d \leftarrow 0$, **decrease** $y\}$
  append d to *Code*
while $(x, y) \neq (x_\mathrm{i}, y_\mathrm{i})$

Algorithm 27 Compact 4-neighbourhood SEE Chain Code (SCC_4)

initialize *Code*
$d \leftarrow 0$
$x \leftarrow x_\mathrm{i} + 1$, $y \leftarrow y_\mathrm{i}$
do
  $c \leftarrow c_{x,y}$
  if $c = 6$ {**if** $d = 0$ $c \leftarrow 2$, **else if** $d = 1$ $c \leftarrow 4$}
  else if $c = 9$ {**if** $d = 2$ $c \leftarrow 8$, **else if** $d = 3$ $c \leftarrow 1$}
  if $d = 0$ $\{n \leftarrow c \wedge 01\mathrm{h}$, $n \leftarrow n \vee ((c >> 1) \wedge 01\mathrm{h})\}$
  else if $d = 1$ $\{n \leftarrow (c >> 1) \wedge 01\mathrm{h}$, $n \leftarrow n \vee ((c >> 2) \wedge 02\mathrm{h})\}$
  else if $d = 2$ $n \leftarrow c \wedge 03\mathrm{h}$
  else if $d = 3$ $\{n \leftarrow (c >> 2) \wedge 03\mathrm{h}\}$
  append n to *Code*
  if $(c \wedge 05\mathrm{h}) = 01\mathrm{h}$ $\{d \leftarrow 0$, **increase** $x\}$
  else if $(c \wedge 0\mathrm{Ah}) = 08\mathrm{h}$ $\{d \leftarrow 1$, **decrease** $x\}$
  else if $(c \wedge 03\mathrm{h}) = 02\mathrm{h}$ $\{d \leftarrow 2$, **increase** $y\}$
  else if $(c \wedge 0\mathrm{Ch}) = 04\mathrm{h}$ $\{d \leftarrow 3$, **decrease** $y\}$
while $(x, y) \neq (x_\mathrm{i}, y_\mathrm{i})$

Algorithm 28 Compact 8-neighbourhood SEE Chain Code (SCC_8)

initialize $Code$
$d \leftarrow 0$
$x \leftarrow x_\mathrm{i} + 1$, $y \leftarrow y_\mathrm{i}$
do
  if $d = 0$ $\{n \leftarrow c_{x,y} \wedge 01\mathrm{h}$, $n \leftarrow n \vee ((c_{x,y} >> 1) \wedge 01\mathrm{h})\}$
  else if $d = 1$ $\{n \leftarrow (c_{x,y} >> 1) \wedge 01\mathrm{h}$, $n \leftarrow n \vee ((c_{x,y} >> 2) \wedge 02\mathrm{h})\}$
  else if $d = 2$ $n \leftarrow c_{x,y} \wedge 03\mathrm{h}$
  else if $d = 3$ $\{n \leftarrow (c_{x,y} >> 2) \wedge 03\mathrm{h}\}$
  append n to $Code$
  if ($c_{x,y} = 6$ and $p = 0$) $\{d \leftarrow 3$, **decrease** $y\}$
  else if ($c_{x,y} = 9$ and $p = 3$) $\{d \leftarrow 1$, **decrease** $x\}$
  else if $(c_{x,y} \wedge 05\mathrm{h}) = 01\mathrm{h}$ $\{d \leftarrow 0$, **increase** $x\}$
  else if $(c_{x,y} \wedge 0\mathrm{Ah}) = 08\mathrm{h}$ $\{d \leftarrow 1$, **decrease** $x\}$
  else if $(c_{x,y} \wedge 03\mathrm{h}) = 02\mathrm{h}$ $\{d \leftarrow 2$, **increase** $y\}$
  else if $(c_{x,y} \wedge 0\mathrm{Ch}) = 04\mathrm{h}$ $\{d \leftarrow 3$, **decrease** $y\}$
while $(x, y) \neq (x_\mathrm{i}, y_\mathrm{i})$

Appendix E

Some numerical results of the estimation of effective responses

I was required to exchange
chimeras of boundless grandeur
for realities of little worth.

To exemplify the effectivity of the method underlying this report, and the application of the techniques developed for cluster labelling (connected component labelling) and local configuration characterization, some examples of the estimation of the effective permittivity of 2D and 3D regular composites follow.

E.1 2D models

The effective permittivity of a 2D arrangement of elliptical inclusions with $\epsilon_{\mathrm{i}} = 10$ regularly distributed in vacuum ($\epsilon_0\,\mathrm{Fm}^{-1}$) with constant volume fraction $p = 0.20$ was obtained, using a 300^2 mesh for the unit cell enclosing one ellipse. Fig. E.1 shows how the component of $\overleftrightarrow{\epsilon}_{\mathrm{eff}}$ in the direction of the macroscopic field $\vec{E}_0$ varies with the aspect ratio of the ellipses for a set of different orientations: aligned (0°), oblique (45°), and transverse

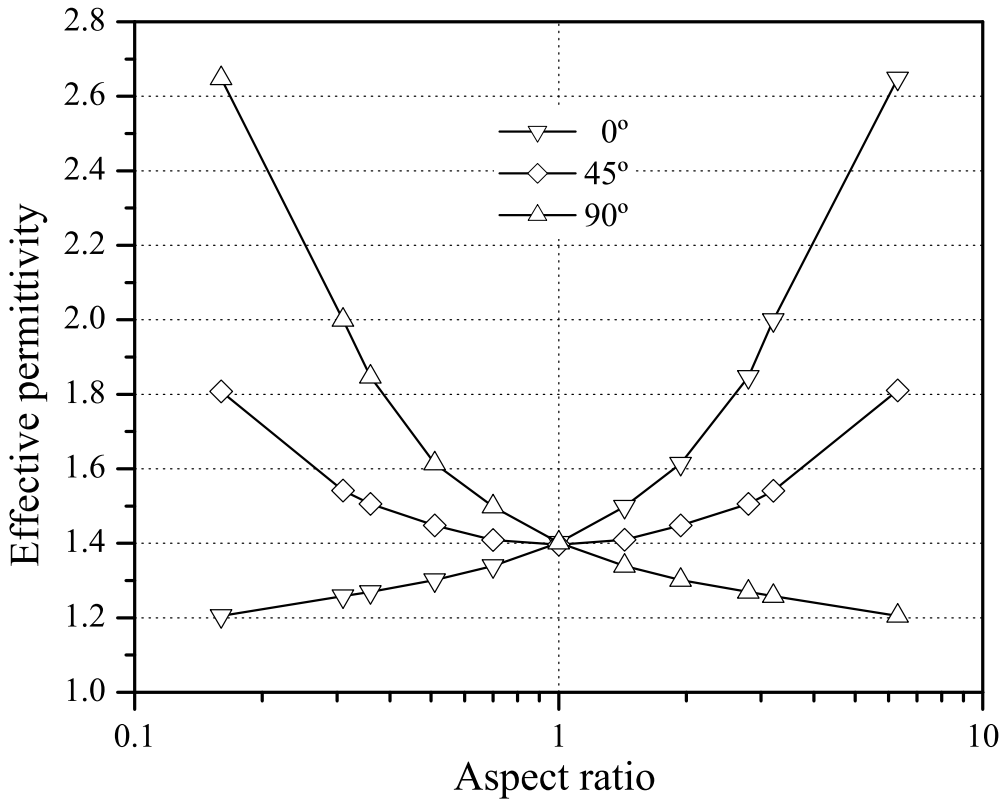


Figure E.1. *Longitudinal component of the effective permittivity of an elliptical inclusion ($\epsilon_{\mathrm{i}} = 10$) in vacuum with varying aspect-ratio (log scale).*

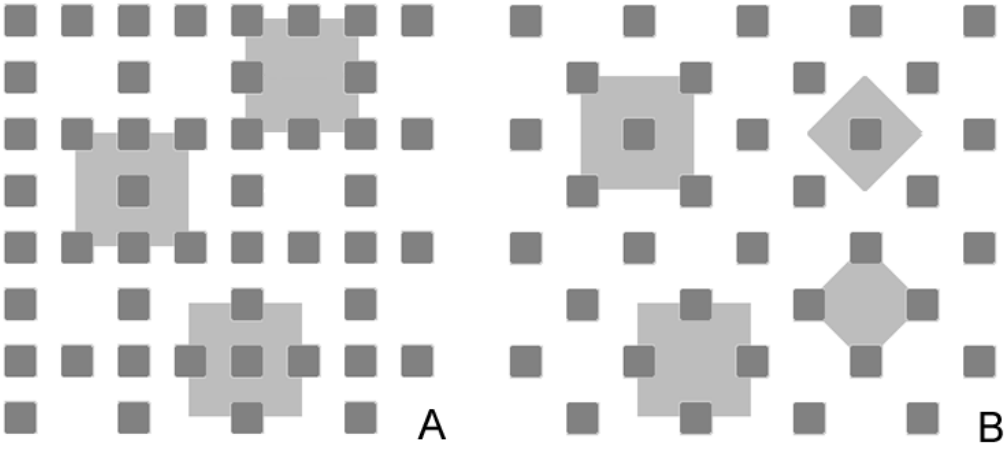


Figure E.2. *Different unit cells (light gray) in two different 2D configurations.*

(90°) with respect to $\vec{E}_0$. The results are in agreement with the expected values. First, the perfect symmetry validates the implementation. Second, the more elongated the inclusions in the direction of the field, the higher the permittivity, as expected from a configuration that approaches longitudinal percolation [69], when the permittivity tends to the upper Wiener bound (parallel arrangement of stratified inclusions).

The method was also applied to a set of regular 2D arrangements, Fig. E.2, of inclusions ($\epsilon_\mathrm{i} = 6$) in vacuum, to study the effect of fractional volume, unit cell, spatial arrangement, and inclusion shape. The effective response of a nonhomogeneous medium should be independent of the selection of the unit cell. Fig. E.2 shows several different square unit cells for two different 2D arrangements. Fig. E.3 details the unit cells shown in Fig. E.2, with circular inclusions. Fig. E.4 shows the effective permittivity of each arrangement with circular inclusions occupying fractional volumes from 0.05 to 1.00, obtained with the different unit cells, using a 300^2 mesh. As expected, no perceptible variation between unit cells in each configuration was found.

Fig. E.4 also shows the Hashin–Shtrikman (H-S) bounds, and the percolation threshold p_th for each of the configurations. I use the term percolation threshold in the sense of Stauffer [233]: The critical threshold of lattice cells

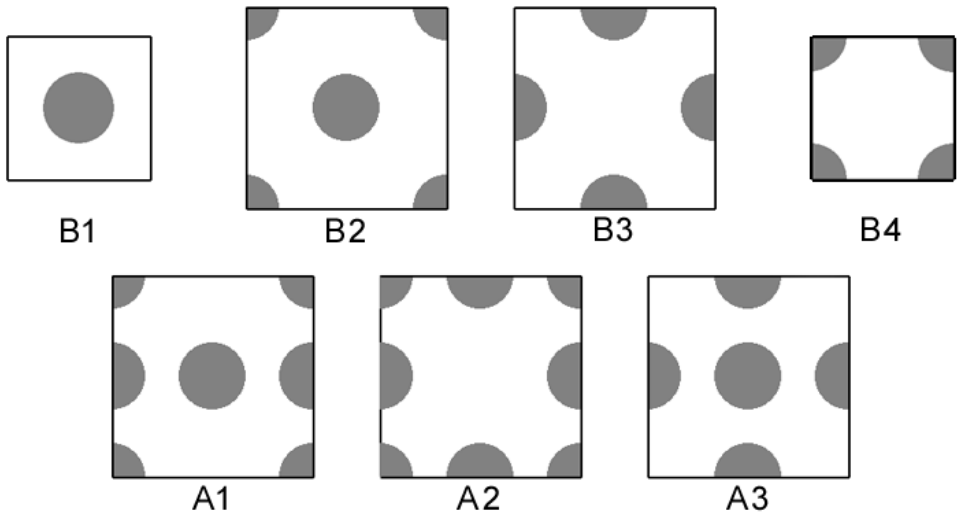


Figure E.3. *Different units cells in the 2D arrangements of Fig. E.2 with circular inclusions.*

that must be filled to create a continuous path of nearest neighbours from one side to another. A somewhat different, but related, definition of percolation threshold can be found in the literature [227]: The abrupt change in the behavior of a certain parameter in a random medium as the volume fraction changes. The samples here are not random, and thus the latter is not applicable. However, percolation induces the same kind of behavior in our samples. Its effect on the effective permittivity is clearly shown in the graph, as reflected by the different behavior of configurations with different percolation thresholds ($p_{\text{th}} = 0.58$ for arrangement A and 0.78 for B, with circular inclusions). The effective permittivity moves from the lower H-S bound to the upper H-S bound with increasing fractional volume, with a greater slope in the region of the percolation threshold. Recall that the H-S lower bound is the Maxwell Garnett formula, exact for low concentrations of spheres. Simulations above the percolation threshold are possible because inclusions are allowed to overlap and form a single cluster. The computation of the volume fraction is made with the real occupancy, not with the volume of individual, overlapped inclusions.

Fig. E.5 shows the influence of inclusion shape on the permittivity of configuration B. Again percolation seems to be the major factor affecting

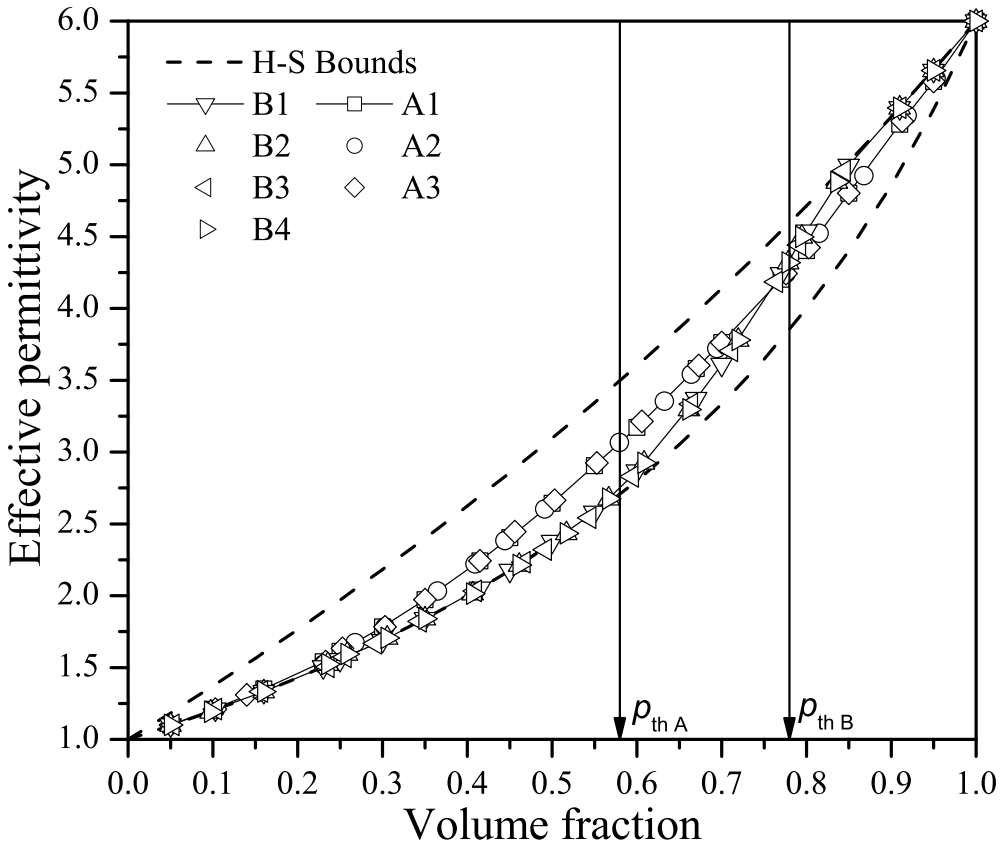


Figure E.4. *Effective permittivity of circular inclusions ($\epsilon_i = 6$) in vacuum in the 2D arrangements of Fig. E.2 computed with the unit cells of Fig. E.3. The dashed lines are the Hashin-Shtrikman bounds. The vertical arrows point out the corresponding percolation thresholds.*

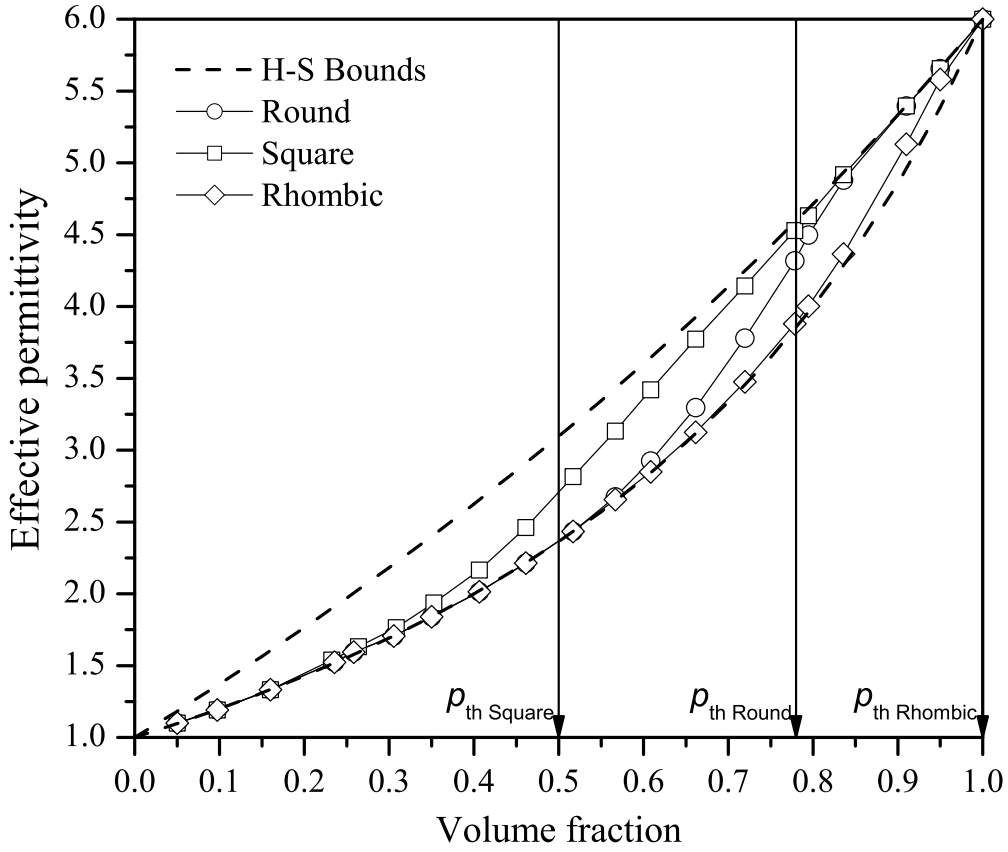


Figure E.5. *Effective permittivity of arrangement B in Fig. E.2 with inclusions of different shape. The vertical arrows point out the corresponding percolation thresholds.*

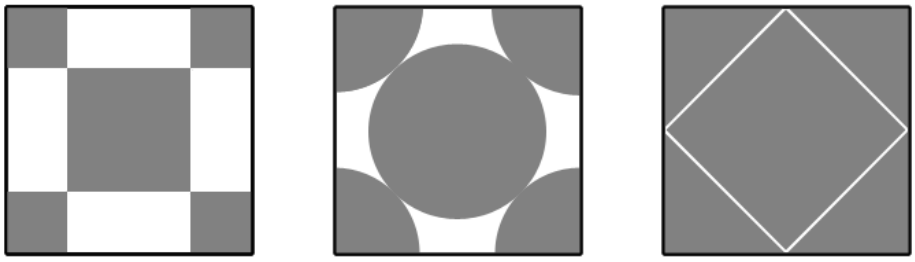


Figure E.6. *The shape of the inclusions influences the percolation threshold in the same 2D arrangement. From left to right: 0.50, 0.78, and 1.00.*

permittivity, as the different percolation thresholds ($p_{\text{th}} = 0.50$ for squares, 0.78 for circles, and 1.00 for rhombuses, see Fig. E.6) determine the transit from the lower to the upper H-S bound.

E.2 3D models

The method was also tested on a Body Centered Cubic (BCC) and a Face Centered Cubic (FCC) 3D configuration (see Figs. E.7 and E.8). In Fig. E.7, the unit cell on the left contains $\frac{1}{8}$ of inclusion in each corner plus a whole inclusion in the center. The unit cell on the right contains $\frac{1}{4}$ of inclusion in each of four parallel edges, and a half inclusion in each of the two opposite faces. In Fig. E.8, the unit cell on the left contains $\frac{1}{4}$ of inclusion in each edge, plus a whole inclusion in the center. The unit cell on the right contains $\frac{1}{8}$ of inclusion in each corner plus a half inclusion in each face.

The results shown in Figs. E.9 and E.10 were obtained using a 50^3 mesh for the unit cells in Figs. E.7 and E.8, for a range of volume fractions between 0.01 and 1.00. Again no difference between unit cells in each arrangement was found. The corresponding percolation thresholds with cubic inclusions are 0.25 and 0.50 for the BCC and the FCC arrangement, respectively. The permittivity curves follow the same behavior as that of the 2D arrangements. The BCC arrangement, with lower p_{th}, grows faster at the beginning than the FCC curve. The change in the FCC curve happens later, but it is more

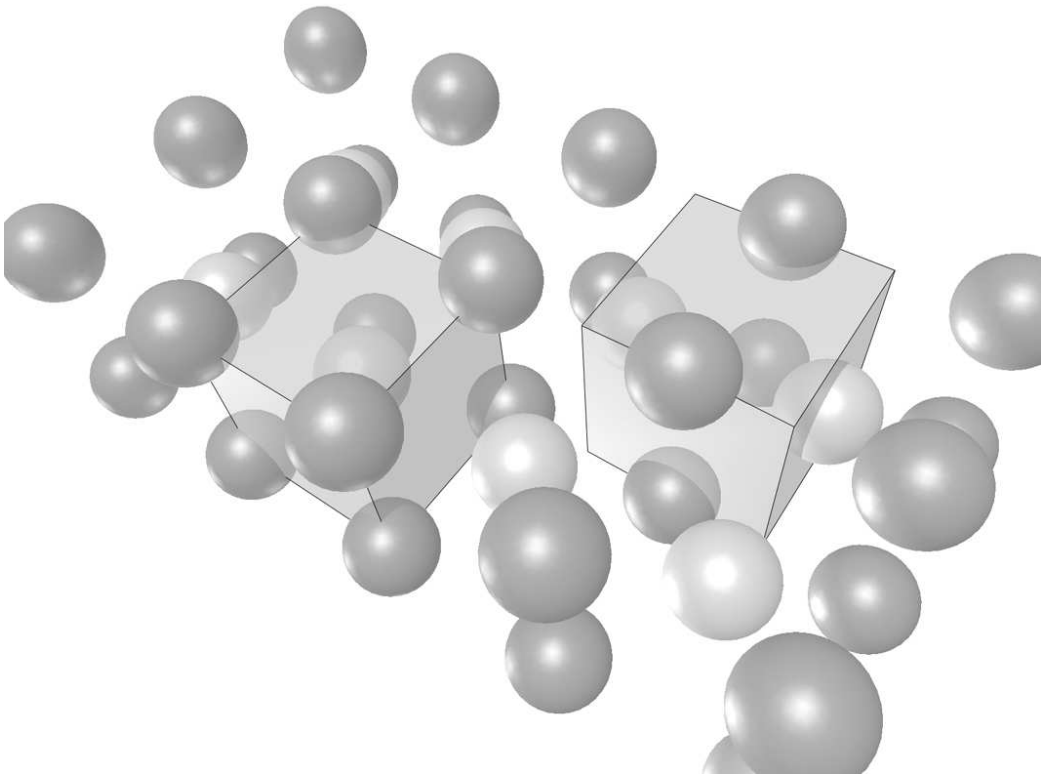


Figure E.7. *A BCC configuration and two different unit cells. Inclusions in different layers have been rendered with different color.*

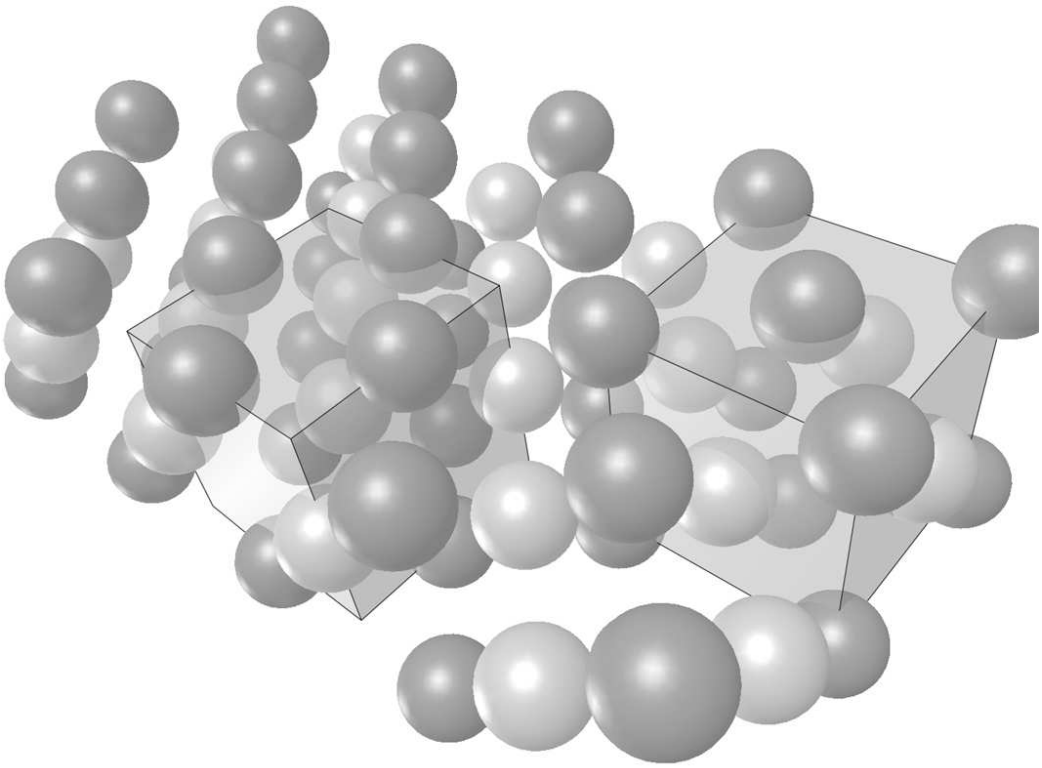


Figure E.8. *A FCC configuration and two different unit cells. Inclusions in different layers have been rendered with different color.*

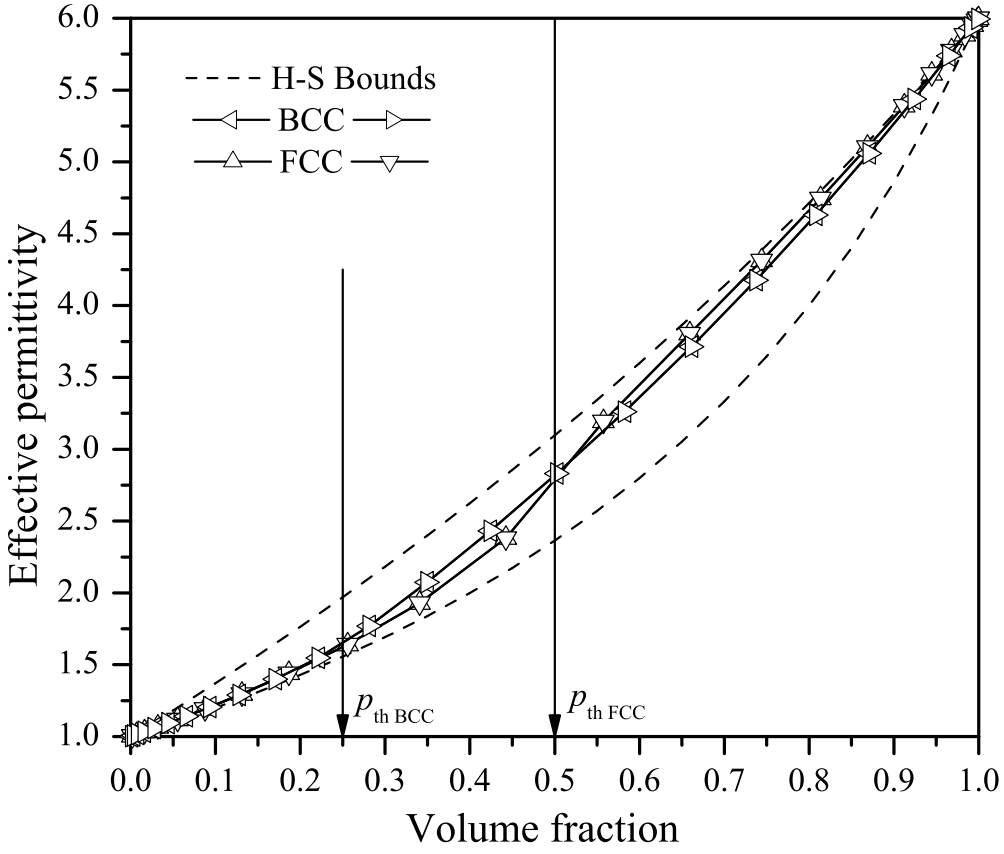


Figure E.9. *Effective permittivity of a BCC and a FCC 3D arrangement of cubic inclusions with the unit cells of Figs. E.7 and E.8. The vertical arrows point out the corresponding percolation thresholds.*

abrupt, and, as a result, the permittivity of the FCC configuration is slightly greater for volume fractions above 0.50. Both configurations leave the lower H–S bound quite soon, in support of the hypothesis that the percolation threshold strongly affects this transition. The curve of the 2D arrangement A of circular inclusions in Fig. E.4, and that of the 2D arrangement B of square inclusions in Fig. E.5, which have the lower of the 2D percolation thresholds ($p_{\text{th}} = 0.58$ and 0.50, respectively) are also the first leaving the lower H–S bound. Note the remarkable similarity between the curve with square inclusions in Fig. E.5 and the FCC curve in Fig. E.9. This is in agreement with the similarity between the B arrangement in Fig. E.2 and a FCC(100) surface plane.

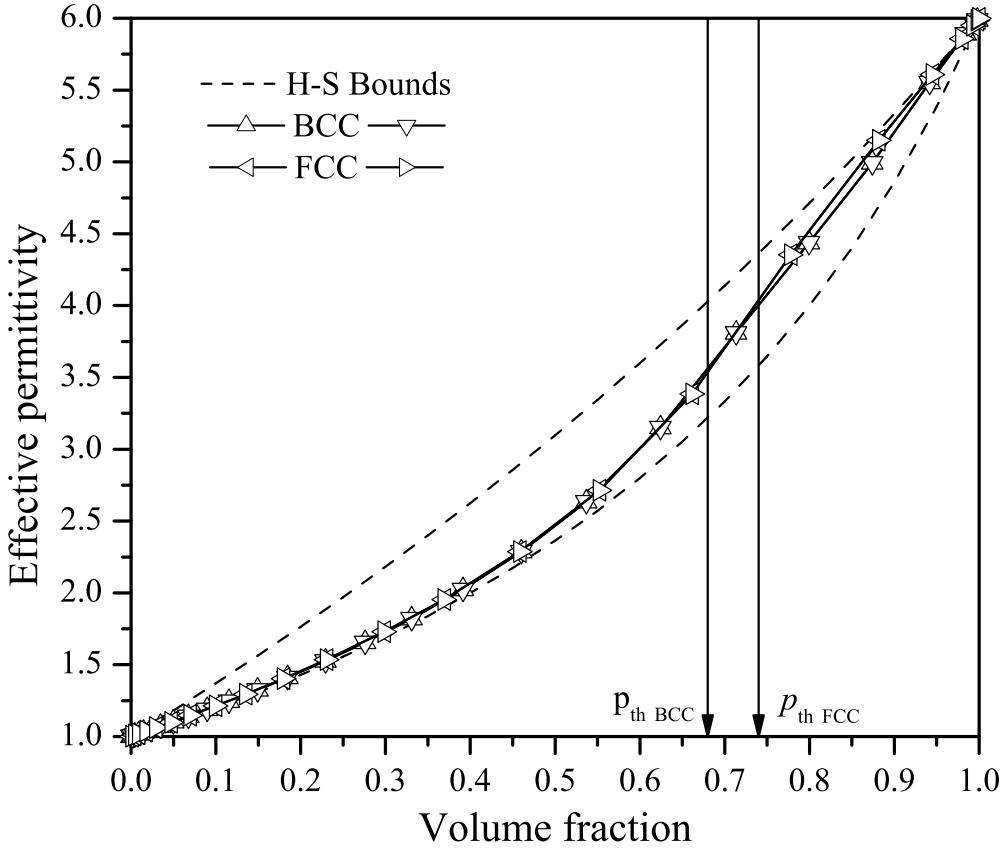


Figure E.10. *Effective permittivity of a BCC and a FCC 3D arrangement of spheric inclusions with the unit cells of Figs. E.7 and E.8. The vertical arrows point out the corresponding percolation thresholds.*

Results in Fig. E.10 are similar. The percolation thresholds with spheres are $p_{\mathrm{th}} = 0.68$ and 0.74 for the BCC and the FCC arrangement, respectively. They are quite close, and both curves run quite near to each other. Again at the higher volume fractions, the FCC arrangement seems to have slightly higher permittivity. The FCC curve again resembles very closely that of the B arrangement, this time with circular inclusions, in Fig. E.5. There is also an excellent agreement with the Maxwell Garnett formula at low volume fractions. Recall that the lower H–S bound is exact for sparse arrangements of spheres.

Acknowledgments

But success shall *crown my endeavours.*
Wherefore not? Thus far I have gone,
tracing a secure way over the pathless seas,
the very stars themselves being witnesses
and testimonies of my triumph.
Why not still proceed
over the untamed yet obedient element?
What can stop the determined heart
and resolved will of man?

Financial support was provided by European Union ERDF funds through the program Interreg IIIB *Atlantic Area* within the project "Multi and Hyperspectral Imagery from Acquisition to Decision Making in Environmental Management". Additional funding was provided by Secretaría Xeral de I+D (Xunta de Galicia), and University of Vigo (Spain).

The Spanish Ministry of Science and Technology (CICyT) and Tacore S.L. provided financial support for the development of the real-time machine vision system in section 2.4.1. Conservas Albo provided countless samples and allowed repeated prototype testing in their production lines.

Prof. G.L. Vignoles and O. Coindreau (LCTS), Snecma Propulsion Solide, and the ESRF (European Synchrotron Radiation Facility) ID19 team provided the composite material 3D data in section 2.4.2.

Bibliography

But here were books, and here were men
who had penetrated deeper and knew more.
I took their word for all that they averred,
and I became their disciple.

[1] A. K. Agrawala and A. V. Kulkarni, 1977. "A sequential approach to the extraction of shape features", *Computer Graphics and Image Processing*, 6:538–557.

[2] D. W. Aha, D. Kibler, M. Albert, 1991. "Instance-based learning algorithms", *Machine Learning*, 6:37–66.

[3] A. Aho, J. Hopcroft, J. Ullman, 1979. *Design and Analysis of Computer Algorithms*. Reading, MA: Addison Wesley.

[4] A. Al-Futaisi and T. W. Patzek, 2003. "Extension of Hoshen-Kopelman algorithm to non-lattice environments", *Physica A*, 321:665.

[5] H. M. Alnuweiri and V. K. Prasanna, 1992. "Parallel architectures and algorithms for image component labelling", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14:1014.

[6] F. Babalievski, 1998. "Cluster counting: The Hoshen-Kopelman algorithm vs. spanning tree approaches", *International Journal of Modern Physics C*, 9:43.

[7] N. S. Bakhvalov and G. P. Panasenko, 1989. *Homogenization: Averaging Processes in Periodic Media*. Dordrecht: Kluwer Academic Publishers Group.

[8] N. S. Bakhvalov and A. Knyazev, 1994. "Fictitious domain methods and computation of homogenized properties of composites with a periodic structure of essentially different components", in G. I. Marchuk

(Ed.), *Numerical Methods and Applications*. Boca Raton, FL: CRC Press. 221–276.

[9] A. Y. A. Beliaev and S. M. Kozlov, 1996. "Darcy equation for random porous media", *Communications on Pure and Applied Mathematics*, 49(1):1–34.

[10] A. Bensoussan, J. L. Lions, G. Papanicolaou, 1978. *Asymptotic analysis of periodic structures*. Amsterdam: North-Holland Publishing.

[11] V. Berdichevsky, I. Kunin, F. Hussain, 1991. "Negative temperature of vortex motion", *Physical Review A (Atomic, Molecular, and Optical Physics)*, 43(4):2050–2051.

[12] V. Berdichevsky, 1997. *Thermodynamics of chaos and order*. Harlow: Longman.

[13] V. Berdichevsky, V. Jikov, G. Papanicolaou, 1999. *Homogenization*. Philadelphia: World Scientific Publishing.

[14] D. Bergman, 1978. "The dielectric constant of a composite material —A problem in classical physics", *Physics Reports*, 43:377–407.

[15] L. Berlyand and K. Golden, 1994. "Exact result for the effective conductivity of a continuum percolation model", *Physical Review B (Solid State)*, 50:2114–2117.

[16] A. Beroual, C. Brosseau, A. Boudida, 2000. "Permittivity of lossy dielectric heterostructures. Effect of shape anisotropy", *Journal of Physics D: Applied Physics*, 33:1969–1974.

[17] A. Beroual, A. Boudida, C. Brosseau, 2000. "How shape anisotropy and spatial orientation of the constituents affect the permittivity of dielectric heterostructures?", *Journal of Applied Physics*, 88:7278–7288.

[18] A. Beroual and C. Brosseau, 2001. "Comparison of dielectric properties determined from a computational approach and experiment for anisotropic and periodic heterostructures", *IEEE Transactions on Dielectrics and Electrical Insulators*, 8(6):921–929.

[19] S. Berthier, 1993. *Optique des Milieux Composites*. Paris: Ed. Polytechnica.

[20] T. M. Besmann, 1990. "Processing Science for Chemical Vapor Infiltration", *High Temperature Materials Laboratory Industry/Government Briefing*. Oak Ridge: Oak Ridge National Laboratory.

[21] R. Bickerdike, A. Brown, G. Huges, H. Ranson, 1962. "The deposition of pyrolytic carbon in the pores of bonded and unbonded carbon powders", *Proceedings of the Fifth AMCS Biennial Conference on Carbon*, 575–582.

[22] J. Bigün, G. H. Granlund, J. Wiklund, 1991. "Multidimensional orientation estimation with applications to texture analysis and optical flow", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(8):775–790.

[23] J. R. Birchak, C. G. Gardner, J. E. Hipp, J. M. Victor, 1974. "High dielectric constant microwave probes for sensing soil moisture", *Proceedings IEEE*, 62:93–98.

[24] M. J. Black, G. Sapiro, D. H. Marimont, D. Heeger, 1998. "Robust anisotropic diffusion", *IEEE Transactions on Image Processing*, 7:421–432.

[25] R. Blanc, C. Germain, J.P. Da Costa, P. Baylou, M. Cataldi, 2006. "Fiber orientation measurements in composite material", *Composites Part A: Applied Science and Manufacturing*, in press.

[26] F. Bloch, 1928. "Über die Quantenmechanik der Electronen in Kristallgittern", *Zeitschrift fur Physik*, 52:555–600.

[27] A. Blumenkrans, 1991. "Two-dimensional object recognition using a two-dimensional polar transform", *Pattern Recognition*, 24:879–890.

[28] C. F. Bohren and D. R. Huffman, 1983. *Absorption and Scattering of Light by Small Particles*. New York: Wiley. 149.

[29] C. J. F. Böttcher, 1952. *Theory of Electric Polarization*. Chicago: Elsevier Publishing.

[30] C. J. F. Böttcher and P. Bordewijk, 1978. *Theory of Electric Polarization*, vol. 2. Amsterdam: Elsevier Publishing.

[31] A. Bourgeat, S. M. Kozlov, A. Mikelić, 1995. "Effective equations of two-phase flow in random media", *Calculus of Variations and Partial Differential Equations*, 3(3):385–406.

[32] E. Bribiesca and A. Guzmán, 1980. "How to describe pure form and how to measure differences in shapes using shape numbers", *Pattern Recognition*, 12:101–112.

[33] E. Bribiesca, 1992. "A geometric structure for two-dimensional shapes and three-dimensional surfaces", *Pattern Recognition*, 25:483–496.

[34] E. Bribiesca, 1999. "A new chain code", *Pattern Recognition* 32:235–251.

[35] C. R. Brice and C. L. Fennema, 1970. "Scene analysis using regions", *Artificial Intelligence*, 1(3-4):205–226.

[36] L. Brillouin, 1946. *Wave propagation in Periodic Structures*. New York: McGraw-Hill.

[37] P. Brodatz, 1966. *Textures: A Photographic Album for Artists and Designers*. New York: Dover.

[38] C. Brosseau and A. Beroual, 2001. "Effective permittivity of composite with stratified particles", *Journal of Physics D: Applied Physics*, 34:704–710.

[39] D. A. G. Bruggeman, 1935. "Berechnung verschiedener physicalischer Konstanten von heterogenen Substanzen I. Dielectrizitatkonstanten und Leitfähigkeiten der Mischkörper aus isotropen Substanzen", *Annalen der Physik*, 22:636–679.

[40] D. A. G. Bruggemann, 1937. "Berechnung verschiedener physikalischer Konstanten von heterogenen Substanzen III. Die elastische Konstanten der Quasiisotropen Mischkörper aus isotropen Substanzen", *Annalen der Physik*, 29:160–178.

[41] J.D. Buckley, 1988. "Carbon-Carbon, an overview", *Ceramic Bulletin*, 67(2):364–368.

[42] J. D. Buckley and D. D. Edie, 1993. *Carbon-Carbon Materials and Composites*. Parkridge, NJ: William Andrew Publishing/Noyes.

[43] H. Bunke and U. Bühler, 1993. "Applications of approximate string matching to 2D shape recognition", *Pattern Recognition*, 26:1797–1812.

[44] H. Bunke and M. Zumbühl, 1999. "Acquisition of 2D shape models from scenes with overlapping objects using string matching", *Pattern Analysis and Applications*, 2:2–9.

[45] F. Catté, P.-L. Lions, J.-M. Morel, T. Coll, 1992. "Image selective smoothing and edge detection by nonlinear diffusion", *SIAM Journal on Numerical Analysis*, 29:182–193.

[46] I. Chakravarty, 1981. "A Single-pass, chain generating algorithm for region boundaries", *Computer Graphics and Image Processing*, 15:182–193.

[47] F. Chang, C. J. Chen, C. J. Lu, 2004. "A linear-time component-labeling algorithm using contour tracing technique", *Computer Vision and Image Understanding*, 93:206–220.

[48] W. Chang and J. Lampe, 1992. "Theoretical and empirical comparisons of approximate string matching algorithms", *Lecture Notes in Computer Science*, 644:172–181.

[49] W. Chang and E. Lawler, 1994. "Sublinear approximate string matching and biological applications", *Algorithmica*, 12(4/5):327–344.

[50] L. T. Chen, L. S. Davis, C. P. Kruskal, 1994. "Efficient parallel processing of image contours", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(1):113–129.

[51] S. W. Chen, S. T. Tung, C. Y. Fang, S. Cherng, A. Jain, 1998. "Extended attributed string matching for shape recognition", *Computer Vision and Image Understanding*, 70:36–50.

[52] H. Cheng and S. Torquato, 1997. "Electric field fluctuation in random dielectric composites", *Physical Review B*, 156(13):8060–8068.

[53] A. V. Cherkaev and L. Slepyan, 1995. "Waiting element structures and stability under extension", *International Journal of Damage Mechanics*, 4:58–82.

[54] W. C. Chew, 1995. *Waves and Fields in Inhomogeneous Media*. New York: IEEE Press.

[55] T. C. Choy, 1999. *Effective Medium Theory*. New York: Oxford University Press.

[56] R. M. Christensen, 1979. *Mechanics of Composite Materials*. New York: Wiley-Interscience.

[57] F. Christin, 2002. "Design, fabrication and application of thermostructural composites like C/C, C/SiC and SiC/SiC composites", *Advanced Engineering Materials*, 4(12):903–912.

[58] D. Cioranescu and F. Murat, 1997. "A strange term coming from nowhere", in A. V. Cherkaev and R. V. Kohn (Eds.), *Topics in the mathematical modeling of composite materials*. Cambridge, MA: Birkhäuser. 31:45–93.

[59] D. Cioranescu and J. Saint-Jean-Paulin, 1999. *Homogenization of Reticulated Structures*. New York: Springer Verlag.

[60] A. Clarke and C. Eberhardt, 2002. *Microscopy Techniques for Materials Science*. Cambridge: Woodhead Publishing.

[61] O. Coindreau, 2003. *Etude 3D des preformes fibreuses: Interaction entre phenomenes physico-chimiques et geometrie*. PhD Thesis, Université of Bordeaux 1 (France).

[62] O. Coindreau, P. Cloetens, G. L. Vignoles, 2003. "Direct 3D microscale imaging of C/C composites with computed holotomography", *Nuclear Instruments and Methods in Physics Research B*, 200:295–302.

[63] O. Coindreau and G.L. Vignoles, 2005. "Assessment of structural and transport properties in fibrous C/C composite preforms as digitized by X-ray CMT. Part I: Image acquisition and geometrical properties", *Journal of Materials Research*, 20:2328–2339.

[64] R. Cole and R. Hariharan, 1998. "Approximate string matching: A simpler faster algorithm", *Proceedings of the 9^{th} ACM-SIAM Symposium on Discrete Algorithms (SODA '98)*, 463–472.

[65] J. M. Constantin, M. W. Berry, B. T. Van der Zanden, 1997. "Parallelization of the Hoshen-Kopelman algorithm using a finite state machine", *International Journal of Supercomputing Applications and High Performance Computing*, 11:34.

[66] T. M. Cover and P. E. Hart, 1967. "Nearest neighbor pattern classification", *IEEE Transactions on Information Theory*, 13:21–27.

[67] R. Courant, K. Friedrichs, H. Lewy, 1928. "Uber die partiellen Differenzengleichungen der mathematischen Physik", *Mathematischen Annalen*, 100:32–74.

[68] A. Dempster, N. Laird, D. Rubin, 1977. "Maximum likelihood from incomplete data via the EM algorithm", *Journal of the Royal Statistical Society B*, 39(1):1–38.

[69] R. E. Díaz, W. M. Merrill, N. G. Alexopoulos, 1998. "Analytic framework for the modeling of effective media", *Journal of Applied Physics*, 84(12):6815–6826.

[70] M. B. Dillencourt, H. Samet, M. Tamminen, 1992. "A general approach to connected-component labeling for arbitrary image representations", *Journal of the ACM* 39:253–280.

[71] R. O. Duda and P. E. Hart, 1973. *Pattern Classification and Scene Analysis*. New York: Wiley.

[72] M. Engeli, 1959. "Over-relaxation and Related Methods", in M. Engeli, Th. Ginsburg, H. Rutishauser, E. Stiefel (Eds.) *Refined Iterative Methods for Computation of the Solution and the Eigenvalues of Self-Adjoint Boundary Value Problems*, Mitteilungen aus dem Institut für angewandte Mathematik an der Eidgenössischen Technischen Hochschule Zürich, 8. Basel: Birkhäuser Verlag.

[73] A. F. Emery and W. W. Carson, 1968. "A modification to the Monte Carlo method —The Exodus method", *Journal of Heat Transfer, Transactions ASME, Series C*, 90:328–332.

[74] B. S. Everitt, 1974. *Cluster Analysis*. London: Heinemann.

[75] B. U. Felderhof, 1984. "Bounds for the complex dielectric constant of a two-phase composite", *Physica A*, 126:430–442.

[76] E. Fix and J. L. Hodges, 1951. "Discriminatory analysis-non parametric discrimination; consistency properties". *Technical Report Project 21-49-004*, 4, USAF School of Aviation Medicine.

[77] M. Flanigan and P. Tamayo, 1995. "Parallel cluster labelling for large-scale Monte Carlo simulations", *Physica A*, 215:461.

[78] G. Floquet, 1883. "Sur les équations différentielles linéaries à coefficients périodiques", *Annales de l'École Normale Supérieure*, 12:47–88.

[79] A. S. Frangakis and R. Hegerl, 1999. "Nonlinear anisotropic diffusion in three-dimensional electron microscopy", *Scale-Space Theories in Computer Vision, Lecture Notes in Computer Science*, 1682:386–397.

[80] S. Frankel, 1977. *Multiconductor Transmission Line Analysis*. Norwood, MA: Artech House. 359–366.

[81] H. Freeman, 1961. "On the encoding of arbitrary geometric configurations", *IRE Transactions on Electronic Computers EC*, 10:260–268.

[82] H. Freeman, 1974. "Computer processing of line drawing images", *ACM Computing Surveys*, 6:57–97.

[83] K. S. Fu, 1982. *Syntactic Pattern Recognition and Applications*. Englewood Cliffs, NJ: Prentice-Hall.

[84] Y. Fu, K. J. Klimkowski, G. J. Rodin, E. Berger, J. C. Browne, J. K. Singer, R. A. van de Geijn, K. S. Vemaganti, 1998. "A Fast Solution Method for Three-Dimensional Many-Particle Problems of Linear Elasticity", *International Journal for Numerical Methods in Engineering*, 42:1215–1229.

[85] K. Fukunaga, 1990. *Introduction to Statistical Pattern Recognition*. 2nd Edition. San Diego: Academic Press.

[86] Z. Galil and K. Park, 1990. "An improved algorithm for approximate string matching", *SIAM Journal of Computing*, 19(6):989–999.

[87] J. C. M. Garnett, 1904. "Colours in metal glasses and in metallic films", *Philosophical Transactions of the Royal Society of London*, 203:385–420.

[88] S. Geman and D. Geman, 1984. "Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 6:721–741.

[89] P. K. Ghost and M. E. Azimi, 1994. "Numerical calculation of effective permittivity of lossless dielectric mixtures using boundary integral method", *IEEE Transactions on Dielectrics and Electrical Insulators*, 1(6):975–982,

[90] A. B. Galler and M. J. Fischer, 1964. "An improved equivalence algorithm", *Communications of the ACM*, 7:301.

[91] H. Gould and S. Tobochnick, 1996. *An Introduction to Computer Simulation Methods: Applications to Physical Systems* (2nd Ed.), vol. 2. New York: Addison-Wesley.

[92] G. Grant and A. F. Reid, 1981. "An efficient algorithm for boundary tracking and feature extraction", *Computer Graphics and Image Processing*, 17:225–237.

[93] L. Greengard and V. Rokhlin, 1997. "A new version of the fast multipole method for the Laplace equation in three dimensions", *Acta Numerica*, 1997:229–269.

[94] L. Greengard and J. Helsing, 1998. "On the numerical evaluation of elastostatic fields in locally isotropic two-dimensional composites", *Journal of the Mechanics and Physics of Solids*, 46(8):1441–1462.

[95] G. Grimmet, 1987. "Interacting particle systems and random media: An overview", *International Statistical Review*, 55:49.

[96] R. K. Guy, 1994. "B_2-Sequences", in *Unsolved Problems in Number Theory*, 2nd ed. New York: Springer-Verlag. 228–229.

[97] S. Gupte and J. Tsamopoulos, 1990. "An effective medium approach for modeling chemical vapor infiltration of porous ceramic materials", *Journal of the Electrochemical Society*, 137(5):1626–1638.

[98] J. M. Hammersley and D. C. Handscomb, 1964. *Monte Carlo Methods*. London: Methuen & Co.

[99] R. M. Haralick, 1981. "Some neighborhood operations", in M. Onoe, K. Preston, A. Rosenfeld (Eds.) *Real Time/Parallel Computing Image Analysis*. New York: Plenum Press.

[100] R. M. Haralick, R. Klette, H. S. Stiehl, M. Viergever, 1999. *Evaluation and Validation of Computer Vision Algorithms*. Boston: Kluwer.

[101] J. A. Hartigan and M. A. Wong, 1979. "Algorithm AS136: A K-means clustering algorithm", *Applied Statistics*, 28:100–108

[102] H. Hartley, 1958. "Maximum likelihood estimation from incomplete data", *Biometrics* 14:174–194.

[103] Z. Hashin and S. Shtrikman, 1962. "A variational approach to the theory of the effective magnetic permeability of multiphase materials", *Journal of Applied Physics*, 33(10):3125–3131.

[104] S. Haykin, 1999. *Neural Networks. A Comprehensive Foundation*. 2nd Edition. New Jersey: Prentice Hall.

[105] J. Helsing and A. Helte, 1991. "Effective conductivity of aggregates of anisotropic grains", *Journal of Applied Physics*, 69:3583.

[106] J. Helsing, 1993. "Bounds to the conductivity of some two-component composites", *Journal of Applied Physics*, 73(3):1240–1245.

[107] M. A. Hilhorst, C. Dirksen, F. W. H. Kampers, R. A. Feddes, 2000. "New dielectric mixture equation for porous materials based on depolarization factors", *Soil Sciences Society of America Journal*, 64:1581–1587.

[108] J. Hopfield, 1982. "Neural networks and physical systems with emergent collective computational abilities", *Proceedings of the National Academy of Sciences of the USA*, 79:2554.

[109] U. Hornung, 1997. *Homogenization and Porous Media*. New York: Springer-Verlag.

[110] J. Hoshen and R. Kopelman, 1976. "Percolation and cluster distribution: Cluster multiple labeling technique and critical concentration algorithm", *Physical Review B*, 14(1):3438–3445.

[111] J. Hoshen, M. W. Berry, K. S. Minser, 1997. "Percolation and cluster structure parameters: The enhanced Hoshen–Kopelman algorithm", *Physical Review E* 56:1455.

[112] J. Hoshen, 1998. "On the application of the enhanced Hoshen–Kopelman algorithm for image analysis", *Pattern Recognition Letters* 12:575–584.

[113] T. P. Iglesias, J. F. Peón-Fernández, J. Martín-Herrero, A. Seoane, 2002. "A mixing rule for permittivity from an approximation of a model in a framework of a mesoscopic scale", *Journal of Molecular Liquids*, 95:147–155.

[114] T. P. Iglesias, N. Banerji, J. F. Peón-Fernández, J. Martín-Herrero, 2004. "Analytical-numerical estimation of effective permittivity in multicomponent systems", *Proceedings of the 2004 URSI EMT-S International Symposium on Electromagnetic Theory*, 2:657–659.

[115] E. Ising, 1925. "Beitrag zur Theorie des Ferromagnetismus", *Zeitschrift für Physik*, 31:253–258.

[116] H. Jeffreys and B. S. Jeffreys, 1988. "Relaxation Methods", §9.18 in *Methods of Mathematical Physics*, 3rd. ed. Cambridge: Cambridge University Press. 307–312.

[117] V. V. Jikov, S. M. Kozlov, O. A. Oleĭnik, 1994. *Homogenization of differential operators and integral functionals*. Berlin: Springer-Verlag.

[118] S. Jin, X. Wang, T. L. Starr, X. F. Chen, 2000. "Robust numerical simulation of porosity evolution in chemical vapor infiltration I: Two space Dimension", *Journal of Computational Physics*, 162(2):467–482.

[119] S. Jin and X Wang, 2002. "Robust numerical simulation of porosity evolution in chemical vapor infiltration II: Two dimensional anisotropic fronts", *Journal of Computational Physics*, 179(2):557–577.

[120] S. Jin and X Wang, 2003. "Robust numerical simulation of porosity evolution in chemical vapor infiltration III: Three space dimension", *Journal of Computational Physics*, 186(2):582–595.

[121] W. Kahan, 1958. *Gauss–Seidel Methods of Solving Large Systems of Linear Equations*. PhD Thesis. University of Toronto.

[122] K. K. Kärkkäinen, A. H. Sihvola, K. I. Nikoskinen, 2000. "Effective permittivity of mixtures: Numerical validation by the FDTD method", *IEEE Transactions on Geosciences and Remote Sensing*, 38(5):1303–1308.

[123] K. K. Kärkkäinen, A. H. Sihvola, K. I. Nikoskinen, 2001. "Analysis of a three-dimensional dielectric mixture with finite difference method", *IEEE Transactions on Geosciences and Remote Sensing*, 39:1013–1018.

[124] M. J. Kearney, 2002. "Compact directed percolation with modified boundary rules: A forest fire model", *Journal of Physics A: Mathematical and General*, 35:L421–L425.

[125] E. Ya. Khruslov, 1991. "Homogenized models of composite media", in *Composite media and homogenization theory*. Cambridge, MA: Birkhäuser Boston Inc. 159–182.

[126] M. Kichenassamy, 1997. "The Perona-Malik paradox", *SIAM Journal of Applied Mathematics*, 57:1328–1342.

[127] S. D. Kim, J. H. Lee, J. K. Kim, 1988. "A new chain-coding algorithm for binary images using run-length codes", *Computer Vision, Graphics, and Image Processing*, 41:114–128.

[128] Y. Kitazume, E. Ohira, T. Endo, 1985. "LSI implementation of a pattern matching algorithm for speech recognition", *IEEE Transactions on Acoustics, Speech and Signal Processing*, 33(1):1–5.

[129] W. E. Kohler and G. C. Papanicolaou, 1981. "Some applications of the coherent potential approximation", in P. L. Chow, W. E. Kohler, G. C. Papanicolaou (Eds.) *Multiple Scattering and Waves*. New York: North Holland. 199–223.

[130] T. Kohonen, 1982. "Self-organized Formation of Topologically Correct Feature Maps", *Biological Cybernetics*, 43:59–69.

[131] T. Kohonen, 1989. *Self-Organization and Associative Memory*. Berlin: Springer-Verlag.

[132] T. Kohonen, 1990. "The self-organizing map", *Proceedings of the IEEE*, 78(9):1464–1480.

[133] T. Kohonen, 1991. "Self-organizing maps: Optimization approaches", in T. Kohonen, K. Mäkisara, O. Simula, J. Kangas (Eds.) *Artificial Neural Networks* 2:981–990.

[134] T. Kohonen, 1997. *Self-Organizing Maps*. Berlin: Springer-Verlag.

[135] J. Koplowitz and A. M. Bruckstein, 1989. "Design of perimeter estimators for digitized planar shapes", *IEEE transactions on Pattern Analysis and Machine Intelligence*, 11:611–622.

[136] S. M. Kozlov and A. L. Piatnitski, 1996. "Degeneration of effective diffusion in the presence of periodic potential", *Annales de l'Institut Henri Poincaré. Section B, Probabilités et Statistiques*, 32(5):571–587.

[137] S. Kullback and R. A. Leibler, 1951. "On information and sufficiency", *Annals of Mathematics and Statistics*, 22:79–86.

[138] Z. Kulpa, 1977. "Area and perimeter measurement of blobs in discrete binary pictures", *Computer Graphics Image Processing*, 6:434–451.

[139] P. A. Lachenbruch and R. M. Mickey, 1968. "Estimation of error rates in discriminant analysis", *Technometrics*, 10:1–11.

[140] G. Landau and U. Vishkin, 1988. "Fast string matching with k differences", *Journal of Computer Systems Science*, 37:63–78.

[141] L. Landau, E. Lifshitz, and L. P. Pitaevski, 1984. *Electrodynamics of Continuous Media*. Oxford: Pergamon.

[142] P. L. Leath, 1976. "Cluster size and boundary distribution near percolation threshold", *Physical Review B*, 14:5046–5055.

[143] V. I. Levenshtein, 1965. "Binary codes capable of correcting deletions, insertions and reversals", *Doklady Akademii Nauk SSSR*, 163(4):845–848 (in Russian). English translation in *Soviet Physics Doklady* (1966) 10(8):707-710.

[144] V. M. Levin, 1999. "The elastic wave propagation in the piezoelectric fiber reinforced composites", in K. Markov and E. Inan (eds.) *Continuum models and discrete systems*. River Edge, NJ: World Scientific Publishing.

[145] M. D. Levine, 1985. *Vision in Man and Machine*. New York: McGraw-Hill.

[146] H. Looyenga, 1965. "Dielectric constants of heterogeneous mixtures", *Physica*, 31:401–406.

[147] K. Lichtenecker and K. Rother, 1931. "Die Herleitung des logarithmischen Mischungsgesetzes aus allgemeinen Prinzipien der stationären Strömung", *Physikalische Zeitschrift*, 32:255–260.

[148] J. Liu, 2001. *Monte Carlo Strategies in Scientific Computing*. New York: Springer Verlag.

[149] G.F. Luger and W.A. Stubblefield, 1993, *Artificial Intelligence: Structures and Strategies for Complex Problem Solving*. 2nd Edition. New York: Chapman Hall.

[150] R. Lumia, L. Shapiro, O. Zuniga, 1983. "A new connected components algorithm for virtual memory computers", *Computer Vision, Graphics and Image Processing*, 22:287–300.

[151] T. Mäenpää, T. Ojala, M. Pietikäinen, M. Soriano, 2000. "Robust texture classification by subsets of Local Binary Patterns", *Proceedings of the 15th International Conference on Pattern Recognition (ICPR'00)*, 3:3947.

[152] K. Z. Markov, D. R. S. Talbot, J. R. Willis, 1996. "On stationary diffusion in heterogeneous media", *IMA Journal of Applied Mathematics*, 56(2):133–144.

[153] K. Z. Markov and L. Preziosi, 1999. *Heterogeneous Media: Micromechanics Modelling Methods and Simulation*. Basel: Birkhäuser Verlag.

[154] P. Marquardt, 1992. "Size-governed dielectric properties of matrix-isolated and percolating mesoscopic conductors", *Journal of Electromagnetic Waves and Applications*, 6(9):1197–1223.

[155] J. Martín-Herrero and J. F. Peón-Fernández, 2000. "Alternative techniques for cluster labeling on percolation theory", *Journal of Physics A: Mathematical and General*, 33:1827–1840.

[156] J. Martín-Herrero, J. F. Peón-Fernández, N. Banerji, T. P. Iglesias, 2002. "Monte Carlo estimation of the effective electric response in composites", in S.Bandyopadhyay (Ed.) *Composite Systems: Macrocomposites, Microcomposites, Nanocomposites*. 300–306.

[157] J. Martín-Herrero and J. L. Alba-Castro, 2003. "High speed machine vision: The canned tuna case", in J. Billingsley (Ed.) *Mechatronics and Machine Vision in Practice: Future Trends*. Hertfordshire: Research Studies Press.

[158] J. Martín-Herrero, M. Ferreiro-Armán, J. L. Alba-Castro, 2004. "A SOFM improves a real time quality assurance machine vision system", *IEEE Proceedings of the 17th International Conference on Pattern Recognition ICPR'04*, 4:301–304.

[159] J. Martín-Herrero, 2004. "Hybrid cluster identification", *Journal of Physics A: Mathematical and General*, 37:9377–9386.

[160] J. Martín-Herrero, 2005. "Two low level image processing techniques for the characterization of composite systems", *Proceedings of the 5th International Conference on Composite Science and Technology*.

[161] J. Martín-Herrero and J. F. Peón-Fernández, 2005. "Computation of longwave electromagnetic response of nonhomogeneous media", *IEEE Transactions on Geoscience and Remote Sensing*, 43(7):1479-1489.

[162] J. Martín-Herrero, J. F. Peón-Fernández, T. P. Iglesias, 2005. "The effective permittivity of lossless granular composites: From periodic to random distributions", *Proceedings of the XXVIII General Assembly of the International Union of Radio Science (URSI-GA2005)*.

[163] J. Martín-Herrero and C. Germain, 2006. "Extraction of fibres in tomographic samples of composites by computer vision", invited contribution to *International Composites Conference ACUN5*, (in press).

[164] J. Martín-Herrero, 2006. "Hybrid object labelling in digital images", *Machine Vision and Applications Journal*, (in press).

[165] A. Martinez and A. P. Byrnes, 2001. "Modeling dielectric-constant values of geologic materials: An aid to ground penetrating radar data collection and interpretation", *Current Research in Earth Sciences*, Kansas Geological Survey, 247(1):1–15.

[166] A. Marzal and E. Vidal, 1993. "Computation of normalized edit distance and applications", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(9):926–932.

[167] J. W. McKee and J. K. Aggarwal, 1977. "Computer recognition of partial views of curved objects", *IEEE Transactions on Computing*, C-26:790–800.

[168] R. C. McPhedran and N. A. Nicorovici, 1997. "Effective dielectric constant of arrays of elliptical cylinders", *Physica A*, 241:173–178.

[169] N. Metropolis, A. Rosenbluth, M. Rosenbluth, A. Teller, E. Teller, 1953. "Equations of state calculations by fast computing machines", *Journal of Chemical Physics*, 21:1087–1091.

[170] G. W. Milton, 1981. "Bounds on the complex permittivity of a two component composite material", *Journal of Applied Physics*, 52(8):5286–5293.

[171] G. W. Milton, 1992. "Composite materials with Poisson's ratios close to -1", *Journal of the Mechanics and Physics of Solids*, 40(5):1105–1137.

[172] T. Minami and K. Shinohara, 1986. "Encoding of line drawings with a multiple grid chain code", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 8(2):269–276.

[173] Ministerium für Wirtschaft des Landes Nordrhein-Westfalen, 1996. *Produkte und Dienstleistungen für die Bildverarbeitung: Stand und Trends*. Düsseldorf.

[174] M. Minsky and S. Papert, 1969. *Perceptrons*. Cambridge: MIT Press.

[175] MPEG Video Group, 1999. *MPEG-4 Requirements, Version 11*, Doc. ISO/IEC JTC1/SC29/WG11 N2723.

[176] A. Moilanen and I. Hanski, 2001. "On the use of connectivity measures in spatial ecology", *Oikos*, 95:147–151.

[177] N. R. Moloney and G. Pruessner, 2003. "Asynchronously parallelized percolation on distributed machines", *Physical Review E*, 67:37701.

[178] J. Monecke, 1994. "Bergman spectral representation of a simple expression for the dielectric response of a symmetric two component composite", *Journal of Physics Condensed Matter*, 6:907–912.

[179] P. M. Morse and H. Feshbach, 1953. "Gauss's Theorem.", in *Methods of Theoretical Physics, Part I*. New York: McGraw-Hill. 37–38.

[180] C. Moukarzel, 1998. "A fast algorithm for backbones", *International Journal of Modern Physics C*, 9:887.

[181] G. Myers, 1986. "An $O(ND)$ difference algorithm and its variations", *Algorithmica* 1:251–266.

[182] M. E. Muller, 1956. "Some continuous Monte Carlo methods for the Dirichlet problem", *Annals of Mathematics and Statistics*, 27(3):569–589.

[183] G. Navarro, 2001. "A guided tour to approximate string matching", *ACM Computing Surveys*, 33(1):31–88.

[184] S. Needleman and C. Wunsch, 1970. "A general method applicable to the search for similarities in the amino acid sequences of two proteins", *Journal of Molecular Biology*, 48:444–453.

[185] P. S. Neelakanta, 1995. *Handbook of Electromagnetic Materials*. Boca Raton, FL: CRC Press.

[186] S. Nemat-Nasser and M. Hori, 1993. *Micromechanics: Overall properties of heterogeneous materials*. Amsterdam: North-Holland Publishing.

[187] M. E. J. Newman and R. M. Ziff, 2001. "A fast Monte Carlo algorithm for site or bond percolation", *Physical Review E*, 64:16706.

[188] M. Nitzberg and T. Shiota, 1992. "Nonlinear image filtering with edge and corner enhancement", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14:826–833.

[189] T. Ojala, M. Pietikäinen, D. Harwood, 1996. "A comparative study of texture measures with classification based on feature distributions", *Pattern Recognition*, 29(1):51–59.

[190] T. Ojala and M. Pietikäinen, 1999. "Unsupervised texture segmentation using feature distributions", *Pattern Recognition*, 32(1):477–486.

[191] T. Ojala and M. Pietikäinen, T. Mäenpää, 2002, "Multiresolution gray-scale and rotation invariant texture classification with local binary patterns", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24:971–987

[192] M. Ostoja-Starzewski, 2002. "Lattice models in micromechanics", *Applied Mechanics Review*, 55(1):35–60.

[193] G. P. Panasenko, 1994. "Homogenization of lattice-like domains: L-convergence". *Report CNRS URA 740, 178.* Lyon/Saint-Etienne: Equipe d'Analyse Numérique.

[194] G. Paul, R. M. Ziff, H. E. Stanley, 2001. "Percolation threshold, Fisher exponent, and shortest path exponent for 4 and 5 dimensions", *Physical Review E*, 64:26115.

[195] O. Pekonen, K. K. Kärkkäinen, A. H. Sihvola, K. I. Nikoskinen, 1999. "Numerical testing of dielectric mixing rules by FDTD method", *Journal of Electromagnetic Waves and Applications*, 13:67–87.

[196] J. F. Peón-Fernández, J. Martín-Herrero, N. Banerji, T. P. Iglesias, 2004. "Numerical study of effective permittivity in composite systems", *Applied Surface Science*, 226:78–82.

[197] P. Perona and J. Malik, 1990. "Scale-space and edge detection using anisotropic diffusion", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12:629–639.

[198] M. Pietikäinen and T. Ojala, 1999. "Nonparametric texture analysis with simple spatial operators", *Proceedings of the* 5^{th} *International Conference on Quality Control by Artificial Vision*, 11–16.

[199] D. Polder and J. H. van Santen, 1946. "The effective permeability of mixtures of solids", *Physica*, 12(5):257–271.

[200] R. B. Potts, 1952. "Some generalized order-disorder transformations", *Proceedings of the Cambridge Philosophical Society*, 48:106 .

[201] D. Profitt and D. Rosen, 1979. "Metrication errors and coding efficiency of chain coding schemes for the representation of lines and edges", *Computer Graphics Image Processing*, 10:318–332.

[202] D. C. Rappaport, 1986. "Cluster number scaling in two-dimensional percolation", *Journal of Physics A: Mathematical and General*, 19:291.

[203] A. R. Rao and B. G. Schunck, 1991. "Computing Oriented Texture Fields", *CVGIP: Graphical Models and Image Processing*, 53(2):157–185.

[204] N. Reuge, G.L. Vignoles, H. Le Poche, F. Langlais, 2003. "Modeling of pyrocarbon chemical vapor infiltration", in P. Vincenzini, A. Lami (Eds.), *Computational Modelling and Simulation of Materials II: Advances in Science and Technology*. Faenza: Techna Group. 36:259–266.

[205] N. Reuge and G.L. Vignoles, 2005. "Modeling of isobaric-isothermal chemical vapor infiltration: Effects of reactor control parameters on a densification", *Journal of Materials Processing Technology*, 166:15-29.

[206] P.A. Rikvold and G. Stell, 1985. "Porosity and specific surface for interpenetrable-sphere models of two-phase random media", *Journal of Chemical Physics*, 31:1014-1020.

[207] F. Rosenblatt, 1962. *Principles of Neurodynamics*. New York: Spartan.

[208] A. Rosenfeld and J. Pfaltz, 1966. "Sequential operations in digital picture processing", *Journal of the ACM*, 13:471–494.

[209] A. Rosenfeld and A. C. Kak, 1976. *Digital Picture Processing*. San Diego: Academic Press.

[210] D. E. Rummelhart and J. L. McLelland, 1986. *Parallel Distributed Processing*. Cambridge: MIT Press.

[211] L. Ryzhik, G. Papanicolaou, J. Keller, 1996. "Transport equations for elastic and other waves in random media", *Wave Motion*, 24(4):327–370.

[212] E. Sadeh and M. A. Franklin, 1974. "Monte Carlo solutions of partial differential equations by special purpose digital computers", *IEEE Transactions on Computing*, C-23:389–397.

[213] M. N. O. Sadiku and D. T. Hunt, 1992. "Solution of Dirichlet problems by the Exodus method", *IEEE Transactions on Microwave Theory and Technology*, 40(1):89–95.

[214] H. Samet, 1981. "Connected component labeling using quadtrees", *Journal of the ACM*, 28:487–501.

[215] H. Samet and M. Tamminen, 1988. "Efficient component labelling of images of arbitrary dimension represented by linear bintrees", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 10(4):579–586.

[216] E. Sánchez-Palencia, 1980. *Nonhomogeneous media and vibration theory*. Berlin: Springer-Verlag.

[217] D. Sankoff, 1972. "Matching sequences under deletion/insertion constraints", *Proceedings of the National Academy of Sciences of the USA*, 69:4–6.

[218] D. Sankoff and J. B. Kruskal, 1983. *Time Warps, String Edits, and Macromolecules: The Theory and Practice of Sequence Comparison*. Reading: Addison-Wesley.

[219] B. Sareni, 1997. "Étude de la permittivité effective des matériaux composites", *Journal of Physique III France*, 7(4):793–801.

[220] B. Sareni, L. Krähenbühl, A. Beroual, A. Nicolas, C. Brosseau, 1997. "A boundary integral equation method for the calculation of the effective permittivity of periodic composites", *IEEE Transactions on Magnetism*, 33:1580–1583.

[221] P. Sellers, 1974. "On the theory and computation of evolutionary distances", *SIAM Journal on Applied Mathematics*, 26:787–793.

[222] L. G. Shapiro and G. C. Stockman, 2001. *Computer Vision*. New Jersey: Prentice Hall.

[223] B. W. Sheldon and T. M. Besmann, 1991. "Reaction and diffusion kinetics during the initial stage of isothermal chemical vapor infiltration", *Journal of the American Ceramics Society*, 74(12):3046–3052.

[224] A. P. Sheppard, M. A. Knackstedt, W. .V. Pinczewski, M. Sahimi, 1999. "Invasion percolation: New algorithms and universality classes", *Journal of Physics A: Mathematical and General*, 32:L521.

[225] O. Sigmund and S. Torquato, 1997. "Design of materials with extreme thermal expansion using a three-phase topology optimization method", *Journal of the Mechanics and Physics of Solids*, 45(6):1037–1067.

[226] A. H. Sihvola, S. Saastamoinen, K. Heiska, 1994. "Mixing rules and percolation", *Remote Sensing Review*, 9:39–50.

[227] A. H. Sihvola, 1999. *Electromagnetic Mixing Formulas and Applications*. London: Institute Electrical Engineers.

[228] A. H. Sihvola, 2002. "How strict are theoretical bounds for dielectric properties of mixtures", *IEEE Transactions on Geosciences and Remote Sensing*, 40(4):880–886.

[229] D. J. Skamser, D. P. Bentz, R. T. Covrdale, 1994. "Calculation of the thermal conductivity and gas permeability in a uniaxial bundle of fibers", *Journal of the American Ceramics Society*, 77(10):2669–2680.

[230] S. Sotirchos and S. Zarkanitis, 1993. "A distributed pore size and length model for porous media reacting with diminishing porosity", *Chemical Engineering Science*, 48:1487:1502.

[231] A. Spanoudaki and R. Pelster, 2001. "Effective dielectric properties of composite materials: the dependence of the particle size distribution", *Physical Review B*, 64:64205-1–64205-6.

[232] T. L. Starr, 1995. "Gas transport model for chemical vapor infiltration", *Journal of Materials Research*, 10(9):2360–2366.

[233] D. Stauffer and A. Aharony, 1994. *Introduction to Percolation Theory*, 2nd ed. London: Taylor & Francis.

[234] R. Stognienko, T. Henning, V. Ossenkopf, 1995. "Optical properties of coagulated particles", *Astronomy and Astrophysics*, 296:797–809.

[235] A. Stöhr, 1955. "Gelöste und ungelöste Fragen über Basen der natürlichen Zahlenreihe. I, II", *J. reine Angew. Math.*, 194:40–65, 111–140.

[236] B. Stroustrup, 1997. "The C++ Programming Language". New York: Addison-Wesley.

[237] K. Stüber and U. Trottenberg, 1982. "Multigrid methods: fundamental algorithms, model problem analysis and applications", in Hackbuich and Trottenberg (Eds.) *Lecture Notes in Mathematics*, 960. Berlin: Springer.

[238] R. Swendsen and J. Wang, 1987. "Nonuniversal critical dynamics in Monte Carlo simulations", *Physical Review Letters*, 58:86.

[239] S. L. Tanimoto, 1990. *The elements of Artificial Intelligence Using Common LISP*. New York: W. H. Freeman.

[240] R. E. Tarjan, 1972. "Depth-first search and linear graph algorithms", *SIAM Journal of Computing*, 1:146.

[241] R. E. Tarjan, 1975. "Efficiency of a good but not linear set union algorithm", *Journal of the ACM*, 22:215.

[242] J. J. Telega, 1990. "Some results of homogenization in plasticity: plates and fissured solids", in M. Kleiber and J. A. Konig (Ed.), *Inelastic Solids and Structures*. Swansea: Pineridge Press. 243–260.

[243] J. M. Teuler and J. C. Gimel, 2000. "A direct parallel implementation of the Hoshen-Kopelman algorithm for distributed memory architectures", *Computer Physics Communications*, 130:118.

[244] D. Tiggemann, 2001. "Simulation of percolation on massively-parallel computers", *International Journal of Modern Physics C*, 12:871.

[245] M. Tomadakis and S. Sotirchos, 1991. "Effective Knudsen diffusivities in structures of randomly overlapping fibres", *AIChE Journal*, 37(1):74–86.

[246] F. Torkamani-Azar and K. E. Tait, 1996. "Image recovery using the anisotropic diffusion equation", *IEEE Transactions on Image Processing*, 5:1573–1578.

[247] S. Torquato, 1999. "Diffusion-controlled reaction and flows in porous media", in K. Markov and L. Preziosi (Eds.), *Heterogeneous Media: Micromechanics Modelling Methods and Simulation*. Basel: Birkhäuser Verlag.

[248] E. Tuncer and S. Gubanski, 2001. "Dielectric relaxation in dielectric mixtures: Application of the finite element method and its comparison with dielectric mixture formulas", *Journal of Applied Physics*, 89(12):8092–8100.

[249] E. Tuncer, Y. V. Serdyuk, S. M. Gubanski, 2002. "Dielectric mixtures: Electrical properties and modeling", *IEEE Transactions on Dielectrics and Electrical Insulators*, 9(5):809–828.

[250] L. Tsang and J. A. Kong, 1981. "Application of strong fluctuation random medium theory to scattering from a vegetation-like half space", *IEEE Transactions on Geosciences and Remote Sensing*, 19(1):62–69.

[251] E. Ukkonen and D. Wood, 1993. "Approximate string matching with suffix automata", *Algorithmica*, 10:353–364.

[252] F. T. Ulaby, R. K. Moore, A. K. Fung, 1986. *Microwave Remote Sensing*. Massachusetts: Artech House.

[253] T. Ung, L. M. Liz-Marzán, P. Maulvaney, 2001. "Optical properties of thin films of Au at SiO_2 particles", *Journal of Physical Chemistry B*, 105:3441–3452.

[254] S. Vaidyaraman, W. J. Lackey, G. G. Freman, P. K. Agrawal, M. D. Langman, 1995. "Fabrication of Carbon-Carbon composites by forced thermal gradient chemical infiltration", *Journal of Materials Resesearch*, 10:1469–1477.

[255] G. L. Vignoles, 2001. "Image Segmentation for hard X-ray phase contrast images of C/C composites", *Carbon*, 39:167-173.

[256] T. Vintsyuk, 1968. "Speech discrimination by dynamic programming", *Cybernetics*, 4:52–58.

[257] R. Wagner and M. Fisher, 1974. "The string to string correction problem", *Journal of the ACM*, 21:168–178.

[258] T. Wagner and P. Plankensteiner, 1999. "Industrial Object Recognition", in B. Jähne (Ed.) *Handbook of Computer Vision and Applications*, 3. San Diego: Academic Press. 300.

[259] J. Wang, 2002. http://www.cz3.nus.edu.sg/∼wangjs/Bworkshop /sw-oner-gg.c

[260] X. Wang, 2002. *Level Set Model of Microstructure Evolution in the Chemical Vapor Infiltration Process*. PhD Theses. Georgia Institute of Technology.

[261] G. W. Wei, 1999. "Generalized Perona-Malik equation for image restoration", *IEEE Signal Processing Letters*, 6(7):165–167.

[262] J. Weickert, 1996. "Theoretical foundations of anisotropic diffusion in image processing", in W. Kropatsch, R. Klette, F. Solina (Eds.) *Theoretical Foundations of Computer Vision*, Computing Supplement 11. Wien: Springer-Verlag. 221–236.

[263] J. Weickert and B. Benhamouda, 1997. "A semidiscrete nonlinear scale-space theory and its relation to the Perona-Malik paradox", in F. Solina, W. Kropatsch, R. Klette, R. Bajcsy (Eds.) *Advances in Computer Vision*. Wien: Springer-Verlag. 1–10.

[264] J. Weickert, 1998. *Anisotropic Diffusion in Image Processing.* Stuttgart: Teubner.

[265] J. Weickert, 1999. "Coherence-enhancing diffusion filtering", *International Journal of Computer Vision*, 31(2/3):111–127.

[266] H. Ch. Weissker, J. Furthmüller, F. Bechstedt, 2003. "Validity of effective-medium theory for optical properties of embedded nanocrystallites from *ab initio* supercell calculations", *Physical Review B*, 67:165322-1–165322-5.

[267] K. W. Whites and F. Wu, 2002. "Effects of particle shape on the effective permittivity of composite materials with measurements for lattices of cubes", *IEEE Transactions on Microwave Theory and Technology*, 50(7):1723–1729.

[268] O. Wiener, 1910. "Zur Theorie der Refraktionskonstanten", *Berichte über die Verhandlungen der Königlich-Sächsischen Gesellschaft der Wisseschaften zu Leipzig*, vol. Math. phys. Klasse, 62:256–277.

[269] K. A. With, 1997. "The application of neutral landscape models in conservation biology", *Conservation Biology*, 11:1069–1080.

[270] K. A. With, 2002. "Using percolation theory to assess landscape connectivity and effects of habitat fragmentation", in K. J. Gutzwiller (Ed.) *Applying Landscape Ecology in Biological Conservation.* New York: Springer-Verlag.

[271] U. Wolff, 1989. "Collective Monte Carlo updating for spin systems", *Physical Review Letters*, 62:361.

[272] F. Y. Wu, 1982. "The Potts model", *Review of Modern Physics*, 54:235.

[273] S. Wu and U. Manber, 1992. "Fast text searching allowing errors". *Communications of the ACM*, 35(10):83–91.

[274] Y. L. You, W. Xu, A. Tannenbaum, M. Kaveh, 1996. "Behavioral analysis of anisotropic diffusion in image processing", *IEEE Transactions on Image Processing*, 5:1539–1553.

[275] A. Youssef, 1993. "A parallel algorithm for random walk construction with application to the Monte Carlo solution of partial differential equations", *IEEE Transactions on Parallel Distributed Systems*, 4(3):355–360.

[276] H. C. Yu and S. V. Sotirchos, 1987. "A generalized pore mode for gas-solid reactions exhibiting pore closure", *AIChE Journal*, 33:382–393.

[277] R. M. Ziff, P. T. Cummings, G. Stell, 1992. "Generation of percolation cluster perimeters by a random walk", *Journal of Physics A: Mathematical and General*, 17:3009.

[278] P. Zingaretti, M. Gasparroni, L. Vecci, 1998. "Fast chain coding of region boundaries", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(4):407–415.

[279] T. I. Zohdi, J. T. Oden, G. J. Rodin, 1996. "Hierarchical modeling of heterogeneous bodies", *Computer Methods in Applied Mechanics and Engineering*, 138(1-4):273–298.

'I have, doubtless, excited your curiosity,
as well as that of these good people;
but you are too considerate to make inquiries.'

'Certainly; it would indeed be very impertinent and inhuman of me
to trouble you with any inquisitiveness of mine.'