SEGMENTATION OF TEXTURED IMAGES ON THREE-DIMENSIONAL LATTICES

Elena Boykova Ranguelova

Thesis submitted for the degree of Ph.D.

Department of Electronic and Electrical Engineering

University of Dublin, Trinity College

December, 2002

DECLARATION

This thesis has not been submitted as an exercise for a degree at any other university. Except where stated, the work described therein was carried out by me alone.

I give permission for the Library to lend or copy this thesis upon request.

Signed: 

May, 2003

ACKNOWLEDGEMENTS

No research is done in isolation, so I wish to acknowledge the goodwill, advice and support that I have received from many others in completing this work.

Firstly, I would like to thank my supervisor, Dr. Anthony Quinn for providing me with the opportunity to undertake this exciting research. I am very grateful for his expert support, constant encouragement and enthusiasm through this project.

I would like to thank Prof. Frank Boland for his support and Dr. Anil Kokaram for the useful discussionis on variety of image processing and other topics. Special thanks to Joe Timoney, Robbie Dempsey, Declan Flood and Tim Forde for reading through the thesis and suggesting corrections. Thanks to Ludvík Tesař for helping me with LaTex or any other issue. I would like to thank Ciaran McGoldrick, Conor Nolan and Robbie Dempsey for their invaluable help with many technical problems. Thanks to Erika Doyle for helping to sort out all kinds of administrative formalities for me.

On a personal level thanks are due to all postgraduates and my friends in the Department and outside College for they were instead of my family here. Special mentions are due to Ed Clark, Joe Timoney, Cecilia Chan, Conor Brennan, Jonathan O'Donovan, Ludvík Tesař, Andreas Schwarzbacher, Fergus Johnston, Siobhán Launders, Lorcan McManus, Paul Comiskey, Vašek Šmidl, Denis Sidorov, Fabrice Goucem, Rozenn Dahyot, François Pitie and all other great people I have met in Dublin for they have been very good friends indeed. Thank you, Siobhán and Rozenn, for your support in hard times.

The research on this project has been supported financially by the HPC initiative of Trinity College, Dublin and by the Department of Electronic and Electrical Engineering.

Finally, I would like to thank all my family for passing to me their passion for learning and research: thank you for your encouragement through all my life, for your love and unconditional support in all my doings.

SUMMARY

The development of supervised segmentation algorithms for textured images existing on three-dimensional (3-D) lattices is the central theme of this thesis. The current state of the art in 3-D segmentation has been reviewed, and novel approaches have been developed.

The Gaussian Markov Random Field (GMRF) is widely used in modelling of spatial systems in image processing and other areas. In this work, its 3-D extension is proposed for synthesis and analysis of stationary textures. A textured volumetric image is then considered to be a mixture of 3-D stationary GMRFs. The 3-D processing is shown to be the only way forward in cases where the correlation between intensities at neighbouring image nodes along the third dimension are the only features, which discriminate between textures.

A fast algorithm for synthesis of 3-D stationary GMRFs is proposed. It is based on the properties of circulant matrices of level three, which can be exploited using the Fast Fourier Transform (FFT). Voxel classification is achieved via a computational-volume based analysis, presenting an alternative to Bayesian approaches, which are currently popular. These deterministic algorithms offer good performance at a far lower computational cost. The relevant theory for the model parameter estimation on these computational volumes is first presented. Thereafter, the segmentation problem in the image space is transformed into classification in the parameter (feature) space. Fast algorithms for classification of nodes on recursively decimated 3-D lattices are then developed.

Shape evolution of three-dimensional structures along the third (z-going) dimension affects the quality of 3-D segmentation results. This phenomenon is closely related to the motion estimation in image sequence (video) processing, and in image registration. However, there are specific difficulties related to volumetric imagery. In this work, the difference between slices is estimated in order to track shape evolution. Several techniques for estimation and compensation of the difference field are

proposed. They are based either on preliminary segmentation results or on analysis of the original raw grey-level data. Successful difference estimation and consequent compensation of the proposed stationary computational volumes results in enhanced segmentation. Finally, a unifying framework for simultaneous computational volume construction, and segmentation on 3-D lattices, is proposed.

Throughout the thesis, extensive simulation studies are reported, using mixtures of synthetic textures. Furthermore initial experiments with real 3-D Magnetic Resonance Image (MRI) volumes are presented to support the theoretical developments, and to point to an important prospective application context.

To my family and
the memory of my grandfather

Посвещавам на моето семеиство и
на паметта на дядо Еленко

Contents

List of Figures

List of Tables

List of Abbreviations

1-D	one-dimensional
2-D	two-dimensional
3-D	three-dimensional
L-D	L-dimensional
MRI	Magnetic Resonance Image
CT	Computed Tomography
GRF	Gibbs Random Field
MRF	Markov Random Field
GMRF	Gaussian MRF
p.d.f.	probability density function
MAP	Maximum Aposteriori Probability
MCMC	Markov Chain Monte Carlo
PD	Proton Density
BM	Block Matching
MAS	Maximum Allowable Square

H-C	Hammersley-Clifford
DFT	Discrete Fourier Transform
IDFT	Inverse DFT
FFT	Fast Fourier Transform
i.i.d.	independent and identically distributed
LS	Least Squares
VBV	Voxel-by-Voxel
PBP	Pixel-by-Pixel
MD	Majority Decision
UR	Unanimity Rule
RUR	Recursive Unanimity Rule
MO	Moravec Operator
IP	Interest Point
ZNCC	Zero-mean Normalized Cross-Correlation
p.m.f.	probability mass function
KLD	Kullback-Leibler Distance

Chapter 1

INTRODUCTION

1.1 3-D IMAGE SEGMENTATION

Three-dimensional (3-D) digital image processing, and in particular 3-D segmentation, is a rapidly developing research area with applications in many scientific fields such as biomedicine, seismology, remote sensing, material science, etc. 3-D segmentation algorithms are sometimes an extension of existing two-dimensional (2-D) algorithms. However, certain features specific to 3-D images justify dedicated research efforts in the development of 3-D algorithms [61].

There are two main types of 3-D images: image sequences (or video) where the third dimension is time, and volumetric images (or still 3-D images) where the third dimension is also spatial. These are widely acquired in biosciences, for instance, to image intrinsic 3-D objects. In this thesis, we are concerned with segmenting images of the latter type.

3-D image segmentation is an analysis operation that aims at dividing the image into meaningful 3-D regions. For example, in a Magnetic Resonance Image (MRI) of a brain, the regions may correspond to different anatomical structures and tissues such as skull, white matter, grey matter, fat, etc. In this work, the grey level image intensities are considered to exist on 3-D lattices and their grouping into

meaningful regions is referred to as 3-D segmentation. In general, the image elements in a segmented region share similar properties such as grey-level intensity, colour, texture, motion, etc. This work considers the textural property as a basis for 3-D texture-based segmentation.

Markov Random Field (MRF) theory provides a consistent way of modelling image elements which form a texture. MRFs have been used widely for 2-D texture classification and segmentation mainly within a Bayesian framework [13,16,23,46,58,59] as well as for motion analysis, segmentation and restoration in image sequence processing [9, 48, 60, 82]. Usually, the solution of the tasks formulated within the Bayesian framework involves extensive computations. Alternatives, such as the computational volume approach we use in this work, are simpler and give good results for a far smaller computational cost [59].

An extension of the MRF model to three dimensions leads to its definition on 3-D lattices. 3-D MRFs have already been used successfully for 3-D image reconstruction and segmentation [10, 15, 94]. We define the most used MRF model, namely the Gaussian MRF (GMRF), on 3-D lattices.

Difference analysis between 2-D slices is another major topic dealt with in this work. It is related to well-researched areas such as motion estimation in video context and registration. Difference analysis for volumetric textured imagery, however, has not be considered before in relation to the 3-D segmentation. The contribution of the present work is the development of methods for difference estimation and compensation in volumetric image segmentation problems.

1.2 THESIS SYNOPSIS

In Chapter 2 the literature on 3-D segmentation is reviewed. The problem of difference analysis in volumetric imagery along with its relation to the areas of motion estimation and image registration are overviewed. Some specifics of the MRI application context are explained.

The GMRF on 3-D lattices is presented in Chapter 3 for the purposes of modelling volumetric textures. The theory of stationary GMRFs is given and the notation used in the rest of the thesis is introduced.

Toroidal boundary conditions and symmetric 3-D GMRF topology result in a circulant structure of level three for the interaction matrix within the joint GMRF probability density function (p.d.f.). Therefore, Chapter 4 starts with presentation of the theory of circulants of level three. The main theoretical result is used to develop a fast 3-D FFT-based GMRF synthesis algorithm.

In Chapter 5, least squares estimation of 3-D GMRF parameters is presented. A technique for node classification is then developed and its performance is illustrated in segmenting mixtures of synthetic 3-D GMRFs.

Fast techniques for segmenting such 3-D mixtures based on recursive decimated lattice topology are developed in Chapter 6. They are shown to overcome the intrinsic limitations of the technique in Chapter 5. The problem of difference along the z-going dimension and its impact on the quality of the segmentation result is explored.

In Chapter 7, methods for difference estimation and compensation using an initial label field from a preliminary segmentation are developed. These methods are inspired by techniques from the image registration and motion analysis literature. They are modified to cope with the properties of label field data. Their ability to improve the results obtained by the recursive methods in Chapter 6 is illustrated.

A novel method for difference estimation and segmentation using he original grey-level data and *a priori* texture codebook information is proposed in Chapter 8. A unified technique for volumetric segmentation, based on entropy minimization is finally developed. It offers the prospect for optimized choice of computational volume size. The methods from Chapters 7 and 8 are tested on both synthetic images and real MRI volumes.

In Chapter 9, the main conclusions are presented and direction for future research challenges in the area are suggested.

Chapter 2

SEGMENTATION OF TEXTURED IMAGES: AN OVERVIEW

2.1 INTRODUCTION

The goal of image segmentation is to identify homogeneous regions in an image. The homogeneity can be based on one or more of several properties, such as texture, colour, distribution of the densities of the image elements, motion field, etc. The result of the segmentation is either an image of labels identifying each homogeneous region, or a set of contours which describe the region boundaries.

Image segmentation can be performed on 2-D images, sequences of 2-D images, 3-D volumetric imagery or sequences of the latter. In general, then, we would wish to address the segmentation task in the context of images defined on lattices of dimension three. Most of the image segmentation research has focused on 2-D images. If the data is defined on 3-D lattices, such as obtained from series of cross-sectional computed tomography (CT) or MRI images, then typically, each image ("slice") is segmented individually. The discriminant features of such data exist however, on any orientation of the slices therefore, encourage the development of ntegrated 3-D algorithms.

There is a wide variety of techniques available for segmenting images. Most of the approaches have been extended from 2-D to 3-D. A common classification of these methods in the literature [61, 94] is as follows: thresholding, edge detection, clustering, region growing, and active contours. One of the most common approaches employs some method of edge detection, after which the segmentation result is proposed as a set of detected region boundaries. Another common approach to segmentation involves examining the textures present in an image and determining regions of homogeneous texture. In this work, we are concerned with the latter approach for segmenting volumetric images. The aim is to produce 3-D label maps rather than region boundaries, since they can be obtained from the regions, but not vise versa.

This chapter gives a brief overview of the main classes of 3-D segmentation methods in Section 2.2, followed by discussion of the texture segmentation problem in Section 2.3. The relation of difference analysis in volumetric imagery- which is used in this work- to motion estimation, image registration, etc. is discussed in Section 2.4. Finally, some background material relating to MRI is presented, as a prospective application context for the algorithms developed in this work in Section 2.5. Conclusions are drawn in Section 2.6.

2.2 MAIN CLASSES OF 3-D SEGMENTATION METHODS

Mathematically, image segmentation is well-defined. An image consists of an array of elements, and the aim is to give each element a *label.* There are different approaches to achieve this goal. Here the main classes of 3-D segmentation are presented.

2.2.1 Thresholding

The simplest segmentation technique is *thresholding*. The threshold segmentation separates the image elements (pixels in 2-D or voxels in 3-D) based only on their intensity. Thresholding has been used in CT images to segment bones and soft tissues from background [61]. The success of this approach depends on the successful selection of a threshold. If the 3-D image data histogram exhibits two well-defined modes, for example, then setting the threshold at the minimum between the modes can give satisfactory results. The thresholding methods have many variations: global (single threshold) or local threshold (depending on the position in the image), multi-thresholding, adaptive thresholding, etc [81].

Thresholding is very susceptible to noise in low contrast images. It tends to produce scattered groups of pixels rather than connected regions. Connectivity algorithms are regularly used as a post-processing step. Generally, threshold-based methods are not suitable for segmenting textured images. Their perceptual qualities are based on higher order interactions, but not the first-order image intensity properties addressed by histogram-based thresholding. It is possible, however, to apply thresholding not on the original grey-level data, but, for instance, on the local variances computed over 3-D blocks [61]. Gregson [32] uses thresholding to detect tissue from the background in 3-D MRI of a heart.

2.2.2 Edge Detection

Methods of *edge detection* are not segmentation methods by nature but the result of the edge detection can be used as a pre-processing stage in segmentation. The main aim is to extract discontinuities (i.e. to identify changepoints). Many operators are used to extract edges such as Roberts, Laplacian and Sobel. A full review is available in [81]. Often, edges indicate region boundaries and they are useful for segmentation but, sometimes regions and edges are not well related. Many edge-linking methods exist for inferring connected regions from edge maps solving the problem and for

performing false edge removal [81, 90].

3-D objects can be described by the volume they occupy or by their bounding surface. Here, the terms *edge* and *edge detection* continue to be used. Just as in 2-D, edges are defined as discontinuities (changepoints) in a chosen image property, be it intensity or a fuller description of the local image statistics [40]. The image discontinuity operators such as the Sobel and Prewitt can be easily extended to 3-D. The Marr-Hildreth operator has been extended to three dimensions as well [61].

In [94], Wu reports on the application of 3-D edge detection methods in combination with matched filters for segmentation of CT and MRI data [95] using Laplacian and Gaussian operators.

2.2.3 Clustering Techniques

Clustering refers to the classification of objects into groups according to certain properties of these objects. From an image segmentation point-of-view, one aims at extracting a *feature vector* from local areas in the image, which will then form *clusters* in the feature space. The task of segmenting the data represented by multi-dimensional feature vectors into a finite number of homogeneous regions is equivalent to dividing the feature space into this number of partitions, each presumably, occupied by a closely associated set (cluster) of features. A standard procedure for clustering is to assign each voxel to the class of the nearest cluster mean [85]. There is a variety of clustering algorithms in the literature, the most popular and simple being the K-means technique [39]. In the case of scalar features (e.g. voxel intensities) clustering can be seen as a method of determining $K-1$ thresholds in the 1-D feature space.

A widely used approach to forming clusters is via *neural networks* or *wavelet transforms*. Borş and Pitas [7] use a supervised α-trimmed mean Radial Basis Function network (neural network) to segment 3-D microscopy images of blood vessels. Each object of the 3-D data is represented as an ellipsoid and their centroids

are estimated using the extended 3-D Hough Transform [81].

The multi-scale approach to feature extraction using a Gabor filter bank has been employed in [68] for pairwise data clustering in unsupervised texture segmentation.

Chang, Tekalp and Sezan [10] use a *Bayesian Maximum a Posteriori (MAP) Probability* approach for automatic segmentation of 3-D MRI data. They apply a modified K-means algorithm where *a priori* information about the spatial connectivity of the segmentation is incorporated via a 3-D Gibbs Random Field (GRF) model. They report excellent performance and superior results compared to segmentation based on conventional K-means clustering for each 2-D slice.

2.2.4 Region Growing Techniques

To produce spatially connected regions requires a criterion of geometrical proximity, in addition to the homogeneity criterion. A simple approach to image segmentation is to start with some user-defined voxels (*seeds*) representing distinct image regions and to *grow* them until the whole image is covered [67]. In this procedure, each region grows until it collides with another region or reaches an area with different properties. Some voxels may remain unclassified. Region growing cannot be used for segmenting complex 3-D volumes, but is useful in segmenting volumetric images composed of large homogeneous regions.

Most region growing algorithms are based on *splitting* and *merging* techniques [61, 94]. The image, or its regions, are split into smaller regions, or small regions are merged together, or splitting and merging are applied alternatively until the homogeneity criteria is satisfied.

Region growing techniques are widely used both for segmenting 3-D medical imagery and image sequences. A 3-D method using a GRF was proposed by Aach [1] to segment different organs and tissues of a MRI of the human head. 3-D segmentation as a pre-processing step in volumetric medical image reconstruction, using supervised region-growing 3-D Markov Random Field (MRF) based method, was

proposed by Choi in [15]. The work was applied to 3-D images of the femur and the knee-joint. Their comparison with a 2-D counterpart reveals the superior performance of the former.

A segmentation and object tracking algorithm for an image sequence is used in [33]. It is of a region-growing type, working on the first two frames and using the motion as a main feature, combined with colour or intensity.

3-D segmentation of an image sequence by 3-D binary split-and-merge methods was proposed by Willemin and Reed in [91]. The criterion for splitting/merging is found using a 3-D polynomial approximation of the regions and the most similar regions are merged using a region adjacency graph. The result of the segmentation is then used for an effective compression and coding scheme eliminating the sequence redundancy. Wu [94] uses a similar approach to segment image sequences.

2.2.5 Active contours

Deformable models, called *active contours* or *snakes* in 2-D and *active surfaces* or *active balloons* in 3-D, were introduced in [45] and were further developed in [18,86]. A deformable surface behaves like an elastic sheet. Initially, it is placed close to an object's boundary and it changes shape to match the shape of the boundary. The forces which change the shape are the external (or image) forces- controlled by the image attributes- and the internal forces, which control the surface regularity [61].

2.2.6 Bayesian Methods

Bayesian methods use probability calculus to quantify the plausibility of a hypothesis. In the case of image segmentation, this hypothesis is about the existence of a particular "hidden field" (label field realization) along with the data. *A priori* knowledge, which can be exploited to improve the results, is used to regularize the inference of the hidden field, given the data. Formal optimization techniques are then used to work on the posterior inference.

The Bayes rule states that:

$$P(\mathbf{L}|\mathbf{X}) \propto P(\mathbf{X}|\mathbf{L})P(\mathbf{L}), \tag{2.1}$$

i.e. the posterior probability $P(\mathbf{L}|\mathbf{X})$ of the label field given the data is proportional to the product of the model probability $P(\mathbf{X}|\mathbf{L})$ and the prior probability of the label realizations $P(\mathbf{L})$. $P(\mathbf{L})$ is defined using local information about the expected segmentation result (such as shape etc.) and aims at encouraging spatial connectivity. For the a priori p.d.f. , GRFs usually are used [10,16,59,85]. The specification of the prior is application-dependent and can cover a wide range of applications. Examples include robust unsupervised segmentation of noisy non-textural images [16], for unsupervised segmentation of textured images [58], for Brain MRI segmentation [37].

The MAP approach aims at maximizing $P(\mathbf{L}|\mathbf{X})$, given the model and the prior. The identification of the MAP segmentation is an optimization task in a huge number of dimensions. It has been attacked via Stimulated Annealing [31], *Markov Chain Monte-Carlo (MCMC)* methods, etc.

The Bayesian methods offer a general framework which can be employed for 2-D or 3-D image data modelling and can incorporate very complex models. These include modelling of spatial and temporal *discontinuities*, *motion* and *occlusion*, etc. These phenomena are important in image sequence processing applications [48,60,85].

2.3 TEXTURE

Whereas segmentation has at least a formal definition, texture has not. A typical definition in the literature is "one or more basic local patterns that are repeated in a periodic manner." [66]. However, it is not clear exactly what a pattern might be or how it might repeat. It is not even clear whether texture is an inherent property of all objects. Even though texture is an intuitive concept, its definition has proven difficult to formalize. Haralick, Shanmugam and Dinstein [35] noted that

"texture has been extremely refractory to precise definition". Over the years, many researchers have expressed the same statement: "There is no universally accepted definition for texture" [21], "Texture eludes precise definition" [66].

Despite this lack of a universally agreed definition, all researchers agree on two points. Firstly, there is significant variation in intensity levels between nearby image elements within a single texture. Secondly, texture is a homogeneous property at some spatial scale larger than the resolution of the image. Some researchers describe texture in terms of the human visual system i.e. that textures do not have uniform intensity, but are nonetheless perceived as homogeneous regions by a human observer [8, 12]. However, a definition based on human perception poses problems when used as the basis for a quantitative texture analysis algorithm.

It is very hard to define the goal of a texture segmentation algorithm, even if the question is restricted to one image taken from the natural world. Texture segmentation is not equivalent to object segmentation. Therefore, there is no one segmentation of an image that can be considered to be "right." The "right" segmentation exists only in the mind of the observer, which can change not only between observers, but within the same observer at different times.

2.3.1 Main Texture Models

Despite the lack of a formal definition, texture does exist and it can be segmented to some extent. Researchers have been developing many *texture models*, with the goal of finding the "basic pattern mechanism" and of modelling it [66]. The main computational approaches to the definition of texture are listed briefly.

Non-parametric quantification of texture

Statistical texture description is suitable for statistical pattern recognition applications. In general, the purpose of a texture model is to provide a means of transforming a local area of an image into a set of numbers, namely the *feature vector*.

The statistical methods exploit the local correlations of the image elements. Some of these methods are applied to the raw image data, others are transform-based.

Measuring spatial frequencies is a basis for many texture recognition methods. Usually, the frequencies are determined from the *autocorrelation function* of the texture or by applying an *optical* or *discrete image transform* [81]. Transforms like Fourier, Hamadard etc. can be used. The description of noisy textures becomes a difficult task. A joint spatial/spatial frequency approach is recommended [61]. The Wigner distribution was used successfully for a variety of synthetic and Brodatz textures in [75].

The *co-occurrence matrices* method describe the texture via the repeated occurrence of some grey-level configuration in the texture [35]. Texture classification can be based on a criterion derived from the co-occurrence matrices such as energy, entropy, maximum probability, contrast, correlation, etc. Although the co-occurrence matrices give good discrimination between textures, the method is computationally expensive [81]. A fast algorithm for computation of the co-occurrence matrix is given in [2].

Edge frequencies can also be used to describe a texture. These methods are easy to implement, but the results are dependent on the choice of the edge detector. A texture classification approach, which is robust to noise, has been developed using a Canny edge detector in [47].

The *grey-level run length method*, based on computing the number of neighbouring pixels of the same grey level in different directions, was used to classify terrain images [30].

The *Laws' texture energy measures* [51,52] are another approach generating texture features by using local masks to detect various types of textures. This approach measures the amount of variation within a fixed-size window. A set of nine 5×5 convolution masks is used to compute texture energy. The 2-D masks are obtained via outer product of pair of convolution vectors corresponding to a grey-level, edge, spot and ripple. Then a *texture energy map* is composed from the filtered images

obtained using each mask. The final result is nine energy map images which can be viewed as a single image with a vector of nine elements per pixel. These attributes then can be used to cluster an image into regions of uniform texture. The Laws' measures of texture have been used to classify ultrasonic liver images in [93].

The *mathematical morphology* approach uses structure primitives and looks for their statistical repetitiveness in the texture. It has been successful in describing granulated materials in [25].

Texture description is highly scale-dependent. *Multiscale methods* rely on decompositions that confine particular texture attributes to distinct scales. Wavelets can act as basis functions for this approach [55]. Pyramid structures for wavelet decomposition were used for texture classification in [11]. An overview and comparison of Gabor and wavelet transform performance is given in [88].

Textures can also be modelled as processes with statistical fractal (or self-similar) properties [34].

Structural quantification of texture

The structural approaches are not as widely used as the statistical approaches. They develop a description using texture primitives and syntactic rules. The latter exploit the analogy between the spatial relations among texture primitives and the structure of a formal language. The main approaches are *shape chain and graph grammars* [63] or *fractals* [12,56]. The *fractal-based* texture description was introduced in [64] using the correlation between the texture coarseness and fractal dimension of a texture.

Parametric models of texture

Models can be found to explain the nature of texture. Then the image data are used to determine the parameters of the model. Parameter estimates may then be used as feature vectors for the classification process. The advantage of the model-based approaches is that they can be used both for texture analysis and synthesis. A good survey of the usage of random fields in image analysis is given in [26].

A model that is defined on a space that can theoretically contain any pattern is the Markov Random Field (MRF) model. Unlike most models, MRFs are capable of generating random, regular and even highly structured textural patterns [66]. The MRF approach originates from both the Gibbs distribution in statistical mechanics and from Markov probability theory. The model has been used widely since the seminal work of Geman and Geman [31]. The theory of MRFs is extensively presented in [92]. Cross and Jain [21] have shown that MRFs can be used for modelling natural scenes. 3-D extensions of the MRF have been used for volumetric medical image segmentation and reconstruction in [10, 15, 37].

A specific MRF model, called the Gaussian MRF (GMRF) has been proposed by Chellappa [14]. According to this model, the image element grey-levels have a jointly Gaussian distribution. The GMRF is the most widely used process for the modelling of various natural and man-made textures. Its effectiveness has been shown for classification and synthesis of Brodatz textures in [13, 57].

In this thesis the GMRF is defined on 3-D lattices to accommodate volumetric textures. Local quantification (estimation) of the GMRF via computational volumes is adopted in preference to Bayesian approaches, particularly in view of their relatively high computational load.

2.4 DIFFERENCE ANALYSIS

A major challenge in 3-D image processing, one which does not arise in the 2-D case, is the phenomenon of *difference* between the 2-D frames or slices. This difference (known also as disparity [3]) is the result of the change of the scene in video sequences (object and /or camera motion) or of the shape evolution of a 3-D object cross-section in volumetric imagery.

MRF models obey the Markov property, i.e. the grey-level at a particular image element depends on the grey-levels within a *neighbourhood* of this element. The difference phenomenon affects the geometry of the neighbourhood and therefore

adjustment and compensation for this effect is required. 3-D motion-compensated neighbourhood geometry for an Autoregressive Model in image sequence processing has been used in [48]. The difference between the frames is the displacement of the position of corresponding pixels belonging to moving objects, whose images are captured in different frames of the sequence corresponding to different moments of time. In the video processing context the difference is due to the motion and hence can be quantified via the various *motion estimation* methods.

In 3-D volumetric imaging, however, the difference between consecutive 2-D slices is not motion-induced. It is caused by the 2-D cross-section shape evolution of a 3-D object (anatomical structure in the medical imaging context) along the third dimension. Hence the neighbourhood geometry must not be displacement/motion-compensated. However, a simple parallelepiped-shaped computational volume, which spans many slices and overlaps different 3-D textures must address the textural non-stationarity problems which arise. Texture *stationarity* is a required assumption within the computational volume, if model parameter estimation is to be successful. Shape evolution in volumetric images often violates this assumption and thus motivates the development of new difference estimation and compensation approaches. This is one of the principal goals of this work.

Although it has its own specific requirements, difference analysis in volumetric imagery required tools which have been well studied in motion estimation, the image correspondence, and registration. These are briefly reviewed now.

2.4.1 Motion Estimation Methods

The motion of 3-D objects and of the camera projects onto the image plane, and induces a 2-D motion. This 2-D motion is called *apparent motion* or *optical flow* [82, 85]. The optical flow assigns a *displacement vector* to each image element and can be recovered from analysis of the intensity or colour in an image sequence. The optical flow estimation is an *ill-posed* problem and so needs a model that makes assumptions

about the structure of the 3-D motion field. Such a model can be classified as parametric or non-parametric [85] and the choice is application dependent [82].

Parametric models

In many applications, the motion field is described in parametric form using a small finite number of parameters. Different models are described and compared in [82]. They depend on the projection coordinate system, and include as the *6-parameter affine model* with orthographic projection, and the *8-parameter non-linear model*, derived from the perspective projection of the 3-D motion into 2-D. The main drawback of these methods is that they are applicable only in the case of 3-D rigid motion.

Non-parametric models

Another approach is to impose a *non-parametric* smoothness constraint over the 2-D motion field.

The *optical flow equation* [38] provides an estimate of the optical flow in terms of spatio-temporal image intensity gradients. These are known as *gradient-based methods.* The spatio-temporal smoothness constraint suggests that the displacement vector varies only slowly over a neighbourhood. Global constraints cause inaccuracies and some modifications are required (directional smoothness etc.). A survey of the methods is given in [82]. For these methods to be successful, it is necessary that there be little change between consecutive images [81].

In the *block motion* model [48, 85], it is assumed that the image consists of moving blocks. Then, the matching of the moving blocks can be performed using *block-matching* (BM) or phase-correlation techniques [48]. These techniques handle translational motion. However, generalized deformable motion models can be incorporated, but these significantly increase the computational load.

Pel-recursive methods are of the predictor-corrector type. The prediction at a pixel is formed by a linear combination of the motion estimates in a neighbour-

hood. The update is based on a *gradient-based* minimization of the displaced frame difference. The extension of this approach to block-based estimation is known as *Wiener-based motion estimation* (WBME) [48].

Bayesian methods utilize a probabilistic smoothness constraint, usually a Gibbs Random Field (GRF),with a MAP estimate being sought. This is an unifying approach, from which all the other methods (gradient-based, BM, WBME) can be derived [48]. Its main drawback is the heavy computational load.

Differential motion analysis

An alternative to the aforementioned methods is the one based on a simple subtraction of images from different frames. A binary *difference image* is constructed by grey-level subtraction and thresholding, where non-zero regions in the result are- in large measure- presumed to be a consequence of motion. To detect the direction of motion, accumulative difference pictures can be constructed, setting one initial frame as a reference [41–43]. This method is very straightforward to implement. It works well for scenes where there ia a sufficient number of frames to be processed and when an object has been moved eventually from its original position.

2.4.2 The Correspondence Problem

The *correspondence problem*, in terms of frames of a video, is the process of "establishing a match between parts of one frame and their counterparts in a subsequent frame that represents the same object at a latter time" [87]. Ullman [87] argues that the correspondence problem is low-level and autonomous. It is tackled by *matching* between some so-called *correspondence tokens*, i.e. the basic image elements, which have to be matched. These correspondence tokens can be single intensity points, sub-regions, edges, corners, etc. The matching is based on a similarity measure defined between the correspondence tokens.

There are many disadvantages in using the raw grey level image elements as

correspondence tokens, so the original data is typically first organized into units, such as interest points, edges, line segments or blocks. Thus, the correspondence problem matches low level tokens, which represent meaningful items on a higher level.

The motion estimation problem can be approached as finding a correspondence of Interest Points (IP). The first step is the detection and then the matching of IP located in two image frames. An iterative algorithm, which maximizes the global probability between all possible correspondences, has been developed in [3]. This algorithm has the drawback of assuming rigid body motion.

In [28], a *non-rigid* assumption has been used for analysis of the cardiac wall motion from MRI. IPs were obtained via magnetically applied pulse sequence, which result in a rectangular grid of markers in the image, that changed their position during the cardiac cycle. The correspondence between these IPs is determined using a correlation technique.

2.4.3 Registration

Image *registration* is a special case of the correspondence problem where the two frames are globally shifted in respect to each other [85]. Registering of two data sets requires the evaluation of a transformation which, when applied to the first data set seeks to align it to the second [61].

A straightforward approach to image registration is to apply some matching technique. A widely used method is *cross-correlation* function computation between two windows: a template window in the first image, and a set of candidate windows in the second image. The primitives within these windows can be both the original grey-levels and some extracted features such as edges, curves, etc. The matching strategy can be performed as an extensive search or using a search strategy in one or more resolution levels (image pyramids).

The registration problem is closely related to disparity analysis and motion es-

timation and to the difference analysis treated in this work. In [96], a disparity analysis method based on boundary matching was used for the registration of the 2-D coronal sections of autoradiographic images in 3-D data reconstruction. In [89], a motion estimator has been used for medical image registration of brain images. It was shown to work for synthetic rigid and affine motion sequences as well as for registering 3-D brain datasets. The cross-correlation BM method has been used in [62] for the alignment of histological sections of rat's brain for the purpose of 3-D reconstruction. A good survey of medical image registration techniques is presented in [54].

2.5 AN APPLICATION: MRI

Accurate segmentation of volumetric medical imagery is an essential step in many applications. It is an important pre-processing step before 3-D rendering, radiation therapy and surgery planning, and in the automatic assessment of signal characteristics from potentially malignant tissues. Magnetic Resonance Imaging (MRI) and Computed Tomography (CT) are the main sources of 3-D images in medicine. Here, a brief presentation of the specifics of the MRI acquisition process for brain imaging is given.

When protons in a hydrogen atom (hence in the water within the brain tissues) are placed in a magnetic field, they oscillate with a frequency depending on the strength of the field. They are capable of absorbing energy from the field and when it is switched off they return to their equilibrium by transmitting the absorbed energy. This re-radiation of energy is observed as the MRI signal. The intensity of a voxel from the MRI data corresponds to averaging the signal over a small area of the brain and over an interval of time. Usually, the tissue area is 1mm^2 in the plane (or slice) parallel to the MRI detector. After a slice has been obtained, the detector moves along the third axis to acquire another image. The slice thickness is usually $3 - 5$ mm and the gap between slices is usually 2 mm. Therefore, there can be a strongly

perceptible difference in the cross-section shape of various anatomical structures between adjacent 2-D slices. The image volumes are obtained by stacking the slices together along the third dimension. The phrase 'third dimension' denotes the axis, along which the resolution is lower. However, the 2-D slices defined along the higher resolution can be physically any of the axes. The 2-D MRI scans can be acquired coronally, sagitally or axially (transaxially) (Fig.2.1), but are all fundamentally 3-D data. An element of a slice is correlated not only to its spatial neighbours within the same slice, but also with spatially adjacent neighbours in nearby slices.

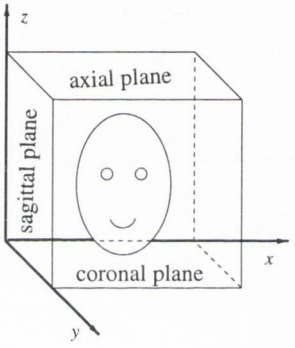

Figure 2.1: MRI planes.

The standard slice orientation is transaxial (or axial) (see Fig. 2.2, left). Slices with sagittal and coronal orientation are shown in Fig. 2.2, middle and right respectively.

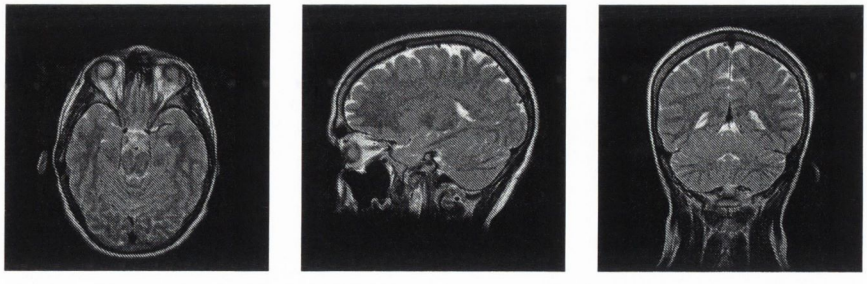

Figure 2.2: Axial, sagittal and coronal views.

The return of the hydrogen nuclei to their equilibrium state takes some time, and is governed by two physical processes. The first is the relaxation back to equilibrium

of the component of the nuclear magnetization, parallel to the magnetic field, which takes time T1 and the second is the relaxation of the perpendicular to the magnetic field component which takes time T2. Hence, the strength of the observed MRI signal depends on three main parameters: the Proton Density (PD) in the tissue (the greater the density, the larger the signal), and the times T1 and T2. For most soft tissues in the brain, the proton density is very homogeneous, and therefore does not contribute signal differences to the final image. The times T1 and T2, however, can be dramatically different for various soft tissues, causing major contrast between them in the resulting image. It is possible to manipulate the MRI signal by changing the way in which the nuclei are exposed to the electromagnetic energy. In this way, the dependence of the final MRI image on the three parameters can be specified by weighting techniques [44]. Figure 2.3 illustrates the same physical slice of the brain as a PD (left), T1-weighted (middle) and T2-weighted (right) image.

Figure 2.3: PD, T1 and T2-weighted axial images of a human brain.

When selecting the type of weighting, a tradeoff is made between factors such as cost, time, signal-to-noise ratio, etc. Considerations about the comfort of the patient are also important in this selection. For instance, the T1 images give anatomical details, but tend to be noisy due to the short acquisition time (< 1000 ms for one slice). T2 images possess bigger contrast between the tissues but take longer to acquire ($3000 - 4000$ ms.). The PD images (typical acquisition time: 2000 ms.), generally manifest the smallest contrast between the tissues. Hence, PD images present the greatest challenges for anatomical segmentation.

2.6 CONCLUSIONS

There exist various techniques for image segmentation. Most methods have been extended from 2-D to 3-D, although specific issues such as the presence of difference along the third dimension are inherent only to the latter. Random field-based modelling has been extensively used, due to its suitability both for analyzing and synthesizing images.

The result of the texture segmentation task is very difficult to assess automatically, and usually, the human observer is the final judge of the quality of the segmentation. Texture segmentation of volumetric imagery encourages a model-based approach, hence a 3-D extension of the GMRF will be proposed in this work. Estimation of local properties using a computational volume approach benefits from fewer computations in comparison with Bayesian MCMC algorithms, for example. It requires, however, the development of suitable difference estimation and compensation techniques. This problem, although related to motion estimation, correspondence, and image registration tasks in image sequence analysis, has a different origin and effect in volumetric images.

An important current source of 3-D images is MR technology in medicine. Identification of anatomical structures in PD and T2 brain MRI is a challenging task. Hence, the anatomical segmentation of such volumetric images offers an important application context for 3-D texture segmentation and difference analysis algorithms.

Chapter 3

IMAGE MODELLING ON 3-D LATTICES

3.1 INTRODUCTION

For the purposes of image analysis and processing it is efficient to have an underlying model for the dominant characteristics of the given data. Although it is often difficult to identify the physical mechanism which generated the observed data, any analytical expression that captures the statistical properties of the image can be used as a model.

Specifically, Markov Random Field (MRF) theory provides a consistent way of modelling context-dependent entities [53]. MRFs are useful for image processing because they can describe local properties and are capable therefore of representing textured areas as well as region sizes, shapes and boundaries. Two-dimensional (2-D) MRFs have been used for modelling 2-D images and other spatial systems [14, 60].

When dealing with video sequences and volumetric data it is natural to use the information provided by the third spatial or temporal dimension and this has led to the extension of the MRF models over three dimensions (3-D). Such MRF-based modelling, segmentation and restoration was successfully applied to different types

of 3-D data: volumetric medical imagery [10, 15, 37] and image sequences [94].

In this chapter the problem of modelling volumetric images via 3-D MRFs is treated. Definitions of the basic notions and notation used through the thesis are given in Section 3.2. The elements of MRF theory that are drawn on for this work are outlined in Section 3.3. The specific model used in the thesis, the 3-D GMRF, is presented in Section 3.4 and in Section 3.5 the labelling problem is stated. Finally, the main conclusions are given in Section 3.6.

3.2 FINITE LATTICES

Typically image data are represented by grey-level variations defined over a finite rectangular or square point lattice.

Definition 3.1 *A* **lattice** *is a two or three-dimensional finite set of nodes.*

A lattice **node** is a point in 2-D or 3-D Euclidean space. Thus, 2-D lattices are often used to represent 2-D images (the image element corresponding to a node is called a *pixel*) and 3-D lattices to represent 3-D data where each node corresponds to a *voxel*.

First, denote a lattice by Λ, which is of dimensions $M \times N$ in the 2-D space and of dimensions $M \times N \times D$ in the 3-D space. A lattice node t is uniquely specified by its coordinates $t = (i, j)$ in the 2-D and $t = (i, j, k)$ in the 3-D case, where i is the image row number (along the x axis), j is the image column number (y axis) and k is the slice number (z axis) in a volumetric stack. Thus, the 2-D lattice is $\Lambda^{2D} = \{t = (i, j) | 1 \leq i \leq M, 1 \leq j \leq N\}$ and the 3-D is $\Lambda^{3D} = \{t = (i, j, k) | 1 \leq \leq M, 1 \leq j \leq N, 1 \leq k \leq D\}$. The total number of nodes in the lattice is N_T $N_T^{2D} = MN$ in 2-D and $N_T^{3D} = MND$ in 3-D). The superscripts will be omitted if he dimensionality does not matter or is clear in the context.

The lattice is visualized as an orthogonal grid with nodes lying at the intersections of the grid lines as shown in Fig. 3.1 where the node t is specified by its

coordinates. Only the outer nodes are shown for Λ^{3D} for simplicity.

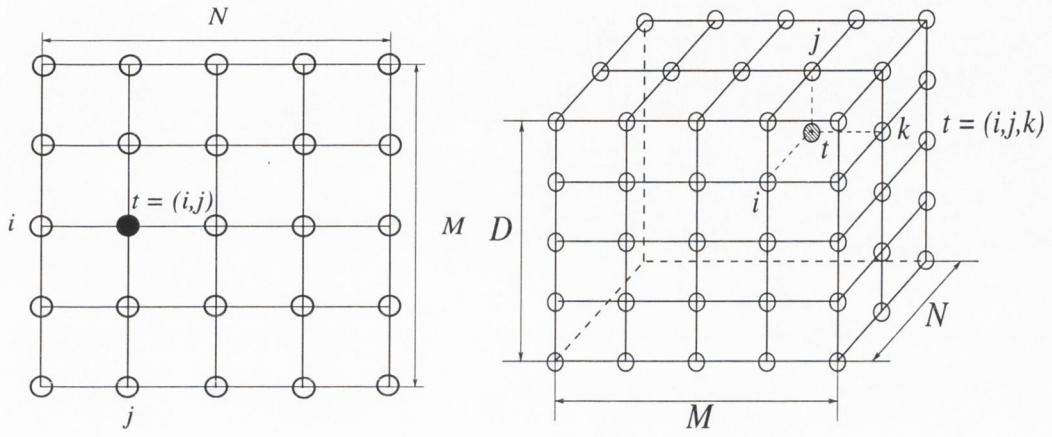

Figure 3.1: Lattice and lattice nodes in 2-D and 3-D respectively.

The image data elements are encoded as 2-D or 3-D array entries, usually as integers in the range $\{0, \ldots, 255\}$ for storage and processing via digital computers. Thus, an alternative representation of the lattice can also be used, namely via the image array of size $M \times N$ in 2-D or of $M \times N \times D$ in 3-D, where a node is depicted as square or cube for an image pixel or a voxel respectively. This is illustrated in Fig. 3.2 where the node is located at row i, column j and on slice k (in 3-D) of the array.

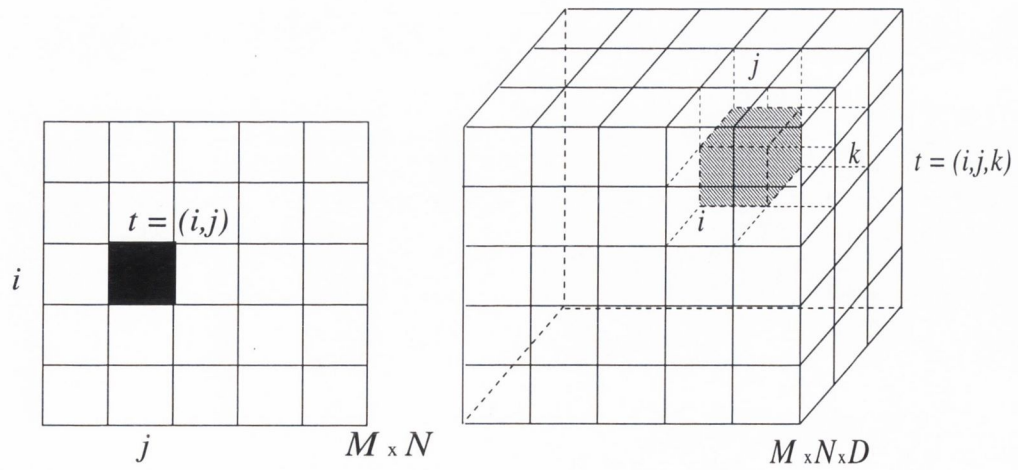

Figure 3.2: Image array in 2-D and 3-D respectively.

3.2.1 Neighbourhood

One of the important characteristics of the image data is the special nature of the statistical dependance of the grey level at a lattice node on those of its neighbours [14].

Definition 3.2 *A* **neighbourhood** η_t *of the node* t *is a subset of lattice nodes* $\eta_t \subseteq \mathbf{\Lambda}$*, which satisfies the following two properties:*
(i) $t \notin \eta_t$*, i.e. node* t *does not belong to its own neighbourhood, and*
(ii) if $s \in \eta_t$ *then* $t \in \eta_s$*, i.e. if* s *is a neighbour of* t*,* t *must be neighbour of* s *and vice versa.*

A **neighbourhood system** $\boldsymbol{\eta} = \{\eta_t | t \in \mathbf{\Lambda}\}$ is the collection of neighbourhoods of all nodes. In this work *symmetric* neighbourhoods, i.e. for which (ii) holds, are used. The neighbourhood $\eta_t(q)$ depends on its *order* q and the sequence of neighbourhoods is recursive:

$$\eta_t(q+1) \triangleq \eta_t(q) \bigcup \arg \min_{s \notin \{\eta_t(q) \bigcup \{t\}\}} D(t,s),$$

where $D(t,s)$ is a predefined distance between two nodes. Then, all nodes s belong to the neighbourhood η_t if their location is within an area centred at the node t defined via $D(t,s) \leq R$ where R takes integer values. This rule is known as the Maximum Allowable Square (MAS) rule [77] as $R = MAS(q)$ denotes the maximum allowable square of the distance of t to any of its neighbours s. The values of q and R are tabulated in Table 3.1 for 2-D and 3-D lattices respectively.

An illustration of the hierarchical nature of the neighbourhood up to the 8-th order for 2-D and 3-D lattices is presented in Fig. 3.3 and Fig. 3.4 respectively. The numbers indicate the lowest possible order of the neighbourhood of the central node (denoted in black on the figures), which satisfies the property $\eta(q_1) \subset \eta(q_2), q_1 < q_2$. In the 3-D case the neighbourhood geometry over the current (k) and previous $k-1, k-2, k-3)$ slices is shown. The notation q_1/q_2 stands for voxels common to both the q_1-th and q_2-th order to illustrate the exact support for each order. The nodes denoted in grey have the same i and j coordinates as the node in black.

Table 3.1: Neighbourhood order and MAS values.

q	$R = MAS_{2D}(q)$	$R = MAS_{3D}(q)$
1	1	1
2	2	2
3	4	3
4	5	4
5	8	5
6	9	6
7	10	8
8	13	9

	8	7	6	7	8	
8	5	4	3	4	5	8
7	4	2	1	2	4	7
6	3	1		1	3	6
7	4	2	1	2	4	7
8	5	4	3	4	5	8
	8	7	6	7	8	

Figure 3.3: 2-D neighbourhood up to eight order.

The geometry of the 2-D and 3-D first, second and third order neighbourhoods s shown in Figures 3.5 and 3.6 respectively. It is derived from the general neighbourhood geometry shown in Figures 3.3 and 3.4.

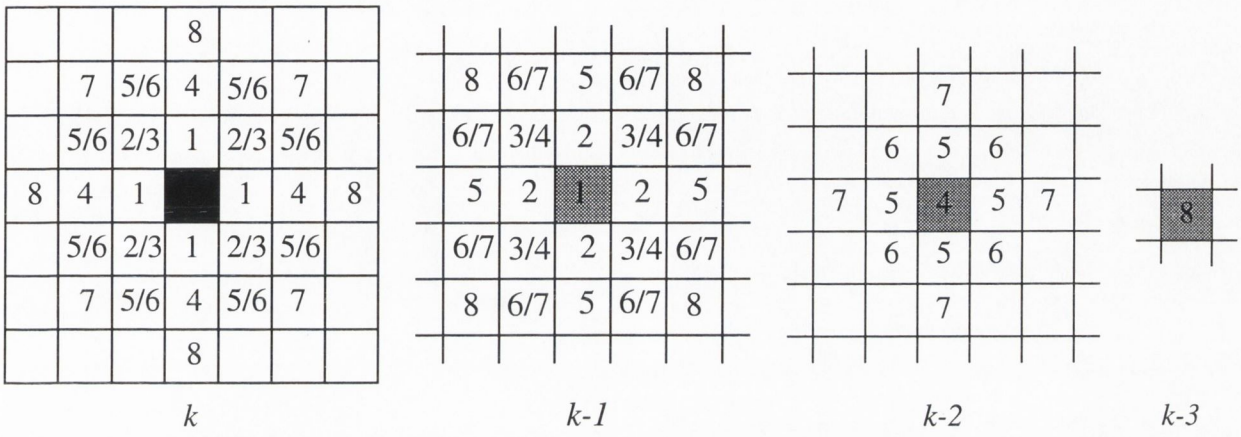

Figure 3.4: 3-D neighbourhood up to eight order.

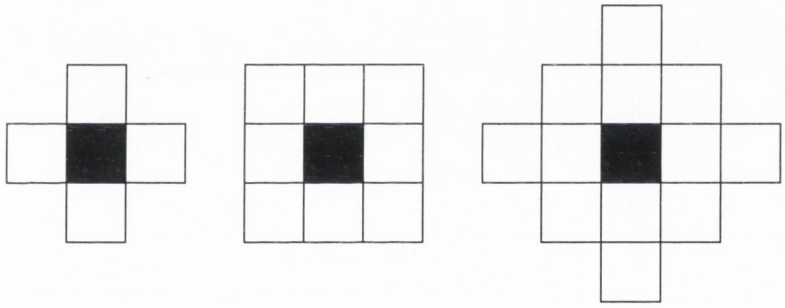

Figure 3.5: 2-D neighbourhood geometry for first, second and third order respectively.

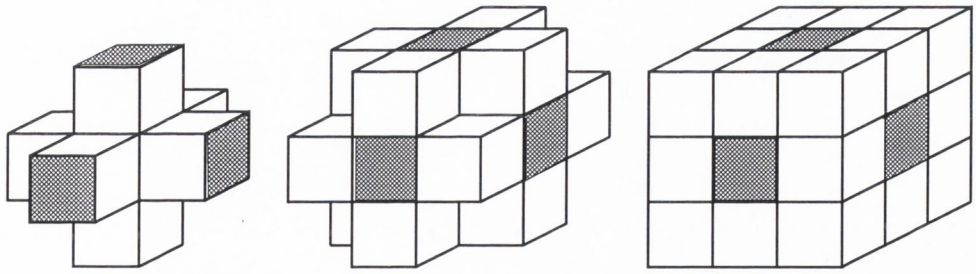

Figure 3.6: 3-D neighbourhood geometry for first, second and third order respectively.

Because a finite lattice is considered, assumptions about the lattice boundaries should be made. Often toroidal or free boundary conditions are used in practice [14]. Thus, the neighbourhoods at the boundary nodes are modified according to the assumed conditions.

3.2.2 Sub-lattices

In practical computations it is often necessary to work over a *sub-lattice*, which is a subset of the original lattice. Define a 3-D **computational volume** ω_t of size $m \times n \times d$ (usually $m=n=2l+1$ and $d=2p+1$) centred around the node t:

$$\omega_t = \{t + (\Delta i, \Delta j, \Delta k)\}, -l \leq \Delta i \leq l, -l \leq \Delta j \leq l; -p \leq \Delta k \leq p \qquad (3.1)$$

within $\mathbf{\Lambda}$. There are n_T total number of nodes in ω_t. The lattice is divided into two subsets $\mathbf{\Lambda} = \mathbf{\Lambda}_I \cup \mathbf{\Lambda}_B$, where $\mathbf{\Lambda}_I$ is the set of all *interior* nodes (N_{TI} in total) which are used for the processing and $\mathbf{\Lambda}_B$ is the set of unprocessed *boundary* nodes. This subdivision is necessary because the computational volumes of the boundary nodes span over the image borders, thus some nodes are left unprocessed. Analogically, one can define a **computational window** ω'_t of size $n \times n$ in 2-D. The concept of a computational volume and 2-D cross-section of it (the computational window) is illustrated in Fig. 3.7.

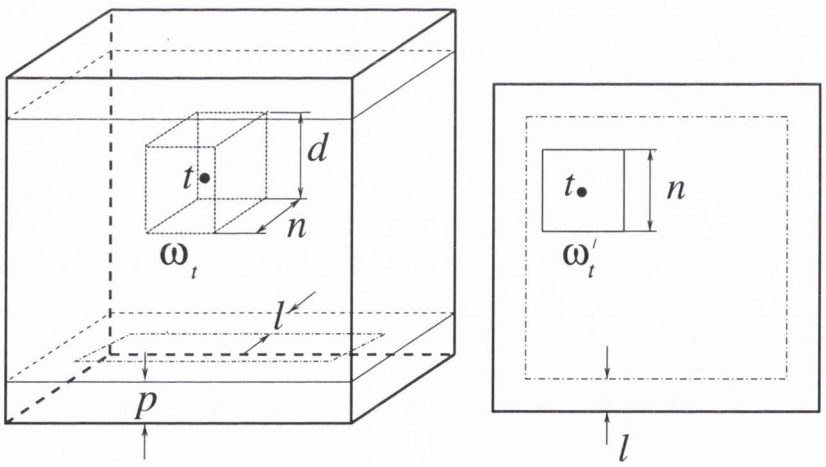

Figure 3.7: Computational volume ω_t and computational window ω'_t.

Other types of subsets of $\mathbf{\Lambda}$ used in this work are sub-sampled or **decimated attices**. Such will be denoted by a subscript to $\mathbf{\Lambda}$ related to the name of the lgorithm they were used for. These lattices constitute of a sub-sampled subset of odes and can be specified by the decimation rate parameters n^d in 2–D and d^d in the hird dimension. It means that only nodes with coordinates $(i+cn^d, j+cn^d, k+cd^d)$,

where $c \in \mathbb{Z}$ is a constant, are considered. This is illustrated in Fig. 3.8 where the nodes of the decimated grid with $n^d = 2$, $d^d = 4$ are denoted in black.

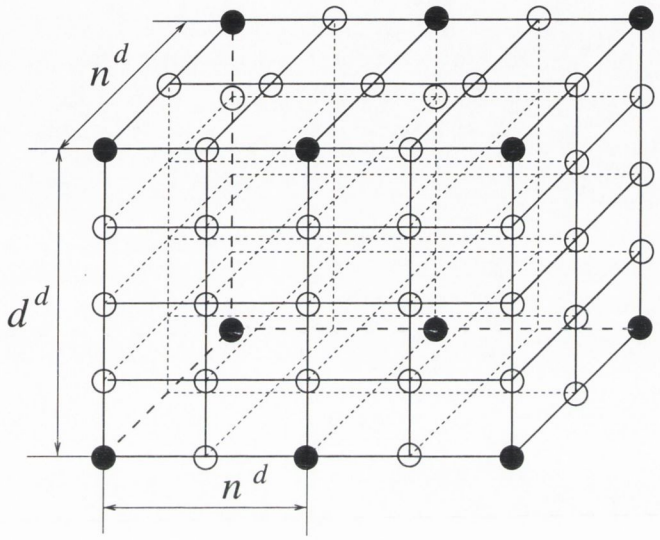

Figure 3.8: Decimated lattice with parameters $n^d = 2$ and $d^d = 4$.

3.2.3 Cliques

Definition 3.3 *A* **clique** C*, defined over a lattice* $\mathbf{\Lambda}$ *with respect to a neighbourhood system* $\boldsymbol{\eta}$ *is a subset of* $\mathbf{\Lambda}$ *($C \subseteq \mathbf{\Lambda}$) such that C is either a single node or a collection of nodes, all of which are neighbours.*

The set of all cliques in a neighbourhood is denoted by $\mathcal{C}$. Let $\mathcal{C}_n, n \in \mathbb{N}^+$ be the set of all cliques containing n nodes. The types of 2-D cliques occurring in first and second order neighbourhoods containing up to 4 nodes are illustrated in Fig. 3.9.

The cliques $\mathcal{C}_2$ containing only two nodes for first ($\eta(1)$), second ($\eta(2)$) and third ($\eta(3)$) order 3-D neighbourhoods are illustrated in Fig. 3.10. There are 13 such cliques indexed as $vi, i = 1 \ldots 9$ for the vertical and $hi, i = 1 \ldots 4$ for the horizontal ones.

The number of possible cliques increases significantly as the lattice dimensions and the neighbourhood order increase, so usually $\mathcal{C}$ is constrained to be a set of *pair-wise* cliques [26].

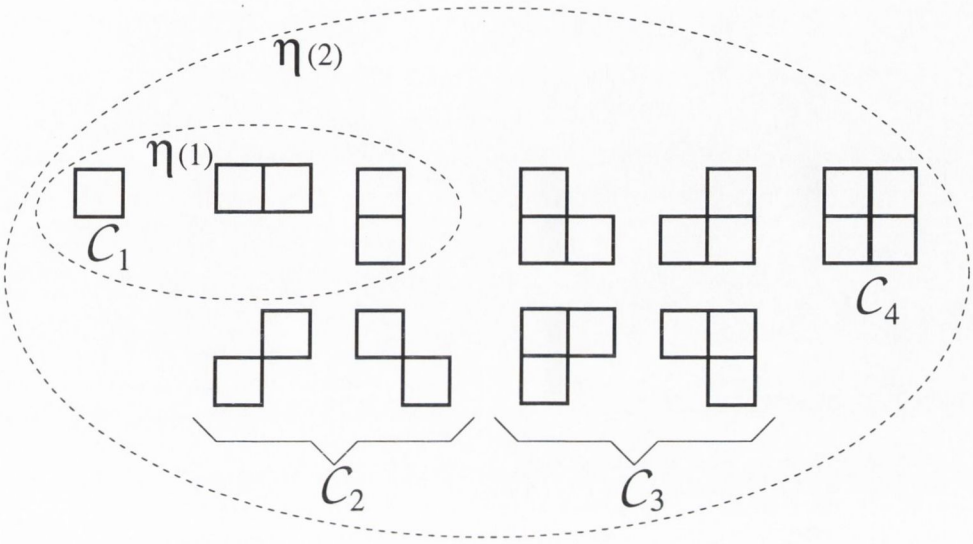

Figure 3.9: 2-D clique types for $\eta(1)$ and $\eta(2)$.

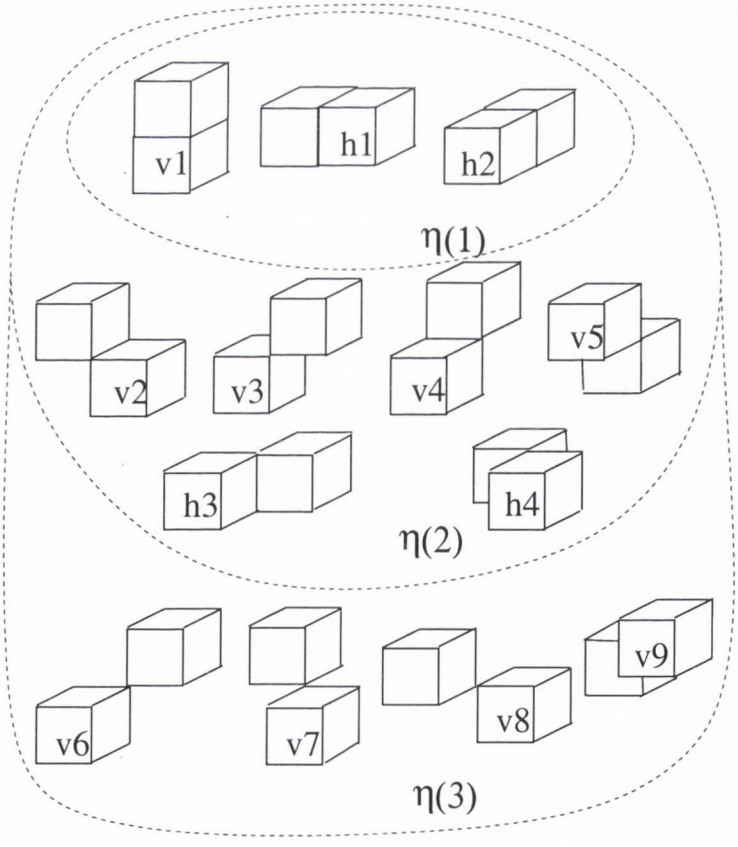

Figure 3.10: 3-D pair-wise cliques for $\eta(1)$, $\eta(2)$ and $\eta(3)$.

3.3 RANDOM FIELDS

Let $\mathbf{X} = \{X_1, \ldots, X_{N_T}\}$ be a family of random variables defined on the lattice $\mathbf{\Lambda}$, in which each random variable X_t takes a value x_t from the set Ξ. The family $\mathbf{X}$ is called a *random field.* The notation $X_t = x_t$ is used for the event that X_t takes the value x_t and the notation $\mathbf{X} = \mathbf{x}$ is used for the joint event, where $\mathbf{x} = \{x_1, \ldots, x_{N_T}\}$ is a *configuration* of $\mathbf{X}$, corresponding to a realization of the field.

In this thesis X_t is considered to be a random variable representing the grey level intensity at the image node t. Typically X_t takes values from the set $\Xi = \{0, \ldots, 2^B - 1\}$, i.e. the intensity has been encoded by B bits (usually $B = 8$). The vector $\mathbf{x} = \{x_t, 1 \leq t \leq |\mathbf{\Lambda}|\}$ consists of the realisations of the grey levels of all nodes in $\mathbf{\Lambda}$.

In this thesis discrete random fields defined over 2-D or 3-D lattices are considered which for brevity shall be called 2-D or 3-D random fields respectively.

3.3.1 Gibbs Random Fields

For a finite lattice $\mathbf{\Lambda}$ with a symmetric neighbourhood structure $\boldsymbol{\eta}$ a **Gibbs energy** function can be defined as:

$$U(\mathbf{x}) = \sum_{t \in \mathbf{\Lambda}, C \in \mathcal{C}_1} V_C(x_t) + \sum_{t \in \mathbf{\Lambda}} \sum_{s \in \eta_t, C \in \mathcal{C}_2} V_C(x_t, x_s), \tag{3.2}$$

where $V_C(x_t)$ and $V_C(x_t, x_s)$ are known as *clique potentials* such that $V_c(\cdot) \geq 0$. This ensures that the Gibbs energy (3.2) is non-negative definite. The clique potentials can be defined for cliques of arbitrary size $\mathcal{C}_n$, but in practice are limited to cliques containing up to two nodes- $\mathcal{C}_1$ and $\mathcal{C}_2$. To a Gibbs energy function of form (3.2) a probability distribution can be assigned:

$$P(\mathbf{X} = \mathbf{x}) = \frac{1}{Z} \exp\left(-\frac{1}{\mathcal{T}} U(\mathbf{x})\right), \tag{3.3}$$

where Z is a positive normalizing constant known as *partition function* and $\mathcal{T}$ is the "temperature" of the field. The temperature controls the sharpness of the distribution. If the temperature is high, all configurations tend to be equally distributed.

Near the zero temperature, the distribution concentrates around the global energy minima.

This family of distributions is known as **Gibbs distributions** and thus the random field $\mathbf{X}$ is known as a **Gibbs Random Field (GRF)**. A GRF is said to be *homogeneous* if the clique potential V_C is independent of the relative position of the clique C in $\mathbf{\Lambda}$. The homogeneity is assumed in most vision models for mathematical and computational convenience [53].

3.3.2 Markov Random Fields

Definition 3.4 *A random field* $\mathbf{X}$ *is called a* **Markov Random Field (MRF)** *on the lattice* $\mathbf{\Lambda}$ *with respect to the neighbourhood system* $\boldsymbol{\eta}$ *if it satisfies the following conditions:*

(i) Positivity

$P(\mathbf{X} = \mathbf{x}) > 0, \forall \mathbf{x} \in \Xi^{N_T}$, *and*

(ii) Markovianity

$p(X_t = x_t | X_v = x_v, \forall v \in \mathbf{\Lambda} \backslash \{t\}) = p(X_t = x_t | X_s = x_s, \forall s \in \eta_t)$.

According to this definition, all possible realizations $\mathbf{x}$ of the MRF should have nonzero joint probability $P(\mathbf{x})$, and the Markov property implies that the conditional probability density function (p.d.f.) at a particular node depends only on the values of the random field within the neighbourhood of this node.

When the positivity condition is met, the joint probability of any random field realization is uniquely determined by its local conditional probabilities [4]. The Markovianity describes the local characteristics of $\mathbf{X}$. A node interacts with only the neighbouring nodes, i.e. only neighbouring nodes have direct interactions with each other. It is always possible to select a sufficiently large η_t so that Markovianity holds and the largest neighbourhood consists of all sites. Any $\mathbf{X}$ is an MRF with respect to such a neighbourhood system.

The specification of an MRF via the local conditional probabilities has several

disadvantages [4]:

(i)There is no direct method for deducing the joint probability $P(\mathbf{x})$ from the conditional probabilities $p(x_t|x_s, s \in \eta_t)$.

(ii)The conditional probabilities themselves are subject to some highly restrictive consistency conditions.

(iii)The natural specification of the equilibrium of a statistical process is in terms of the joint probability rather than the conditional distribution of the variables.

However, MRFs can always be described by their *joint p.d.f.*, which is a Gibbs distribution (Section 3.3.1).

3.3.3 Hammersley-Clifford Theorem

An MRF is characterized by its local property (the Markovianity) whereas a GRF is characterized by its global property (the Gibbs distribution) [53]. The *Hammersley-Clifford* (H-C) *theorem* establishes the equivalence of these two types of properties. The theorem's relevance in the image processing community was established by Geman and Geman [31]. Its original proof was developed in 1971, but was published in 1990 [17].

Theorem 3.1 *The random field* $\mathbf{X}$ *is an MRF if and only if* $\mathbf{X}$ *is a GRF.*

The H-C theorem establishes the important connection between the local and global specifications of an MRF. It gives theoretical justification to the design of an MRF via local or global properties depending on their availability in a specific application context.

The importance of the theorem in practice is that it provides a simple way of specifying the joint probability. This can be done by defining the clique potential functions. Thus, any *a priori* knowledge or preference about the interactions between the sites can be embraced. How to choose the forms and parameters of the potential functions for an effective model is a major topic in MRF theory and practice.

3.4 STATIONARY GAUSSIAN MARKOV RANDOM FIELD MODEL

The Gibbs energy function (3.2) can be specified in different ways and this leads to different MRF models. An extensive presentation and comparison of different classes of MRFs models can be found in [58]. These include the *Auto-logistic, Strauss, Multi-level logistic, Auto-binomial, General-spin Ising, Auto-Poisson, Auto negative binomial*, and the widely used *Spatial autoregressive* models.

The choice of model is application-dependent. For instance, an auto-logistic model is appropriate for modelling binary textures; multi-level logistic (or "colour-blind") models are suitable not only for textures but also to model such categories as labels on a map [49]. The most widely used models are the Spatial autoregressive (AR) models. These appear in a variety of applications such as texture feature extraction, classification [13] and segmentation [57].

The subject of this work is the so called **auto-normal model**, also called the **Gaussian MRF (GMRF)** [14,53]. Fields defined over 2-D and mainly 3-D lattices are considered and they will be referred to as 2-D and 3-D GMRFs respectively.

The GMRF model is a special class of auto-models, for which only cliques containing up to two nodes, i.e. $C \in \mathcal{C}_1 \bigcup \mathcal{C}_2$ are used and the Gibbs energy is of form (3.2). For discrete problems the clique potentials can be specified by a number of *parameters.*

Defining the clique potentials in the Gibbs energy as

$$V_C(x_t) = \frac{(x_t - \mu_t)^2}{2\sigma^2}, \tag{3.4}$$

and

$$V_C(x_t, x_s) = -\theta_{t,s}\frac{(x_t - \mu_t)(x_s - \mu_s)}{\sigma^2}, s \in \eta_t, \tag{3.5}$$

where μ is a *mean* grey level value, σ is grey-level *variance* and $\theta_{t,s}$ is the *correlation* between the nodes t and s, leads to specifying the GMRF model.

The GMRF model is defined for a symmetric neighbourhood system $\boldsymbol{\eta}$ (i.e. $\theta_{t,s} = \theta_{s,t}$), for which additional constraint for a spatial symmetry has been imposed, namely $\theta_{t,s} = \theta_{t,-s}$, where by '$-s$' a node with the same, but 'negative' coordinate offsets from t as s is denoted. Let the coordinate offsets within η_t be δi, δj and δk. Therefore, if $s = t + (\delta i, \delta j, \delta k)$, then $-s = t - (\delta i, \delta j, \delta k) = t + (-\delta i, -\delta j, -\delta k)$.

Thus, there are n_q unique values of the correlation parameters within a neighbourhood consisting of $2n_q$ nodes. η_t is considered to consist of two symmetrical halves- a 'positive' and a 'negative': $\eta_t = \eta_t^+ \bigcup \eta_t^-$, i.e. if $s \in \eta_t^+, -s \in \eta_t^-$. This is illustrated for a second order ($q = 2$) 3-D GMRF with $n_q = 9$ in Figure 3.11.

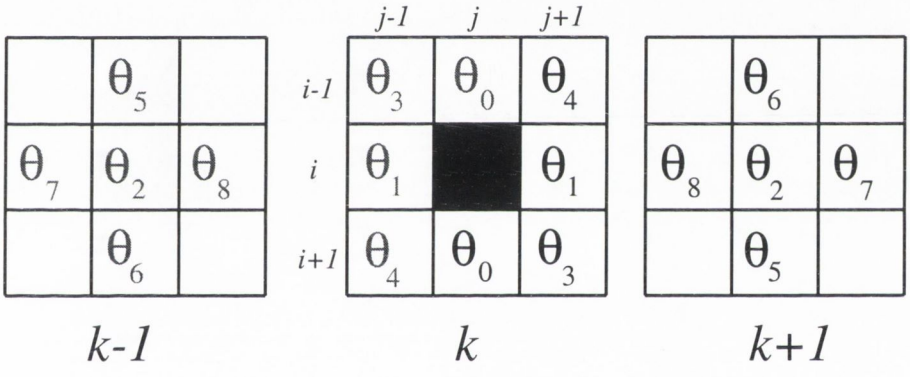

Figure 3.11: Parameters for second order 3-D GMRF.

In the figure the node t is denoted by a black square and θ_i is written inside each square corresponding to a neighbouring node $s \in \eta_t$, for which the correlation with t has the θ_i value and the parameters for the 'positive' half are denoted by black. Then, the correlation parameters vector using the adopted ordering can be written as:

$$\boldsymbol{\theta} = [\theta_0, \theta_1, \theta_2, \theta_3, \theta_4, \theta_5, \theta_6, \theta_7, \theta_8]^T \tag{3.6}$$

and the correspondence of each parameter with the cliques from Fig. 3.10 is: $\theta_0 \leftrightarrow h2$, $\theta_1 \leftrightarrow h1$, $\theta_2 \leftrightarrow v1$, $\theta_3 \leftrightarrow h4$, $\theta_4 \leftrightarrow h3$, $\theta_5 \leftrightarrow v5$, $\theta_6 \leftrightarrow v4$, $\theta_7 \leftrightarrow v3$ and $\theta_8 \leftrightarrow v2$.

For first order ($q = 1$) 3-D GMRF there are $n_q = 3$ correlation parameters. Then the elements of vector $\boldsymbol{\theta} = [\theta_0, \theta_1, \theta_2]^T$ are a subset of the parameters for the second order GMRF as illustrated in Fig. 3.12.

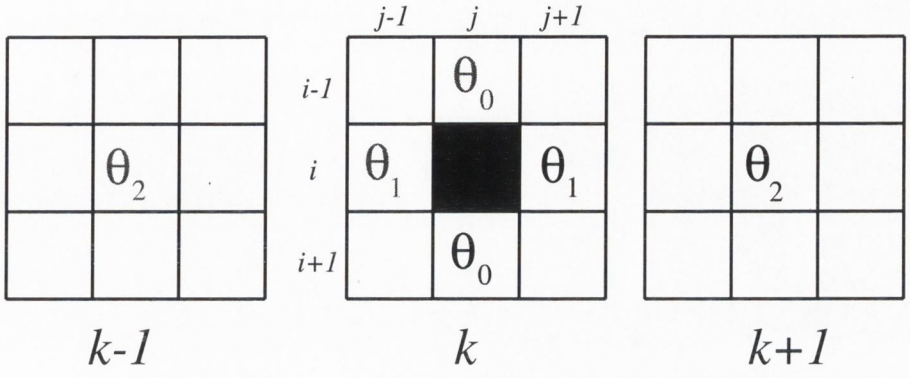

Figure 3.12: Parameters for first order 3-D GMRF.

For third order $q = 3$ 3-D GMRF there are $n_q = 13$ parameters and the vector can be written as:

$$\boldsymbol{\theta} = [\theta_0, \theta_1, \theta_2, \theta_3, \theta_4, \theta_5, \theta_6, \theta_7, \theta_8, \theta_9, \theta_{10}, \theta_{11}, \theta_{12}]^T \tag{3.7}$$

and the ordering within the neighbourhood structure is illustrated in Fig.3.13.

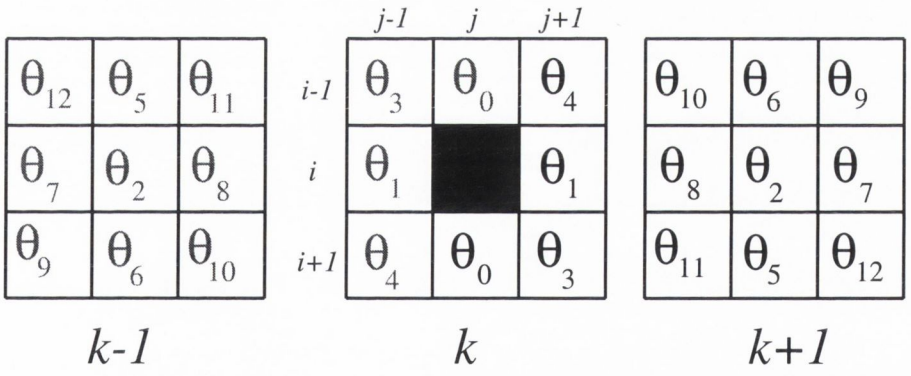

Figure 3.13: Parameters for third order 3-D GMRF.

The four extra parameters in comparison to the second order correspond to the following cliques from Fig. 3.10: $\theta_9 \leftrightarrow v6$, $\theta_{10} \leftrightarrow v7$, $\theta_{11} \leftrightarrow v9$ and $\theta_{12} \leftrightarrow v8$.

Thus, the GMRF is defined via a compact *parameter vector* $\mathbf{p} \in \mathbb{R}^{n_q+2}$. The parameter vector can be written as:

$$\mathbf{p} = [\mu, \sigma, \boldsymbol{\theta}^T]^T, \tag{3.8}$$

where $\boldsymbol{\theta} = [\theta_1, \ldots, \theta_{n_q}]^T$ is the vector of the ordered correlation parameters within the neighbourhood of order q, i.e. $n_q = |\boldsymbol{\theta}|$. Sometimes it is useful to refer only to the 'horizontal' subset of the correlation parameters in 2-D ($i - j$ plane) $\boldsymbol{\theta}_h = \{\theta_{t,s}, s \in \eta_t^+, s = t+(\cdot, \cdot, 0)\}$. Similarly, the subset of the 'vertical' correlation parameters along the third dimension can be denoted as $\boldsymbol{\theta}_v = \{\theta_{t,s}, s \in \eta_t^+, s = t + (\cdot, \cdot, k), k \neq 0\}$. For example for the second order 3-D GMRF

$$\boldsymbol{\theta}_h = [\theta_0, \theta_1, \theta_3, \theta_4]^T \tag{3.9}$$

and

$$\boldsymbol{\theta}_v = [\theta_2, \theta_5, \theta_6, \theta_7, \theta_8]^T \tag{3.10}$$

as follows from Fig. 3.11. From Fig. 3.13 for the third order 3-D GMRF the expression for $\boldsymbol{\theta}_h$ is the same as (3.9) and for $\boldsymbol{\theta}_v$ is:

$$\boldsymbol{\theta}_v = [\theta_2, \theta_5, \theta_6, \theta_7, \theta_8, \theta_9, \theta_{10}, \theta_{11}, \theta_{12}]^T. \tag{3.11}$$

3.4.1 Global Specification

Substituting the single clique potential functions (3.4) and the double clique potentials (3.5) in the Gibbs energy (3.2) it follows:

$$U(\mathbf{x}) = \frac{1}{2\sigma^2} \sum_{t \in \boldsymbol{\Lambda}} (x_t - \mu_t)^2 - \frac{1}{\sigma^2} \sum_{t \in \boldsymbol{\Lambda}} \sum_{s \in \eta_t} (x_t - \mu_t)\theta_{t,s}(x_s - \mu_s) \tag{3.12}$$

Thus, the joint probability of all N_T nodes from $\boldsymbol{\Lambda}$ is given by:

$$P(\mathbf{x}) = \frac{\sqrt{\det(\mathbf{B})}}{\sqrt{(2\pi\sigma^2)^{N_T}}} \exp\left[-\frac{(\mathbf{x} - \boldsymbol{\mu})^T \mathbf{B} (\mathbf{x} - \boldsymbol{\mu})}{2\sigma^2}\right], \tag{3.13}$$

where $\boldsymbol{\mu}$ is an $N_T \times 1$ vector of conditional means, and $\mathbf{B} = [b_{st}]$ is the $N_T \times N_T$ *interaction matrix* with elements given by:

$$b_{st} = \begin{cases} 1, & \text{if } s = t, \\ -\theta_{s,t}, & \text{if } s \in \eta_t, \\ 0, & \text{otherwise.} \end{cases} \tag{3.14}$$

The probability (3.13) is multivariate normal and $\mathbf{B}$ is related to the covariance matrix by $\mathbf{\Sigma}^{-1} = \mathbf{B}/2\sigma^2$. Thus, the multivariate Gaussian distribution is a special case of the Gibbs distribution family (3.3). The necessary and sufficient condition for (3.13) to be a valid p.d.f. is that $\mathbf{B}$ be symmetric and positive-definite [53]. Such a condition (extended for 3-D GMRFs), dictates that the valid parameter subspace be $\sum |\boldsymbol{\theta}| < 0.5$ [50].

3.4.2 Local Specification

The expression in the exponent of (3.13) can be rewritten as

$$U = -\frac{1}{2\sigma^2} \sum_{t \in \mathbf{\Lambda}, s \in \eta_t} (x_s - \mu_s) b_{st} (x_t - \mu_t). \tag{3.15}$$

By fixing t and completing the square it follows:

$$\begin{aligned} U &= -\frac{1}{2\sigma^2}(x_t - \mu_t)^2 + \frac{1}{\sigma^2} \sum_{s \in \eta_t} (x_t - \mu_t)\theta_{t,s}(x_s - \mu_s) + \dots \\ &= -\frac{1}{2\sigma^2} \left(x_t - \mu_t - \sum_{s \in \eta_t} \theta_{t,s}(x_s - \mu_s) \right)^2 + \dots, \end{aligned} \tag{3.16}$$

where ... stands for terms that do not depend on x_t. This expression was obtained using the assumption of symmetry $\theta_{t,s} = \theta_{s,t}$ and implies the following local conditional probability:

$$p(x_t | x_s, s \in \eta_t) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp \left[-\frac{1}{2\sigma^2} \left(x_t - \mu_t - \sum_{s \in \eta_t} \theta_{t,s}(x_s - \mu_s) \right)^2 \right] \tag{3.17}$$

It is a Gaussian univariate normal distribution with conditional mean

$$E(x_t | x_s, s \in \eta_t) = \mu_t + \sum_{s \in \eta_t} \theta_{t,s}(x_s - \mu_s) \tag{3.18}$$

and conditional variance

$$var(x_t | x_s, s \in \eta_t) = \sigma^2 \tag{3.19}$$

Thus the model defined via N_T conditional probabilities (3.17) is the model defined via the joint distribution (3.13) (satisfying the H-C theorem). The grey

level of node t obeying (3.17) can be expressed as a linear combination of the grey levels at the neighbourhood nodes:

$$x_t = \mu_t + \sum_{s \in \eta_t} \theta_{t,s}(x_s - \mu_s) + e_t, \tag{3.20}$$

where the correlated Gaussian noise has the following structure:

$$E[e_t e_s] = \begin{cases} \sigma^2, & t = s, \\ -\theta_{t,s}\sigma^2, & s \in \eta_t, \\ 0, & \text{otherwise.} \end{cases} \tag{3.21}$$

This leads to a model specified by N_T equations (3.20) with joint p.d.f. which can be written in form (3.13), imposing weaker conditions on the $\mathbf{B}$ matrix [53]. Thus, under the condition $\sum |\boldsymbol{\theta}| < 0.5$ the models (3.17) and (3.20) are equivalent and are specifying a stationary GMRF with joint probability of form (3.13).

3.5 THE LABELLING PROBLEM

Many image processing and computer vision tasks can be posed as labelling problems. In this framework the solution to a problem is a set of *labels* assigned to image data nodes. In this thesis the task of labelling data, which is a mixture of stationary GMRFs is considered. The original grey level data are seen as existing on the lattice $\mathbf{\Lambda}$ together with a hidden *label field.* The image data are assumed to be generated by T underlying image models (stationary GMRFs) and the task is to assign to each node the corresponding label, which is an indicator of suitability of the image model.

3.5.1 Label Field

In this work a **label field** $\mathbf{L} = \{1, \ldots, T\}^{N_T}$ is considered to be a discrete set of labels over the lattice. A **label** $l \in \mathbf{L}$ is an event which may happen to a site. The

notation l_t means that the label corresponds to a site t. The labelling problem is to assign the set of labels $\mathbf{L}$ to each of the nodes in $\mathbf{\Lambda}$. The labelling is a mapping from $\mathbf{\Lambda}$ to $\mathbf{L}$. The problem of region labelling can be considered of type mapping "regular sites with discrete labels" [53]. This is illustrated on Fig. 3.14. The task is to partition the image data into mutually exclusive regions, each of which has some uniform and homogeneous property with significantly different value from those of the neighbouring regions.

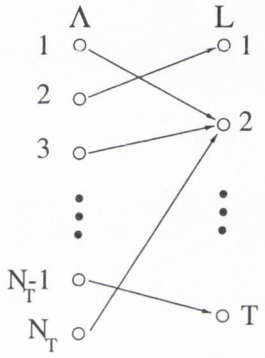

Figure 3.14: Mapping with discrete label set.

In this thesis, the emphasis is put on the texture property. Then nodes, within the same texture region are assigned the same unique label.

3.5.2 Codebook

The labelling can be considered as the inference of $\mathbf{L}$ given the observed data $\mathbf{X}$. This inference can be done in a *supervised* or *unsupervised* manner. If the number of underlying stationary GMRF models T is known in advance along with their parameter vectors $\mathbf{p}$, the task is considered supervised, otherwise it is unsupervised. In this work the supervised approach is used.

In a supervised approach knowledge of T and the representative parameter estimates $\mathbf{c}_i = \widehat{\mathbf{p}}_i, i = 1, \ldots, T$ for each of the stationary models is available *a priori*. The parameter estimates become entries in a **codebook** $\mathbf{C} = \{\mathbf{c}_1, \mathbf{c}_2, \ldots, \mathbf{c}_T\}$. Thus, the inference of the hidden label field $\mathbf{L}$ is performed given the observed data X and the codebook $\mathbf{C}$.

3.6 CONCLUSIONS

This chapter presented a model for images existing on 3-D lattices, namely the 3-D GMRF. The theoretical framework for 3-D MRF has been given, introducing concepts and notations used through the thesis such as lattice, neighbourhood, cliques and clique potentials. The notions of computational volume and decimated lattice have been formally defined.

The 3-D stationary GMRF model has been considered in detail and it was shown how it can represent the 3-D image data via a compact parameter vector $\mathbf{p}$. The global and local specifications of the model were given and the important interaction matrix $\mathbf{B}$ has been introduced.

Finally, the label field and the codebook, as major notions used in a supervised labelling problem, were defined.

Thus, the main task of this thesis can be formulated as region labelling of a mixture of stationary GMRFs on 3-D lattices in a supervised manner. To solve this task it is first important to be able to synthesize and analyze 3-D stationary GMRFs, which are the subjects of the following chapters.

Chapter 4

SYNTHESIS OF A STATIONARY GMRF

4.1 INTRODUCTION

In applications such as image synthesis, coding and compression, it is very important to be able to generate GMRF models. For simulation purposes in the current work it will be necessary to synthesize GMRFs with desired parameters.

When modelling images on finite lattices, boundary conditions must be adopted as explained in Chapter 3. It is argued that assuming *toroidal* conditions leads to computationally efficient algorithms and thus are used widely [14]. If such conditions are applied along each lattice dimension, the interaction matrix in the joint probability of a GMRF model (Section 3.4.1) becomes a matrix with a special structure. For a L-D lattice the structure is known as *block-circulant of level L*. The theory of circulant matrices of levels one and two has been developed extensively in the literature [22, 40, 65], but this is not the case with circulants of higher level. This extension is important in volumetric imagery processing. For this reason it is addressed in this chapter.

Firstly, the theory of the block-circulant matrices of level three is developed

in a consistent way in Section 4.2, followed by presentation of a main theoretical result, namely the diagonalization theorem in Section 4.3. It is used as a basis of several efficient algorithms for calculations with circulant matrices of level three. The result is the mathematical foundation for the development of a novel 3-D GMRF synthesis algorithm, presented in Section 4.4. What follows is a simulation example illustrating all theoretical developments in Section 4.5. Finally, the main conclusions are summarized in Section 4.6.

4.2 BLOCK CIRCULANT MATRICES OF LEVEL THREE

4.2.1 Definitions

Definition 4.1 *A* **circulant matrix** *(or* **circulant** *for short) of order m is a square matrix C having the following structure:*

$$\mathsf{C} = \begin{pmatrix} \mathsf{c}_1 & \mathsf{c}_2 & \mathsf{c}_3 & \dots & \mathsf{c}_m \\ \mathsf{c}_m & \mathsf{c}_1 & \mathsf{c}_2 & \dots & \mathsf{c}_{m-1} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \mathsf{c}_2 & \mathsf{c}_3 & \mathsf{c}_4 & \dots & \mathsf{c}_1 \end{pmatrix}. \tag{4.1}$$

A circulant has at most m different entries and is determined completely by its first row $\mathbf{c} = (\mathsf{c}_1, \mathsf{c}_2, \mathsf{c}_3, \dots, \mathsf{c}_m)$: $\mathsf{C} = \mathrm{circ}(\mathsf{c}_1, \mathsf{c}_2, \mathsf{c}_3, \dots, \mathsf{c}_m) = \mathrm{circ}(\mathbf{c})$. The class of all circulant matrices of order m is denoted by $\mathcal{C}(m)$, i.e. $\mathsf{C} \in \mathcal{C}(m)$. Circulant matrices are special case of Toeplitz matrices.

The structure of a circulant matrix is obtained by using a *forward shift permutation* [22]. This operation can be expressed in a matrix form by the *permutation matrix* of order m defined as:

$$\boldsymbol{\pi}_m = circ(\underbrace{0, 1, 0, \dots, 0}_{m}) \tag{4.2}$$

Then, any circulant matrix of order m can be represented as:

$$\mathsf{C} = \mathsf{c}_1\mathbf{I}_m + \mathsf{c}_2\boldsymbol{\pi}_m + \ldots + \mathsf{c}_m\boldsymbol{\pi}_m^{m-1} = \sum_{i=1}^{m} \mathsf{c}_k\boldsymbol{\pi}_m^{(k-1)}, \tag{4.3}$$

where $\mathbf{I}_m$ is the identity matrix of order m and $\boldsymbol{\pi}_m^k$ is the matrix $\boldsymbol{\pi}_m$ on the power of k. It can be easily seen that $\mathbf{I}_m = \boldsymbol{\pi}_m^0$.

A **block-circulant matrix** (or **block-circulant** for short) consists of circulant blocks, but the matrix it is not necessarily a circulant nor Toeplitz itself. However, for GMRF modelling, a specialization of block-circulants is considered. These are *the block circulant matrices with circulant blocks.* Denote the class of all such matrices by $\mathcal{C}(m,n)$, i.e. they are constructed from n different blocks each being a circulant or order m. Then, the matrix $\mathbf{C} \in \mathcal{C}(m,n)$ is of size $mn \times mn$ and has the following structure:

$$\mathbf{C} = \begin{pmatrix} \mathsf{C}_1 & \mathsf{C}_2 & \mathsf{C}_3 & \ldots & \mathsf{C}_n \\ \mathsf{C}_n & \mathsf{C}_1 & \mathsf{C}_2 & \ldots & \mathsf{C}_{n-1} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \mathsf{C}_2 & \mathsf{C}_3 & \mathsf{C}_4 & \ldots & \mathsf{C}_1 \end{pmatrix}. \tag{4.4}$$

The notation is $\mathbf{C} = \mathrm{bcirc}(\mathsf{C}_1, \mathsf{C}_2, \ldots, \mathsf{C}_n)$, where the b in bcirc can be omitted when the block structure is clear.

The level at which the block-circularity occurs can increase. Thus, going to the third level, there is a matrix which is block-circulant and each block is itself of class $\mathcal{C}(m,n)$.

A square matrix of order mnp (and of size $mnp \times mnp$) is of class $\mathcal{C}(m,n,p)$ if it can be divided into p different blocks of order mn with block-circulant structure. Each of the p blocks can be subdivided into n different blocks, each of which is a circulant of order m. Such matrix is known as *block-circulant matrix of level three.* To summarize:

- A circulant of level 1 is an ordinary circulant $\mathsf{C} \in \mathcal{C}(m)$.
- A circulant of level 2 is a block-circulant $\mathbf{C}$ of class $\mathcal{C}(m,n)$.

• A circulant of level 3 is a block-circulant whose blocks are level 2 circulants, $\mathbf{C} \in \mathcal{C}(m, n, p)$.

• In general, a circulant of level $l \geq 2$ is a block-circulant whose blocks are circulants of level $l - 1$.

4.2.2 Storage of Block Circulant Matrices

Block circulant matrices with circulant blocks of any level (as well as circulant matrices themselves) have the important property that all the information is contained in the first row (column) and so this, alone, needs to be stored.

To illustrate the concept, consider a circulant matrix $\mathbf{C}$ of level 3 with $m = 4$, $n = 3$ and $p = 2$, i.e. $\mathbf{C} \in \mathcal{C}(4, 3, 2)$. The structure of the matrix of size 24×24 is presented in Fig. 4.1. Clearly, the matrix is fully specified by the entries of the first row.

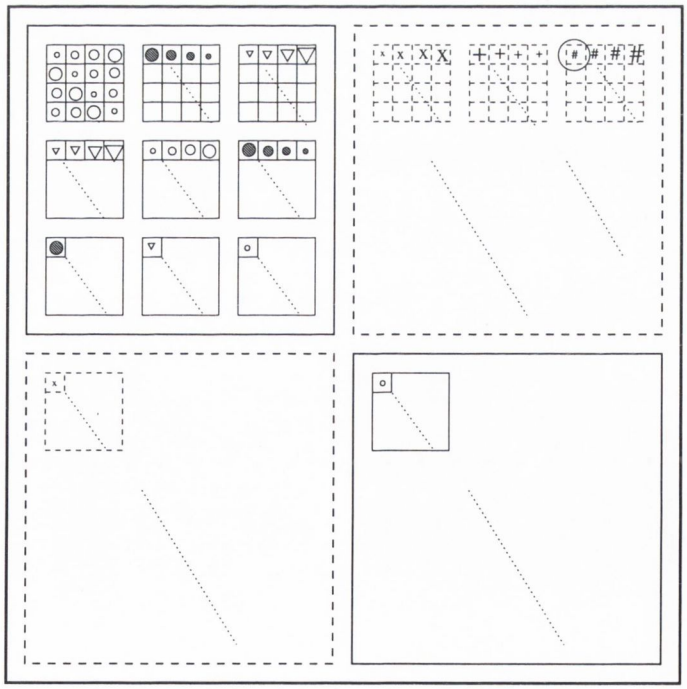

Figure 4.1: Structure of a block-circulant of level three.

These entries can be arranged in a 3-D *generating tensor* $\mathbf{T}$ of size $m \times n \times p$ via:

$$\mathbf{T}(i,j,k) = \mathbf{C}_k(1, i + (j-1)m) = \mathbf{C}(1, (i + (j-1)m) + (k-1)mn), \qquad (4.5)$$

where $i = 1, \ldots, m$, $j = 1, \ldots, n$, $k = 1, \ldots, p$ and $\mathbf{C}_k$ are the circulant of level 2 blocks. $\mathbf{T}$ simply reads the matrix subrows into a tensor, compactly representing $(mnp)^2$ elements by the sufficient subset of only mnp elements. The tensor for the matrix from Fig. 4.1 is of size $4 \times 3 \times 2$ and is given in Fig. 4.2.

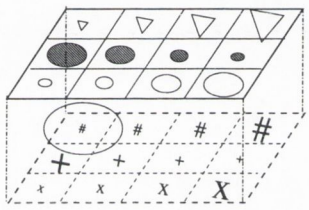

Figure 4.2: Generating tensor for a block-circulant of level three.

The encircled element in both figures, with entry the smallest '#', can be indexed via (4.5): $\mathbf{T}(1,3,2)$=$\mathbf{C}_2(1,9)$=$\mathbf{C}(1,21)$.

4.3 DIAGONALIZATION OF CIRCULANTS OF LEVEL THREE

4.3.1 Three-Dimensional Discrete Fourier Transform

Definition 4.2 *Let* $\omega = \exp(-\mathbf{j}\frac{2\pi}{m}), \mathbf{j} = \sqrt{-1}$ *be the m-th root of unity,* $m \in \mathbb{Z}$. *Then the* **Fourier matrix** *of order m (of size* $m \times m$*) is defined as:*

$$\mathbf{F} = \frac{1}{\sqrt{m}} \begin{pmatrix} 1 & 1 & 1 & \ldots & 1 \\ 1 & \omega & \omega^2 & \ldots & \omega^{m-1} \\ 1 & \omega^2 & \omega^4 & \ldots & \omega^{2(m-1)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & \omega^{m-1} & \omega^{2(m-1)} & \ldots & \omega^{(m-1)(m-1)} \end{pmatrix} \qquad (4.6)$$

The matrix of order m of this kind is denoted by $\mathbf{F}_m$. The Fourier matrix is unitary, i.e. $\mathbf{F}_m\mathbf{F}_m^* = \mathbf{F}_m^*\mathbf{F}_m = \mathbf{I}_m$, thus $\mathbf{F}_m^{-1} = \mathbf{F}_m^*$. The star operation denotes conjugate transposition operation. Another square matrix of order m is defined as well:

$$\mathbf{\Omega}_m = \text{diag}(1, \omega, \omega^2, \ldots, \omega^{m-1}). \tag{4.7}$$

The Fourier matrix is linked directly to the Discrete Fourier Transform (DFT).

Definition 4.3 *Let* $\mathbf{x} = (x_1, x_2, \ldots, x_m)^T$ *and* $\mathbf{y} = (y_1, y_2, \ldots, y_m)^T$ *be complex vectors. The linear transformation:*

$$\mathbf{y} = \mathbf{F}\mathbf{x} \tag{4.8}$$

where $\mathbf{F}$ *is the Fourier matrix is known as the* **Discrete Fourier Transform (DFT)** *of* $\mathbf{x}$*. Its inverse is given by:*

$$\mathbf{x} = \mathbf{F}^{-1}\mathbf{y} = \mathbf{F}^*\mathbf{y}. \tag{4.9}$$

The DFT can be defined in higher dimensions. The 3-D DFT used in this thesis is given by the following definition [61]:

Definition 4.4 *The* **3-D DFT** *of a 3-D tensor* $\mathbf{X}$ *of size* $m \times n \times p$ *is a tensor* $\mathbf{Y}$ *of the same size with entries:*

$$\begin{aligned}\mathbf{Y}(u,v,w) =& \frac{1}{\sqrt{mnp}} \sum_{i=1}^{m}\sum_{j=1}^{n}\sum_{k=1}^{p} \mathbf{X}(i,j,k) \\ & \exp\left[-2\pi\mathbf{j}\left(\frac{(u-1)(i-1)}{m} + \frac{(v-1)(j-1)}{n} + \frac{(w-1)(k-1)}{p}\right)\right], \\ & u = 1, \ldots, m,\ v = 1, \ldots, n,\ w = 1, \ldots, p\end{aligned} \tag{4.10}$$

If any two dimensions are 1's, this reduces to the one-dimensional (1-D) DFT(4.8). With one dimension equal to 1, it is equivalent to the 2-D DFT. The 3-D DFT operation can be denoted for short by: $\mathbf{Y} = DFT(\mathbf{X})$. In the volumetric processing context, the tensor $\mathbf{X}$ is usually the image data array.

Definition 4.5 *The* **inverse 3-D DFT (IDFT)** *of a 3-D tensor* $\mathbf{Y}$ *of size* $m \times n \times p$ *is a tensor* $\mathbf{X}$ *of the same size with entries:*

$$\begin{aligned}\mathbf{X}(i,j,k) = &\frac{1}{\sqrt{mnp}} \sum_{u=1}^{m} \sum_{v=1}^{n} \sum_{w=1}^{p} \mathbf{Y}(u,v,w) \\ &\exp\left[2\pi \mathbf{j} \left(\frac{(u-1)(i-1)}{m} + \frac{(v-1)(j-1)}{n} + \frac{(w-1)(k-1)}{p} \right) \right], \\ &i = 1, \ldots, m, \; j = 1, \ldots, n, \; k = 1, \ldots, p\end{aligned} \tag{4.11}$$

It can be denoted by $\mathbf{X} = IDFT(\mathbf{Y})$ for short.

The 3-D DFT can also be written as a matrix multiplication like the 1-D DFT (Definition 4.3). The following matrix of size $mnp \times mnp$ can be defined:

$$\boldsymbol{\mathcal{F}} = \mathbf{F}_m \otimes \mathbf{F}_n \otimes \mathbf{F}_p, \tag{4.12}$$

where $\otimes$ denotes Kronecker product.

Then, instead of applying (4.10), $\mathbf{Y}$ can be obtained from $\mathbf{X}$ by using the $\boldsymbol{\mathcal{F}}$ matrix, which is the matrix of the 3-D DFT as follows:

1. `Compose vector x of size` $mnp \times 1$ `by reading sequentially the rows of` $\mathbf{X}$.
2. `Compute` $\mathbf{y} = \boldsymbol{\mathcal{F}}\mathbf{x}$.
3. `Compose` $\mathbf{Y}$ `from y by the inverse operation of step 1.`

The matrix $\boldsymbol{\mathcal{F}}^H$ is the matrix for the IDFT.

The DFT can be implemented in a computationally efficient manner. In the 1-D case, the Cooley-Tukey [19] Fast Fourier Transform (FFT) algorithm requires at most $N \log_2 N$ complex multiplications to transform a sequence of a radix-2 length. The equivalent matrix multiplication would require N^2 complex multiplications.

The 2-D and 3-D DFTs can be seen as a sequence of 1-D transforms applied to each dimension. The total number of operations required for the 3-D transform of an $N_1 \times N_2 \times N_3$ array is $N_1 N_2 N_3 \log_2(N_1 N_2 N_3)$ compared with $(N_1 N_2 N_3)^2$ operations involved in the multiplication of a $N_1 N_2 N_3 \times N_1 N_2 N_3$ matrix by a vector of length $N_1 N_2 N_2$.

In order to reduce the number of complex multiplications required, the 3-D FFT is used to perform the multiplications of the form $\boldsymbol{\mathcal{F}}\mathbf{x}$. Then, the above algorithm will be denoted by $\mathbf{Y} = FFT(\mathbf{X})$ for brevity.

4.3.2 Diagonalization Theorem

It is known that any circulant matrix of level 1 or 2 can be diagonalized by the 1-D and 2-D DFTs respectively [22, 40]. The result can be extended as follows:

Theorem 4.1 *For a block circulant matrix of level 3* $\mathbf{C} \in \mathcal{C}(m, n, p)$*, the following representation applies:*

$$\mathbf{C} = \boldsymbol{\mathcal{F}}^H \boldsymbol{\Lambda} \boldsymbol{\mathcal{F}} \tag{4.13}$$

where $\boldsymbol{\mathcal{F}}$ *is the matrix defined via (4.12) and* $\boldsymbol{\Lambda}$ *is the diagonal matrix of the eigenvalues of* $\mathbf{C}$*.*

The derivation of (4.13) is given in Appendix A.

4.3.3 Computations with Circulants of Level Three

The diagonalization Theorem 4.1 can be used for computing the eigenvalues of $\mathbf{C}$. The eigenvalues are used for many operations involving level 3 circulants- determinant evaluation, inverse, matrix-vector product, and quadratic forms.

Computing the Determinant

From (4.13) it follows that $\boldsymbol{\mathcal{F}}\mathbf{C} = \boldsymbol{\Lambda}\boldsymbol{\mathcal{F}}$ and only the first rows of each side of the expression are considered (using MATLAB notation): $(\boldsymbol{\mathcal{F}}\mathbf{C})(1,:) = \boldsymbol{\lambda}\boldsymbol{\mathcal{F}}(1,:)$, where $\boldsymbol{\lambda}$ is the vector of the eigenvalues of $\mathbf{C}$. From (4.12) and (4.6) it follows that:

$$\boldsymbol{\mathcal{F}}(1,:) = \frac{1}{\sqrt{mnp}} \underbrace{(1, 1, \ldots, 1)}_{mnp}. \tag{4.14}$$

Therefore,

$$\boldsymbol{\lambda} = (\boldsymbol{\mathcal{F}}\mathbf{C})(1,:)\sqrt{mnp}. \tag{4.15}$$

As explained in Section 4.3.1 the multiplication of a vector, representing a 3-D tensor, by the matrix $\boldsymbol{\mathcal{F}}$ is equivalent to taking a 3-D DFT of the tensor. Hence, $(\boldsymbol{\mathcal{F}}\mathbf{C})(1,:)$ is equivalent to performing a 3-D DFT of the generating tensor of $\mathbf{C}$, which is $\mathbf{T}$ defined by (4.5).

This leads to a fast method for computing the eigenvalues of $\mathbf{C}$ using the 3-D FFT:

1. Construct the generating tensor $\mathbf{T}$ of the $\mathbf{C}$ matrix by (4.5).
2. Take the 3-D FFT (Section 4.3.1) of $\mathbf{T}$.
3. Multiply the result by $\sqrt{mnp}$.
4. Read off the result row-wise into a vector $\boldsymbol{\lambda} = [\lambda_i]$ of length mnp.

Then the determinant of $\mathbf{C}$ is simply given by:

$$\det(\mathbf{C}) = \prod_{i=1}^{mnp} \lambda_i. \tag{4.16}$$

Quadratic Form

For any $mnp \times 1$ vector $\mathbf{x}$ one may wish to compute the quadratic form $\mathbf{x}^H\mathbf{C}\mathbf{x}$. From Theorem 4.1 it follows:

$$\begin{aligned} \mathbf{x}^H\mathbf{C}\mathbf{x} &= \mathbf{x}^H\boldsymbol{\mathcal{F}}^H\boldsymbol{\Lambda}\boldsymbol{\mathcal{F}}\mathbf{x} = (\boldsymbol{\mathcal{F}}\mathbf{x})^H\boldsymbol{\Lambda}(\boldsymbol{\mathcal{F}}\mathbf{x}) \\ &= \mathbf{y}^H\boldsymbol{\Lambda}\mathbf{y} = \sum_{i=1}^{mnp} \lambda_i |y_i|^2, \end{aligned} \tag{4.17}$$

where the $\boldsymbol{\Lambda}$ entries are computed by (4.15).

Therefore, the algorithm is as follows:

1. Compute the eigenvalues $\boldsymbol{\lambda}$ by (4.15).
2. Create $m \times n \times p$ tensor $\mathbf{X}$ from the vector $\mathbf{x}$.
3. Take the 3-D FFT of $\mathbf{X}$ and call the result $\mathbf{Y}$.
4. Compose the complex vector $\mathbf{y} = [y_i]^H, i = 1, \ldots, mnp$ by reading off $\mathbf{Y}$ row-wise.
5. Compute the quadratic form by (4.17).

4.4 SYNTHESIS OF A 3-D GMRF

There are many algorithms for generation of 2-D GMRFs with properties that are *similar* to a chosen image. Both deterministic and probabilistic sampling methods can be used. There is a very computationally effective algorithm, proposed by Chellappa [14], for synthesis of finite-lattice 2-D GMRFs. This algorithm exploits the properties of the circulant matrices of level 2. An extension of this approach, yielding an algorithm for generation of 3-D GMRFs is proposed here [70]. It is based on the computational properties of the symmetric block-circulant matrices of level 3.

4.4.1 The Structure of the GMRF Interaction Matrix

As explained in Section 3.2.1 when working with finite lattices, assumptions about the neighbourhood geometry of the nodes from the lattice boundaries must be adopted. Chellappa argues in [14] that the adoption of *toroidal boundary conditions* leads to a special structure of the GMRF interaction matrix (Section 3.4.1).

Toroidal boundary assumption means that the neighbours of a site from the lattice boundaries are chosen among nodes which would be spatial neighbours if the lattice had been wrapped in a toroidal manner. Formally, consider the lattice from Fig. 3.1 and node t located at some of the lattice boundaries. Then, for any neighbour $s \in \eta_t^+$ of a site t, some of the following inequalities apply: $1 > i + \delta i$, $i+\delta i > M$, $1 > j+\delta j$, $j+\delta j > N$, $1 > k+\delta k$, $k+\delta k > D$. Then, the corresponding coordinate offsets of $s = t + (\delta i, \delta j, \delta k)$, for which some of the above hold, have to be updated according to the modulo counting valid for the toroidal geometry:

$$\begin{aligned} i + \tilde{\delta i} &= M + (i + \delta i), i + \tilde{\delta i} = (i + \delta i) \mathrm{mod} M, \\ j + \tilde{\delta j} &= N + (i + \delta i), j + \tilde{\delta j} = (j + \delta j) \mathrm{mod} N, \\ k + \tilde{\delta k} &= D + (k + \delta k), k + \tilde{\delta k} = (k + \delta k) \mathrm{mod} D. \end{aligned} \tag{4.18}$$

Analogically, the expressions for nodes $-s \in \eta_t^-$ can be derived.

To illustrate the toroidal boundary assumption, consider a second order ($q = 2$) 3-D GMRF and the location of the neighbours of an internal lattice node according to the adopted ordering of the correlation parameters (see Figure 3.11). This is summarized in Table 4.1.

Table 4.1: Second order neighbourhood of an internal node $t = (i, j, k)$.

θ_0	θ_1	θ_2	θ_3	θ_4	θ_5	θ_6	θ_7	θ_8
$(i-1,j,k)$	$(i,j-1,k)$	$(i,j,k-1)$	$(i-1,j-1,k)$	$(i+1,j-1,k)$	$(i-1,j,k-1)$	$(i+1,j,k-1)$	$(i,j-1,k-1)$	$(i,j+1,k-1)$
$(i+1,j,k)$	$(i,j+1,k)$	$(i,j,k+1)$	$(i+1,j+1,k)$	$(i-1,j+1,k)$	$(i+1,j,k+1)$	$(i-1,j,k+1)$	$(i,j+1,k+1)$	$(i,j-1,k+1)$

If a lattice boundary node is considered, for instance the first node of the data at location $(1, 1, 1)$, the neighbours with the toroidal assumption imposed are defined as given in Table 4.2.

Table 4.2: Second order neighbourhood of the node $t = (1, 1, 1)$.

θ_0	θ_1	θ_2	θ_3	θ_4	θ_5	θ_6	θ_7	θ_8
$(M,1,1)$	$(1,N,1)$	$(1,1,D)$	$(M,N,1)$	$(2,N,1)$	$(M,1,D)$	$(2,1,D)$	$(1,N,D)$	$(1,2,D)$
$(2,1,1)$	$(1,2,1)$	$(1,1,2)$	$(2,2,1)$	$(M,2,1)$	$(2,1,2)$	$(M,1,2)$	$(1,2,2)$	$(1,N,2)$

Each entry b_{st} of the interaction matrix $\mathbf{B}$ of the GMRF (Section 3.4.1) corresponds to the interaction between nodes t and s. The matrix has been obtained by fixing $t = (1, 1, 1)$ and building the entries by (3.14) using the non-zero elements of $\boldsymbol{\theta}$ [26]. The matrix $\mathbf{B} = \mathbf{B}(\boldsymbol{\theta})$ is dependent on the GMRF correlation parameter vector defined in Section 3.4. Therefore, the only non-zero entries of the matrix are the entries within η_t. Hence, the Markovianity property (Definition 3.4,(ii)) of the GMRF implies *sparsity* of the interaction matrix. The spatial symmetry of the neighbourhood geometry implies symmetry of $\mathbf{B}$. And, finally, the toroidal boundary assumption along each lattice dimension implies block-circulant (of level three) structure of the matrix.

For example, a second order 3-D GMRF, has interaction matrix given by:

$$\mathbf{B} = \text{bcirc}(\mathbf{B}_1, \mathbf{B}_2, \ldots, \mathbf{B}_D), \tag{4.19}$$

where each $MN \times MN$ block $\mathbf{B}_k$ is a block-circulant matrix of level 2, representing the interaction between the first and k-th slice of the 3-D data. Each $\mathbf{B}_k = \text{bcirc}(\mathsf{B}_{1,k}, \mathsf{B}_{2,k}, \ldots, \mathsf{B}_{N,k})$, where each block $\mathsf{B}_{j,k}$ is an $M \times M$ circulant matrix, which encodes the interaction between the first and j-th column for all rows in slice k. The non-zero circulant blocks $\mathsf{B}_{j,k}$ of size $M \times M$ are as follows (see Fig. 3.11):

$$\begin{aligned}
\mathsf{B}_{1,1} &= \text{circ}(1, -\theta_1, 0, 0, \ldots, 0, -\theta_1), \\
\mathsf{B}_{2,1} &= \text{circ}(-\theta_0, -\theta_3, 0, 0, \ldots, 0, -\theta_4), \\
&\vdots \\
\mathsf{B}_{N,1} &= \text{circ}(-\theta_0, -\theta_4, 0, 0, \ldots, 0, -\theta_3) = (\mathsf{B}_{2,1})^T, \\
\mathsf{B}_{1,2} &= \text{circ}(-\theta_2, -\theta_7, 0, 0, \ldots, 0, -\theta_8), \\
\mathsf{B}_{2,2} &= \text{circ}(-\theta_5, 0, 0, \ldots, 0), \\
&\vdots \\
\mathsf{B}_{N,2} &= \text{circ}(-\theta_6, 0, 0, \ldots, 0), \\
\mathsf{B}_{1,D} &= (\mathsf{B}_{1,2})^T, \\
\mathsf{B}_{2,D} &= \mathsf{B}_{N,2}, \\
&\vdots \\
\mathsf{B}_{N,D} &= \mathsf{B}_{2,2}.
\end{aligned}$$

All other circulants $\mathsf{B}_{j,k} = \mathbf{0}$, $2 < j < N, 2 < K < D$.

The non-zero block circulants of level 2 of size $MN \times MN$ are defined as follows:

$$\begin{aligned}
\mathbf{B}_1 &= \text{bcirc}(\mathsf{B}_{1,1}, \mathsf{B}_{2,1}, \mathbf{0}, \ldots, \mathbf{0}, (\mathsf{B}_{2,1})^T), \\
\mathbf{B}_2 &= \text{bcirc}(\mathsf{B}_{1,2}, \mathsf{B}_{2,2}, \mathbf{0}, \ldots, \mathbf{0}, \mathsf{B}_{N,2}), \\
&\vdots \\
\mathbf{B}_D &= (\mathbf{B}_2)^T.
\end{aligned}$$

All other block-circulants $\mathbf{B}_k = \mathbf{0}$, $2 < k < D$. The whole matrix $\mathbf{B}$ is then constructed from (4.19).

It can be seen that the matrix $\mathbf{B}$ possesses the block-circulant of level three structure discussed in the previous sections. It can be seen that $\mathbf{B}$ is very sparse, due to the relatively small size of the GMRF neighbourhood compared with the lattice size, while the size of $\mathbf{B}$ is fairly large. Therefore, the effective storage is very important. The generating tensor of $\mathbf{B}$ is denoted by $\mathbf{A}$, and is obtained via (4.5) after the substitutions $\mathbf{C} = \mathbf{B}$, $\mathbf{T} = \mathbf{A}$, $\mathbf{C}_k = \mathbf{B}_k$, $m = M$, $n = N$, and $p = D$:

$$\mathbf{A}(i, j, k) = \mathbf{B}_k(1, i + (j-1)M) = \mathbf{B}(1, (i + (j-1)M) + (k-1)MN), \quad (4.20)$$

where $i = 1, \ldots, M$, $j = 1, \ldots, N$ and $k = 1, \ldots, D$.

In this way, the very sparse $\mathbf{B}$ of size $MND \times MND$ can be stored in the more compact 3-D tensor $\mathbf{A}$ of size $M \times N \times D$, i.e. it is of the same size as the original image lattice.

From the described procedure of constructing the interaction matrix, it follows that its first row contain the interactions of the the node $(1, 1, 1)$ with its neighbours. This combined with the toroidal boundary assumption results in the generating tensor structure as illustrated in Figure 4.3 for the case of a second order 3-D GMRF. All non-specified entries are zero.

By comparing the structure of $\mathbf{A}$ and the neighbourhood geometry (Figures 4.3 and 3.11) it can be observed that there is an easy 'graphical' way of constructing $\mathbf{A}$. This is done by aligning the central slice of the neighbourhood geometry with the $\mathbf{A}(1, 1, 1)$ entry, assigning 1 to it and taking all θs with a minus sign, while respecting the toroidality. This is equivalent of applying formulas (3.14) and (4.20).

It is worth noting that in the joint probability expression (3.13) computations of both determinant and quadratic form involving the interaction matrix are required. Hence, a direct evaluation of it can be computationally unattractive. Due to the properties of the block-circulant of level 3 $\mathbf{B}$, however, the fast algorithms developed in Section 4.3 using the generating tensor $\mathbf{A}$ can be applied. Thus, direct storage and

manipulation of **B** is avoided, making the 3-D GMRF joint probability computations feasible even for big lattices.

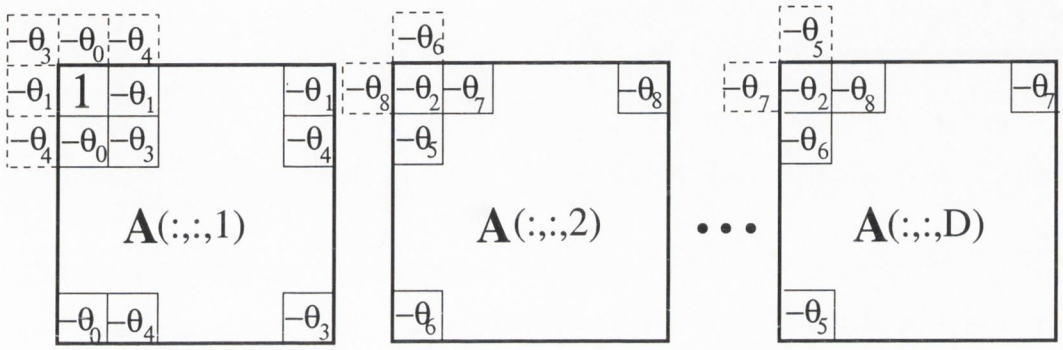

Figure 4.3: The generating tensor **A** for a second order 3-D GMRF.

4.4.2 Algorithm

The synthesis algorithm starts with the matrix-vector representation

$$\mathbf{Bx} = \mathbf{e}$$

$$\mathbf{x} = \mathbf{B}^{-1}\mathbf{e}, \tag{4.21}$$

related to (3.20) written for a 3-D GMRF of size $M \times N \times D$, where $\mathbf{x}$ is the $N_T \times 1$ vector containing the realization of the field at all nodes. $\mathbf{B}$ is the GMRF interaction matrix (Sections 3.4.1 and 4.4.1), and $\mathbf{e}$ is the $N_T \times 1$ vector for the correlated Gaussian noise.

The algorithm begins with a generation of the vector $\mathbf{e}$ aiming to construct $\mathbf{x}$, obeying (4.21). Substituting (4.13) in (4.21) it follows:

$$\mathbf{x} = \boldsymbol{\mathcal{F}}^H \boldsymbol{\Lambda}^{-1} \boldsymbol{\mathcal{F}} \mathbf{e}, \tag{4.22}$$

where $\boldsymbol{\Lambda}$ is the diagonal matrix of eigenvalues of $\mathbf{B}$.

Furthermore, the relationship between the GMRF and the SAR models can be used to simplify the procedure starting with a vector $\mathbf{n}$ containing independent and identically distributed realizations of Gaussian noise with zero mean and standard

deviation σ^2 instead of $\mathbf{e}$. Then $\sqrt{\mathbf{B}}$ should be used instead of $\mathbf{B}$ to obtain correct model structure [14].

Following the concepts from Section 4.3, fast 3-D FFT implementation of (4.22) is proposed. The following structures are used: $\mathbf{X}$ being a field of original dimensions $M \times N \times D$, encoded in the vector $\mathbf{x}$, $\mathbf{N}$ is a 3-D structure equivalent to $\mathbf{n}$ and $\mathbf{A}$ is the generating tensor of $\mathbf{B}$. μ is the mean parameter defined in Section 3.4.

Algorithm :

1. Generate an $M \times N \times D$ array $\mathbf{N} \sim \sigma^2 N(\mathbf{0}, \mathbf{I})$.
2. Apply a 3-D FFT on $\mathbf{N}$ and save the result in $\mathbf{Y}$.
3. Compose the $M \times N \times D$ tensor $\mathbf{A}$ from $\boldsymbol{\theta}$ using Fig. 4.3.
4. Apply a 3-D FFT to $\mathbf{A}$ and save the result in $\mathbf{A}$.
5. Reassign $\mathbf{Y}(i,j,k) \leftarrow \mathbf{Y}(i,j,k)/\sqrt{\mathbf{A}(i,j,k)}$.
6. Apply a 3-D inverse FFT to $\mathbf{Y}$ and save the result in $\mathbf{Y}$.
7. $\mathbf{Y} + \mu$ is a sample of the GMRF $\mathbf{X}$.

In the case of 2-D GMRFs, Lankshmanan and Derin [50] give a *sufficient condition* for $\boldsymbol{\theta}$, so that $\mathbf{B} = \mathbf{B}(\boldsymbol{\theta})$ is positive definite assuring that (3.13) is a valid joint p.d.f.: $\sum |\boldsymbol{\theta}| < 0.5$. This condition is easy to check, but describes only a subspace of the valid parameter space. The authors point out that the above result can be extended for real and complex GMRFs defined over L-D lattices. In the simulations in this thesis, the 3-D GMRFs, generated by the above algorithm, satisfy the sufficient condition above.

4.5 SIMULATION EXAMPLE

As an illustrative example consider a 3-D lattice with dimensions $M = N = 4$ and $D = 3$. Then, the total number of nodes is $N_T = 48$. A second order 3-D GMRF with entries of the correlation parameters vector $\boldsymbol{\theta}$ ($q = 2, n_q = 9$) given in Table 4.3 is assumed.

Table 4.3: Correlation parameters for a second order 3-D GMRF.

θ_0	θ_1	θ_2	θ_3	θ_4	θ_5	θ_6	θ_7	θ_8
-0.002	0.047	-0.11	0.15	-0.017	0.044	-0.02	0.08	-0.019

Toroidal boundary assumptions have been adopted, thus the following circulant matrices (of level 1) $\mathsf{B}_{j,k}, j = 1, \ldots, 4, k = 1, 2, 3$ of size 4×4 can be constructed:

For $k = 1$:

$$\begin{aligned} \mathsf{B}_{1,1} &= \text{circ}(1, -0.047, 0, -0.047) \\ \mathsf{B}_{2,1} &= \text{circ}(0.002, -0.15, 0, 0.017) \\ \mathsf{B}_{3,1} &= \text{circ}(0, 0, 0, 0) = \mathbf{0} \\ \mathsf{B}_{4,1} &= (\mathsf{B}_{2,1})^T \end{aligned}$$

For the second 2-D slice of the 3-D block ($k = 2$):

$$\begin{aligned} \mathsf{B}_{1,2} &= \text{circ}(0.11, -0.08, 0, 0.019) \\ \mathsf{B}_{2,2} &= \text{circ}(-0.044, 0, 0, 0) \\ \mathsf{B}_{3,2} &= \mathbf{0} \\ \mathsf{B}_{4,2} &= \text{circ}(0.02, 0, 0, 0) \end{aligned}$$

For $k = 3$: $\mathsf{B}_{1,3} = (\mathsf{B}_{1,2})^T$, $\mathsf{B}_{2,3} = \mathsf{B}_{4,2}$, $\mathsf{B}_{3,3} = \mathbf{0}$, $\mathsf{B}_{4,3} = \mathsf{B}_{2,2}$.

The three symmetric block-circulant matrices of level 2 $\mathbf{B}_k, k = 1, 2, 3$ corresponding to the interaction between the 2-D slices in the 3-D data have the following form:

$$\begin{aligned}
\mathbf{B}_1 &= \text{bcirc}\left(\mathsf{B}_{1,1}, \mathsf{B}_{2,1}, \mathbf{0}, (\mathsf{B}_{2,1})^T\right) \\
\mathbf{B}_2 &= \text{bcirc}\left(\mathsf{B}_{1,2}, \mathsf{B}_{2,2}, \mathbf{0}, \mathsf{B}_{4,2}\right) \\
\mathbf{B}_3 &= \text{bcirc}\left((\mathsf{B}_{1,2})^T, \mathsf{B}_{4,2}, \mathbf{0}, \mathsf{B}_{2,2}\right) = (\mathbf{B}_2)^T.
\end{aligned}$$

For example, the $\mathbf{B}_1$ matrix of size 16×16 is of class $\mathcal{C}(4,4)$ and has the form:

$$\mathbf{B}_1 = \begin{pmatrix} \mathsf{B}_{1,1} & \mathsf{B}_{2,1} & \mathsf{B}_{3,1} & \mathsf{B}_{4,1} \\ \mathsf{B}_{4,1} & \mathsf{B}_{1,1} & \mathsf{B}_{2,1} & \mathsf{B}_{3,1} \\ \mathsf{B}_{3,1} & \mathsf{B}_{4,1} & \mathsf{B}_{1,1} & \mathsf{B}_{2,1} \\ \mathsf{B}_{2,1} & \mathsf{B}_{3,1} & \mathsf{B}_{4,1} & \mathsf{B}_{1,1} \end{pmatrix} = \begin{pmatrix} \mathsf{B}_{1,1} & \mathsf{B}_{2,1} & \mathbf{0} & (\mathsf{B}_{2,1})^T \\ (\mathsf{B}_{2,1})^T & \mathsf{B}_{1,1} & \mathsf{B}_{2,1} & \mathbf{0} \\ \mathbf{0} & (\mathsf{B}_{2,1})^T & \mathsf{B}_{1,1} & \mathsf{B}_{2,1} \\ \mathsf{B}_{2,1} & \mathbf{0} & (\mathsf{B}_{2,1})^T & \mathsf{B}_{1,1} \end{pmatrix} =$$

$$\left(\begin{array}{cccc|cccc|cccc|cccc}
1 & -0.047 & 0 & -0.047 & 0.002 & -0.15 & 0 & 0.017 & 0 & 0 & 0 & 0 & 0.002 & 0.017 & 0 & -0.15 \\
-0.047 & 1 & -0.047 & 0 & 0.017 & 0.002 & -0.15 & 0 & 0 & 0 & 0 & 0 & -0.15 & 0.002 & 0.017 & 0 \\
0 & -0.047 & 1 & -0.047 & 0 & 0.017 & 0.002 & -0.15 & 0 & 0 & 0 & 0 & 0 & -0.15 & 0.002 & 0.017 \\
-0.047 & 0 & -0.047 & 1 & -0.15 & 0 & 0.017 & 0.002 & 0 & 0 & 0 & 0 & 0.017 & 0 & -0.15 & 0.002 \\
\hline
0.002 & 0.017 & 0 & -0.15 & 1 & -0.047 & 0 & -0.047 & 0.002 & -0.15 & 0 & 0.017 & 0 & 0 & 0 & 0 \\
-0.15 & 0.002 & 0.017 & 0 & -0.047 & 1 & 0.13 & 0 & 0.017 & 0.002 & -0.15 & 0 & 0 & 0 & 0 & 0 \\
0 & -0.15 & 0.002 & 0.017 & 0 & -0.047 & 1 & -0.047 & 0 & 0.017 & 0.002 & -0.15 & 0 & 0 & 0 & 0 \\
0.017 & 0 & -0.15 & 0.002 & -0.047 & 0 & -0.047 & 1 & -0.15 & 0 & 0.017 & 0.002 & 0 & 0 & 0 & 0 \\
\hline
0 & 0 & 0 & 0 & 0.002 & 0.017 & 0 & -0.15 & 1 & -0.047 & 0 & -0.047 & 0.002 & -0.15 & 0 & 0.017 \\
0 & 0 & 0 & 0 & -0.15 & 0.002 & 0.017 & 0 & -0.047 & 1 & -0.047 & 0 & 0.017 & 0.002 & -0.15 & 0 \\
0 & 0 & 0 & 0 & 0 & -0.15 & 0.002 & 0.017 & 0 & -0.047 & 1 & -0.047 & 0 & 0.017 & 0.002 & -0.15 \\
0 & 0 & 0 & 0 & 0.017 & 0 & -0.15 & 0.002 & -0.047 & 0 & -0.047 & 1 & -0.15 & 0 & 0.017 & 0.002 \\
\hline
0.002 & -0.15 & 0 & 0.017 & 0 & 0 & 0 & 0 & 0.002 & 0.017 & 0 & -0.15 & 1 & -0.047 & 0 & -0.047 \\
0.017 & 0.002 & -0.15 & 0 & 0 & 0 & 0 & 0 & -0.15 & 0.002 & 0.017 & 0 & -0.047 & 1 & -0.047 & 0 \\
0 & 0.017 & 0.002 & -0.15 & 0 & 0 & 0 & 0 & 0 & -0.15 & 0.002 & 0.017 & 0 & -0.047 & 1 & -0.047 \\
-0.15 & 0 & 0.017 & 0.002 & 0 & 0 & 0 & 0 & 0.017 & 0 & -0.15 & 0.002 & -0.047 & 0 & -0.047 & 1
\end{array}\right) \tag{4.23}$$

In this example the $\mathbf{B}$ matrix is a symmetric 48×48 matrix of class $\mathcal{C}(4,4,3)$ of the form:

$$\mathbf{B} = \begin{pmatrix} \mathbf{B}_1 & \mathbf{B}_2 & \mathbf{B}_3 \\ \mathbf{B}_3 & \mathbf{B}_1 & \mathbf{B}_2 \\ \mathbf{B}_2 & \mathbf{B}_3 & \mathbf{B}_1 \end{pmatrix} = \begin{pmatrix} \mathbf{B}_1 & \mathbf{B}_2 & (\mathbf{B}_2)^T \\ (\mathbf{B}_2)^T & \mathbf{B}_1 & \mathbf{B}_2 \\ \mathbf{B}_2 & (\mathbf{B}_2)^T & \mathbf{B}_1 \end{pmatrix}, \tag{4.24}$$

where the subblocks are defined as given above.

The corresponding generating tensor $\mathbf{A}$ has the following structure (see Fig. 4.3):

$$\mathbf{A}(:,:,1) = \begin{pmatrix} 1 & -0.047 & 0 & -0.047 \\ 0.002 & -0.15 & 0 & 0.017 \\ 0 & 0 & 0 & 0 \\ 0.002 & 0.0017 & 0 & -0.15 \end{pmatrix}$$

$$\mathbf{A}(:,:,2) = \begin{pmatrix} 0.11 & -0.08 & 0 & 0.019 \\ -0.044 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0.02 & 0 & 0 & 0 \end{pmatrix}$$

$$\mathbf{A}(:,:,3) = \begin{pmatrix} 0.11 & 0.019 & 0 & -0.08 \\ 0.02 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ -0.044 & 0 & 0 & 0 \end{pmatrix} \tag{4.25}$$

The 48 eigenvalues of $\mathbf{B}$ can be computed using $\mathbf{A}$ and the algorithm, described in Section 4.3.3:

$$\mathbf{\Lambda}(:,:,1) = \begin{pmatrix} 0.6940 & 1.1760 & 1.6580 & 1.1760 \\ 1.0040 & 1.5540 & 1.4360 & 0.8860 \\ 1.3140 & 1.2640 & 1.2140 & 1.2640 \\ 1.0040 & 0.8860 & 1.4360 & 1.5540 \end{pmatrix}$$

$$\mathbf{\Lambda}(:,:,2) = \begin{pmatrix} 0.6190 & 1.0895 & 1.2170 & 0.7465 \\ 0.9679 & 1.5063 & 1.0339 & 0.4954 \\ 1.0950 & 1.0335 & 0.6290 & 0.6905 \\ 0.7461 & 0.6166 & 0.8121 & 0.9417 \end{pmatrix}$$

$$\mathbf{\Lambda}(:,:,3) = \begin{pmatrix} 0.6190 & 0.7465 & 1.2170 & 1.0895 \\ 0.7461 & 0.9417 & 0.8121 & 0.6166 \\ 1.0950 & 0.6905 & 0.6290 & 1.0335 \\ 0.9679 & 0.4954 & 1.0339 & 1.5063 \end{pmatrix} \tag{4.26}$$

Then, $\det(\mathbf{B}) = 2.2115e - 010$ according to (4.16).

To illustrate the efficiency of the FFT-based computations with circulants of level 3, several interaction matrices $\mathbf{B} = \mathbf{B}(\boldsymbol{\theta})$ with different dimensions were constructed with $\boldsymbol{\theta}$ given in Table 4.3. The dimensions of the lattice were chosen to be $M = N = 15$ and variable $D = 3, 5, 7, 9, 11$, because these settings were used in the experiments throughout the thesis. The corresponding interaction matrices dimensions were $N_T \times N_T$, where $N_T = 675, 1125, 1575, 2025, 2475$ respectively for each D. The corresponding generating tensors $\mathbf{A}$ are of size $15 \times 15 \times D$.

The elapsed times for the calculation of the determinants of these matrices via the standard MATLAB function (on a 500 MHz PC with 130596 KB RAM) and via the 3-D FFT-based method described in Section 4.3.3 are plotted versus N_T in semilog scale in Figure 4.4.

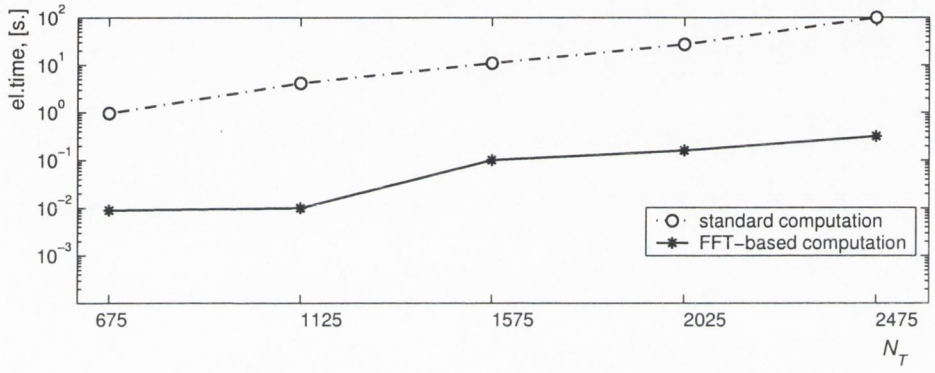

Figure 4.4: Standard vs. FFT-based computation of the determinant of $\mathbf{B}$.

The repetitive computation of the joint GMRF p.d.f. (3.13) involving determinant and quadratic forms evaluations is a principal computational overhead. Thus, the significant speedup achieved is vital for maintaining reasonable computation times when working with lattices with increasing dimensions, as required by methods presented in later chapters.

Finally, to illustrate the result of the synthesis algorithm, a GMRF of dimensions $4 \times 4 \times 3$ and generating tensor $\mathbf{A}$ given by (4.25) is shown next.

The GMRF parameters were $\mu = 128$, $\sigma = 20$ and $\boldsymbol{\theta}$ as given in Table 4.3. It can be shown that these correlation parameters satisfy the sufficient condition as $\sum |\boldsymbol{\theta}| = 0.489$.

The three slices of the Gaussian noise array (the grey levels are scaled) $\mathbf{E}$ and the resulting $\mathbf{X}$ are shown in Figures 4.5 and 4.6.

More synthetic volumes of size $128 \times 128 \times 11$ used in the experiments in this work are given in Appendix C, Section C.1.

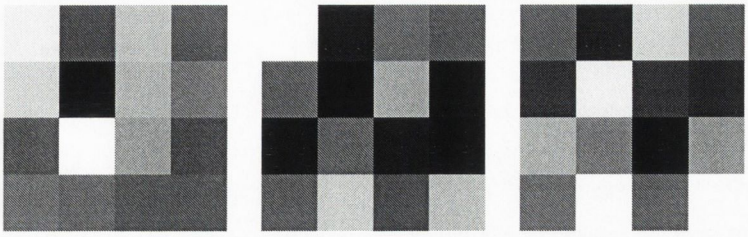

Figure 4.5: The Gaussian noise sequence **E**.

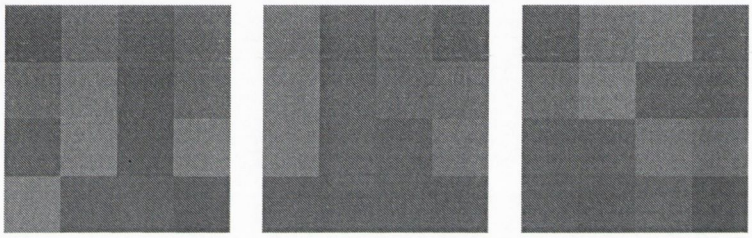

Figure 4.6: The synthesized GMRF **X**.

4.6 CONCLUSIONS

In this chapter a novel and effective algorithm for sampling GMRFs over 3-D lattices has been presented. It allows the synthesis of stationary volumes with given parameters. The toroidal assumption results in the circulant of level 3 structure of the model interaction matrix **B**. This structure makes possible the use of fast 3-D FFT-based implementation of the algorithm. Thus, a tool for testing the accuracy of the stationary GMRF parameter estimation and the labelling of mixture of such fields, being subject of the following chapters, has been created.

Chapter 5

VOXEL-BY-VOXEL SEGMENTATION

5.1 INTRODUCTION

As explained in Chapters 2 and 3 image segmentation is a labelling problem. The main task of this work is to perform region labelling of a mixture of stationary 3-D GMRF data in a supervised manner. Thus, a method is needed for constructing the codebook (Section 3.5.2), which carries the information for the number of underlying stationary models and their representative parameters.

A technique for estimating the parameters of a stationary 3-D GMRF is developed. The codebook entries are obtained by off-line processing of selected stationary sub-lattices of the original grey-level data using this technique. These estimates are the representative *features* for each stationary model. Similarly, a *feature vector* is calculated for each lattice node from the non-stationary data. Then given the codebook, the appropriate label for the node is selected by comparing its feature vector with the codebook entries using some *distance measure*. Therefore, the labelling is performed via parameter estimation followed by *classification*. The classification is done in a *voxel-by-voxel* manner.

In Section 5.2 the technique for parameter estimation of a stationary GMRF is presented. The supervised labelling of nodes from the 3-D non-stationary data is presented in Section 5.3 and the voxel-by-voxel approach is discussed in Section 5.4. Finally, the main conclusions are given in Section 5.5.

5.2 PARAMETER ESTIMATION FOR THE STATIONARY GMRF MODEL

The parameters of a 2-D GMRF can be estimated in several ways, such as by Besag's coding scheme [4], by pseudo-likelihood maximization [5], by minimizing the sum of square errors (Least Squares (LS)) [14] or by maximum likelihood estimation. The LS approach can be extended for L-D GMRFs. In this work, it is used for estimating the parameters of a stationary 3-D GMRF.

The estimation of the parameters is performed under the assumption of the *stationarity* of the GMRF model within a sub-lattice or computational volume as defined in Section 3.2.2. The volume centered at node t is ω_t. The total number of nodes in the volume is $n_T = mnd$.

A zero-mean process within $\omega_t \subset \Lambda_I$ is assumed (which can be arranged via pre-processing, namely, subtraction of the sample mean), i.e. $\mu_t = 0$ and the model equation (3.20) becomes:

$$x_t = \sum_{s \in \eta_t^+} \theta_{t,s}(x_s + x_{-s}) + e_t, \tag{5.1}$$

To estimate the parameters of the process X_t at the node t, all the data from ω_t are used. The aim is to minimize the sum of squared errors, $\sum_{r \in \omega_t} e_r^2$, between the grey level at a node $r \in \omega_t$ and its modelled value, where

$$e_r = x_r - \sum_{s \in \eta_r^+} \theta_{r,s}(x_s + x_{-s}) \tag{5.2}$$

Hence:

$$x_r = \mathbf{y}_r^T \boldsymbol{\theta}_r + e_r, \tag{5.3}$$

where $\mathbf{y}_r = [x_s + x_{-s}, s \in \eta_r^+]^T$, and $\boldsymbol{\theta}_r = [\theta_{r,s}, s \in \eta_r^+]^T$. The latter is assumed constant over ω_t, i.e. $\boldsymbol{\theta}_r = \boldsymbol{\theta}_t = [\theta_{t,s}, s \in \eta_t^+]^T$, $\forall r \in \omega_t$. Vectors $\mathbf{y}_r$ and $\boldsymbol{\theta}_t$ are each equal in length to n_q.

Gathering (5.3) into vector-matrix form for all $r \in \omega_t$:

$$\mathbf{x}_t = \mathbf{Y}_t \boldsymbol{\theta}_t + \mathbf{e}_t, \tag{5.4}$$

where $\mathbf{x}_t = [x_r, r \in \omega_t]^T$ and $\mathbf{e}_t = [e_r, r \in \omega_t]^T$ are column vectors of length n_T obtained by raster scanning across ω_t. The matrix $\mathbf{Y}_t$ is of size $n_T \times n_q$ with rows given by $\mathbf{y}_r^T, r \in \omega_t$.

The equations (5.4) are a system of standard normal equations, a solution for which can be found using the LS method, as shown in Appendix B.

Thus, using the LS technique, the parameter vector (3.8) $\mathbf{p}_t = [p_{t,1}, \ldots, p_{t,n_f}]^T$,$n_f = n_q + 2$ for the stationary GMRF model of order q within ω_t has been estimated. This estimate, $\widehat{\mathbf{p}}_t$, is considered to be a vector of representative features $f_{t,\mathrm{j}}$ for the model and is called the **feature vector** for the node $t \in \mathbf{\Lambda}_I$, i.e. $\mathbf{f}_t = \widehat{\mathbf{p}}_t$:

$$\mathbf{f}_t = [\widehat{\mu}_t, \widehat{\sigma}_t, \widehat{\boldsymbol{\theta}}_t^T]^T, \tag{5.5}$$

where the estimates $\widehat{\boldsymbol{\theta}}_t$, $\widehat{\mu}_t$ and $\widehat{\sigma}_t$ are given by (B.8), (B.9) and (B.10) respectively.

The n_f-dimensional feature space Φ is a Euclidean space defined as the range of possible $f_{t,\mathrm{j}}$s. Then, $\mathbf{f}_t$ can be viewed as a point (ordered n_f-tuple) in Φ (Fig. 5.1).

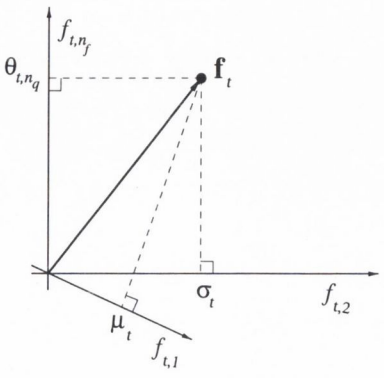

Figure 5.1: Feature vector as a point in the n_f-dimensional feature space, Φ.

5.3 SUPERVISED SEGMENTATION OF GMRF MIXTURES

The stationary GMRF (Section 3.4) is the model for 3-D homogeneous texture in this work. A data volume to be segmented is considered to be a mixture of T textures, modelled by T stationary GMRFs on 3-D lattices. Then, each inferred label $l_t = \mathsf{i}$, $\mathsf{i} \in \mathbf{L}$ indicates that the lattice node t belongs to the i-th stationary texture model.

As stated in Section 3.5.2 the problem of segmentation, namely the inference of $\mathbf{L}$ given the data, can be performed in a supervised or unsupervised manner. If the codebook $\mathbf{C}$ is available *a priori* then this is a supervised segmentation. Otherwise, if it is learned during the segmentation, the problem is unsupervised. The supervised segmentation has two distinct parts:

(i) Codebook generation and

(ii) Classification of the nodes.

5.3.1 Codebook Generation

The codebook may be generated in different ways. For instance, there might be a database of all constituent textures available off-line. The codebook generation then, is a translation of the off-line knowledge into $\mathbf{C}$. Each codebook vector is obtained by averaging of the respective LS estimates at nodes from manually chosen sublattices from the grey-level data $\mathbf{X}$:

$$\mathbf{c}_{\mathsf{i}} = \frac{1}{n_c} \sum_{r|l_r=\mathsf{i}} \mathbf{f}_r. \tag{5.6}$$

Here, $\mathbf{f}_r$ are the feature vectors (5.5) calculated at the representative nodes r which are labelled with class i and n_c is the number of such nodes. The representative sublattices are chosen carefully to satisfy the texture stationarity assumption necessary to obtain the LS estimates as described in Section 5.2.

5.3.2 Distance Measures

After the codebook has been created, a feature vector for each node is obtained via the LS estimation procedure. Therefore, the segmentation task in the image space has been transformed into classification in Φ. Hence, a measure of similarity between the feature vector and the codebook entries is needed.

A general expression of a *distance* between the two n_f-dimensional vectors $\mathbf{f}_t$ and $\mathbf{c}_i$ is

$$D(\mathbf{f}_t, \mathbf{c}_i) = (\mathbf{f}_t - \mathbf{c}_i)^T \mathbf{S}_i^{-1} (\mathbf{f}_t - \mathbf{c}_i), \tag{5.7}$$

where $\mathbf{S}_i^{-1}$ is the inverse of the positive definite distance matrix of size $n_f \times n_f$. The effect of this matrix is to scale the distance along each feature axis (Fig. 5.1).

In practice, the two most commonly used distance measures for classification purposes are:

(i) the simple Euclidean distance, D_E, whose distance matrix is the identity matrix $\mathbf{I}_{n_f}$ and

(ii) the Mahalanobis distance [27], D_M, whose distance matrix is the inverse covariance matrix for all $\mathbf{c}_i$s

The performance of the distance measures for the task of texture classification has been tested using the 'leave-one-out' method on an extensive number of textures in [76] and superiority of D_M has been shown.

5.3.3 Parameter Estimation Accuracy

To evaluate the accuracy of the stationary GMRF parameter estimation method, the parameters of some synthetic GMRFs have been estimated using the LS technique presented in Section 5.2. The GMRFs have been generated via the synthesis algorithm from Section 4.4.2. The actual parameter values p and their estimates (features) f, for the first order GMRFs are given in Table 5.1, for the second order in Table 5.2, and for the third order in Table 5.3 for illustrative purposes.

Table 5.1: Parameter and estimate comparisons for first order GMRFs.

	model 1		**model 2**	
	p	f	p	f
μ	122	121.6	120	119.48
σ	20	19.833	20	19.994
θ_0	0.3	0.316	0.32	0.332
θ_1	-0.09	-0.12	-0.09	-0.092
θ_2	0.15	0.136	-0.2	-0.196

Table 5.2: Parameter and estimate comparisons for second order GMRFs.

	model 1		**model 2**		**model 3**	
	p	f	p	f	p	f
μ	122	121.47	120	119.49	124	120.6
σ	20	19.729	20	19.648	20	19.793
θ_0	-0.002	-0.0064	-0.004	-0.0386	-0.04	-0.0039
θ_1	0.047	0.0469	0.048	0.0471	0.047	0.0474
θ_2	-0.11	-0.1145	0.084	0.0821	-0.02	-0.0254
θ_3	0.15	0.1491	0.15	0.1495	0.15	0.1389
θ_4	-0.017	-0.0136	-0.017	-0.0218	-0.017	-0.0205
θ_5	0.044	0.0376	-0.013	-0.0188	0.12	0.0224
θ_6	-0.02	-0.0188	0.021	0.0180	0.05	0.0498
θ_7	0.08	0.0788	-0.09	-0.0942	-0.01	-0.0101
θ_8	-0.019	-0.0208	0.038	0.0359	0.006	0.0127

The reported estimates are obtained by averaging the LS estimates over the number of computational volumes within $\mathbf{\Lambda}_I$. The computational volume dimensions were chosen to be $n = 15$ and $d = 5$. It is important to note the size of the ω_t is a key operating parameter for the success of the segmentation. The n and d parameters

must be chosen big enough to yield reliable estimates and at the same time small enough to protect the texture stationarity assumption.

Table 5.3: Parameter and estimate comparisons for third order GMRFs.

	model 1		**model 2**		**model 3**	
	p	f	p	f	p	f
μ	120	119.6	122	121.48	124	123.49
σ	20	19.3	22	21.583	24	23.49
θ_0	-0.01	-0.0110	-0.02	-0.024	-0.015	-0.014
θ_1	0.13	0.1290	0.13	0.131	0.13	0.13
θ_2	0.02	0.0180	-0.03	-0.035	0.002	0.0009
θ_3	0.012	0.0026	0.01	0.012	0.011	0.009
θ_4	-0.07	-0.0720	-0.06	-0.056	-0.08	-0.083
θ_5	0.014	0.0134	-0.02	-0.026	0.1	0.094
θ_6	-0.002	-0.0004	0.0015	0.0016	-0.3	-0.032
θ_7	-0.007	-0.0087	0.002	0.0028	0.02	0.0137
θ_8	0.015	0.0170	-0.003	-0.006	-0.07	-0.072
θ_9	0.05	0.0420	-0.02	-0.019	0.001	0.0024
θ_{10}	0.03	0.0327	-0.04	-0.042	-0.007	-0.011
θ_{11}	-0.04	-0.0430	0.05	0.046	-0.005	-0.002
θ_{12}	0.08	0.0780	-0.07	-0.0646	0.009	0.0046

The normalized Euclidean distance $D_{NE}(\cdot)$ between two vectors $\mathbf{x}$ and $\mathbf{y}$ is a simplified version of the Mahalanobis distance, which requires less computations and quantifies effectively the human perception of similarity/difference between 2-D GMRF textures:

$$D_{NE}(\mathbf{x}, \mathbf{y}) = \sum_{\mathrm{j}} \frac{(x_{\mathrm{j}} - y_{\mathrm{j}})^2}{x_{\mathrm{j}}^2 + y_{\mathrm{j}}^2} \tag{5.8}$$

Since, later, this will be employed in voxel classification, it is used here to assess the accuracy of the estimates in Tables 5.1 to 5.3.

Specifically, the distance can be calculated between the actual parameter vectors $\mathbf{p}_i$ and their estimates (the feature vectors) $\mathbf{f}_i$ as well as between the vectors representing the i-th model. The latter will be vital in classification. The results are shown in Figure 5.2. Although the scale is not exact, the figures illustrate the distances between the parameter and feature vectors in Φ.

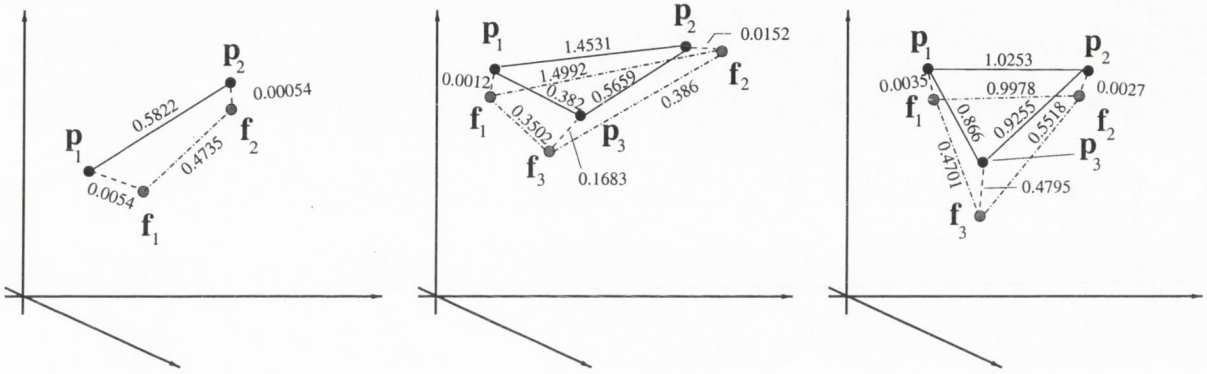

Figure 5.2: Normalized Euclidian distances between the parameter vectors and their estimates for first, second and third order synthetic GMRFs respectively.

The results indicate that the estimation geometry accurately reflects the geometry of the original parameter vectors. This is a key requirement for the successful performance of a *classifier* in the feature space Φ.

5.4 VOXEL-BY-VOXEL SEGMENTATION

After constructing the codebook, for each node $t \in \Lambda_I$ a feature vector $\mathbf{f}_t$ (5.5) is computed using the volume ω_t as discussed in Section 5.2. The next stage of the supervised segmentation is the classification of nodes.

The task of **classification** is one of assigning an input (data observation) to a category (class) [27]. Thus, it is a labelling problem (Section 3.5), where each class is denoted by a unique label. Classification is an important part of any pattern recognition system [81]. Its aim is to distinguish between different data entries and,

based on their similarity, to classify them into predefined classes. In the context of the 3-D region labelling we would like to classify single nodes or connected groups of nodes, called *regions* into classes of different textures.

Similar feature vectors form clusters in the feature space. Usually similarity is defined in terms of a distance function. Forming the clusters is equivalent to dividing the space into sub-spaces, each of which corresponds to a different class. Then the task of the classification can be viewed as determining the sub-space of the feature space to which the given feature vector and hence its associated node belong.

For a classifier to be successful, it is important that the classes are separable, i.e. that discrimination hyper-surfaces between the clusters exist. The surfaces can be defined by discriminant functions [81]. Another way to construct the classifier is based on the *minimum distance principle* [81, 90], which is a special case of a classifier with discriminant functions, but which is easily implemented on digital computers.

An illustration of the minimum distance classifier principle used in this work is shown in Figure 5.3.

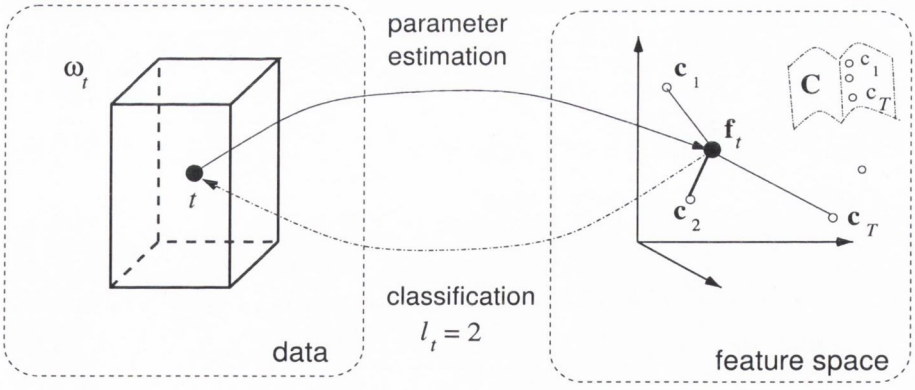

Figure 5.3: Minimum distance classifier.

The output of the minimum distance classifier is the class label assignment:

$$l_t = \arg\min_{\mathrm{i}} D(\mathbf{f}_t, \mathbf{c}_\mathrm{i}), \forall \mathrm{i} \in \{1, \ldots, T\}. \tag{5.9}$$

l_t is the element of the label field $\mathbf{L}$ as defined in Section 3.5.1 corresponding to the node $t \in \Lambda_I$. For the case depicted in the figure, the node t has been determined as belonging to the second class ($l_t = 2$), because the distance between the second codebook entry $\mathbf{c}_2$ and the feature vector $\mathbf{f}_t$ is the minimal one.

The most straightforward approach is to classify each node in Λ sequentially. The labels, assigned to all nodes represent the result of the classification as a *label map*, which is a realization of the label field. The unprocessed nodes $t \in \Lambda_B$ in the label map are denoted in black. This method shall be called: Voxel-by-Voxel (VBV) in the 3-D or Pixel by Pixel (PBP) segmentation in the 2-D case.

To test the classification accuracy and speed of the VBV and PBP schemes, synthetic 3-D data has been generated. Several stationary 3-D GMRFs of order 1 have been synthesized using the algorithm in Section 4.4.2 and a ground truth volumetric mask has been used to compose the mixture of stationary fields, i.e. to obtain the non-stationary data.

5.4.1 A two texture problem

Two stationary GMRFs of order $q = 2$ and set of parameters $\mathbf{p}$, given in the columns 'model 1' and 'model 2' (denoted by p in Table 5.2), have each been sampled on a 3-D lattice with dimensions $M = N = 128$, $D = 11$. The volumes are shown slice by slice in Figures C.1 and C.2, Appendix C. It should be noted that the values of μ, σ, and the set of 2-D correlation parameters $\boldsymbol{\theta}_h$ (3.9) are chosen to be very close. On the other hand, the values for the parameters controlling the correlation in the third dimension, $\boldsymbol{\theta}_v$, (3.10) are chosen to be different for the two models.

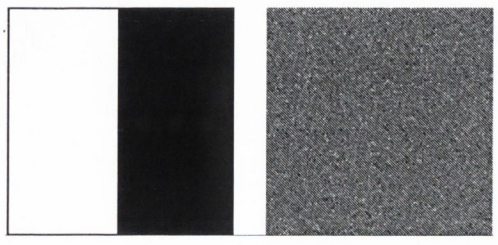

Figure 5.4: Ground truth mask and grey-level data, two texture problem: slice #6.

The ground truth mask and the synthetic data for the middle slice #6 are shown in Fig.5.4 and those for the whole volume are given in Fig. C.7.

The codebook entries have been constructed as described in Section 5.3.1 using the two original texture volumes and the values are given in the f columns 'model 1' and 'model 2' in Table 5.2. The accuracy of the VBV and PBP scheme has been measured by evaluating the percentage of misclassification δ by comparing the label map with the ground truth:

$$\delta = \frac{N_m}{N_{TI}}.100\%, \tag{5.10}$$

where N_m is the total number of misclassified nodes in $\mathbf{\Lambda}_I$. Computational volumes with $m=n=15$ voxels and various *depths* $d=3,5,7,9$ and 11 slices have been used. The $\mathbf{\Lambda}_B$ dimensions are $l = 7$ voxels and $p=1,2,3,4$ and 5 slices respectively (Fig. 3.7). N_{TI} is a function of d as well. In all cases the Mahalanobis distance D_M has been used.

The results for all internal nodes are summarized in Table 5.4.

Table 5.4: VBV segmentation: misclassification δ [%] and elapsed time [s], two texture problem.

Depth	**mean**	**Slice #**									**el.time**	
	δ	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**Tot.**	**Per sl.**
$d=3$	**2.57**	2.13	2.73	2.59	3.67	2.17	3.03	2.9	2.29	1.69	209.48	**23.28**
$d=5$	**1.46**	-	1.67	1.72	1.6	1.49	1.55	0.98	1.24	-	334.23	**47.35**
$d=7$	**1.21**	-	-	1.32	1.29	1.42	1.09	0.95	-	-	361.77	**72.35**
$d=9$	**1.11**	-	-	-	1.18	1.09	1.06	-	-	-	290.26	**96.75**
$d=11$	**1**	-	-	-	-	1	-	-	-	-	121	**121**

Better performance using deeper computational volumes comes with higher computational cost.

Increasing the depth of the computational volume increases the number of nodes n_T used for the LS estimates. This produces more accurate features $\mathbf{f}_t, \forall t \in \mathbf{\Lambda}_I$ and increases the classification accuracy. Since the $\boldsymbol{\theta}_h$ are similar, the accumulation

of enough data, necessary to estimate correctly the discriminant $\boldsymbol{\theta}_v$ parameters is very important. For a better view of the effect of the depth in the VBV scheme, consider the middle slice #6 of the volume. When d increases, the misclassification δ decreases as shown in Fig. 5.5.

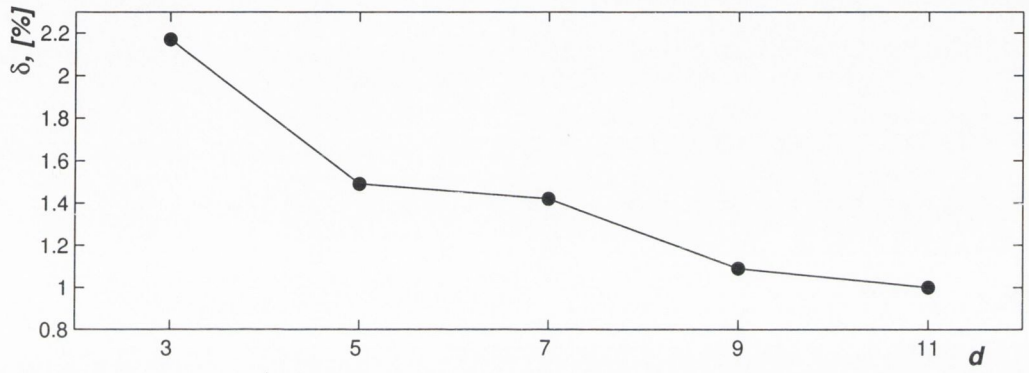

Figure 5.5: VBV segmentation accuracy for the two texture problem: slice #6.

The label maps shown in Fig. 5.6 illustrate the perceptual improvement in the segmentation with d.

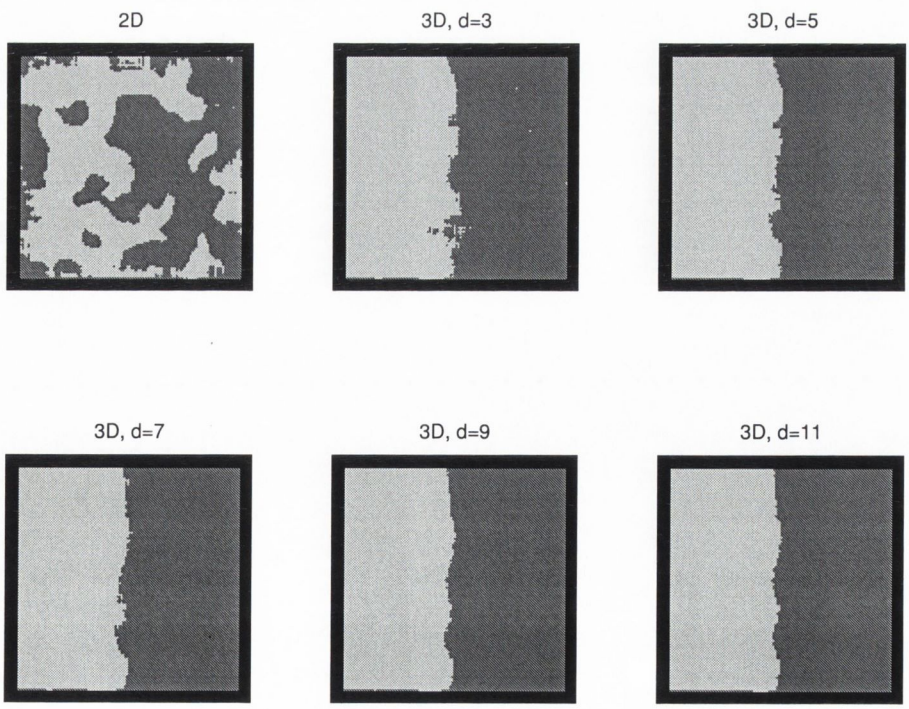

Figure 5.6: VBV segmentation results for the two texture problem: slice #6.

The key requirement for 3-D rather than 2-D modelling of 3-D GMRF mixtures is therefore underlined, in those cases where the key discriminant is $\boldsymbol{\theta}_v$. The 2-D PBP scheme works with a 2-D GMRF model, i.e. only the parameters μ, σ and the correlation parameters corresponding to the 2-D neighbourhood geometry estimated from the 2-D data are used. For a second order GMRF these are the same as $\boldsymbol{\theta}_h$ (see Figures 3.5 and 3.6). This independent classification of slice #6 fails ($\delta = 29.9\%$). The PBP execution time was 4.12 s. The segmentation results for all slices are given in Figures D.1 -D.3, Appendix D.

5.4.2 A three textures problem

Three stationary 3-D GMRFs of order $q = 3$ have been generated as previously described. The actual parameter values and their estimates used for codebook entries are given in Table 5.3. The data volumes are given in Figures C.4 - C.6, Appendix C. Similar values have been chosen for μ, σ and $\boldsymbol{\theta}_h$ for the three models and different values for $\boldsymbol{\theta}_v$ (3.11).

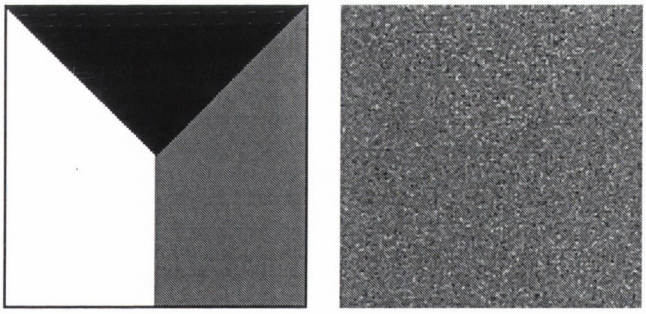

Figure 5.7: Ground truth mask and grey-level data, three texture problem: slice #6.

The ground truth mask and the synthetic data for the middle slice #6 are shown in Fig.5.7 and those for the whole volume are given in Fig. C.8.

The operational settings are the same as for the two texture problem. The misclassification δ rates and the elapsed times for all processed slices are reported in Table 5.5.

The same trend as before has been observed. For slice #6 this trend is depicted in Fig. 5.8.

Table 5.5: VBV segmentation: misclassification δ [%] and elapsed time [s], three texture problem.

Depth	mean	Slice #									el.time	
	δ	2	3	4	5	6	7	8	9	10	Tot.	Per sl.
$d = 3$	**10.38**	8.73	10.46	10	10.69	8.62	10.75	8.93	11.15	14.13	343.09	**38.12**
$d = 5$	**4.37**	-	4.79	4.56	4.12	4.59	3.89	3.52	5.12	-	535.23	**76.46**
$d = 7$	**2.55**	-	-	2.65	2.77	2.49	2.15	2.7	-	-	584.47	**116.89**
$d = 9$	**2.11**	-	-	-	2.11	2.06	2.16	-	-	-	475.09	**158.36**
$d = 11$	**2.23**	-	-	-	-	2.23	-	-	-	-	204.49	**204.49**

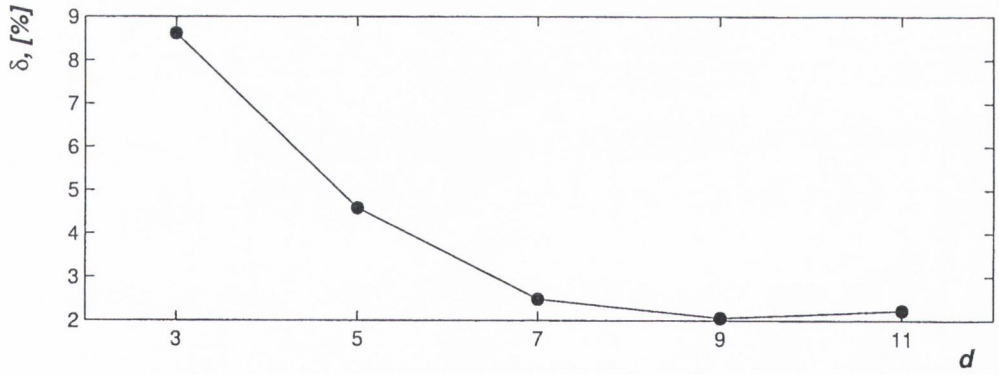

Figure 5.8: VBV segmentation accuracy for the three texture problem: slice #6.

It should be noted that the values of δ increase with the number of textures present. The label maps for the central slice are shown in Fig. 5.9 for various d. The 2-D PBP result took only 2.3 s.c̃ompared with the elapsed times for the 3-D VBV classification (Table 5.5), but the performance of the former is poor, with $\delta = 42.8\%$.

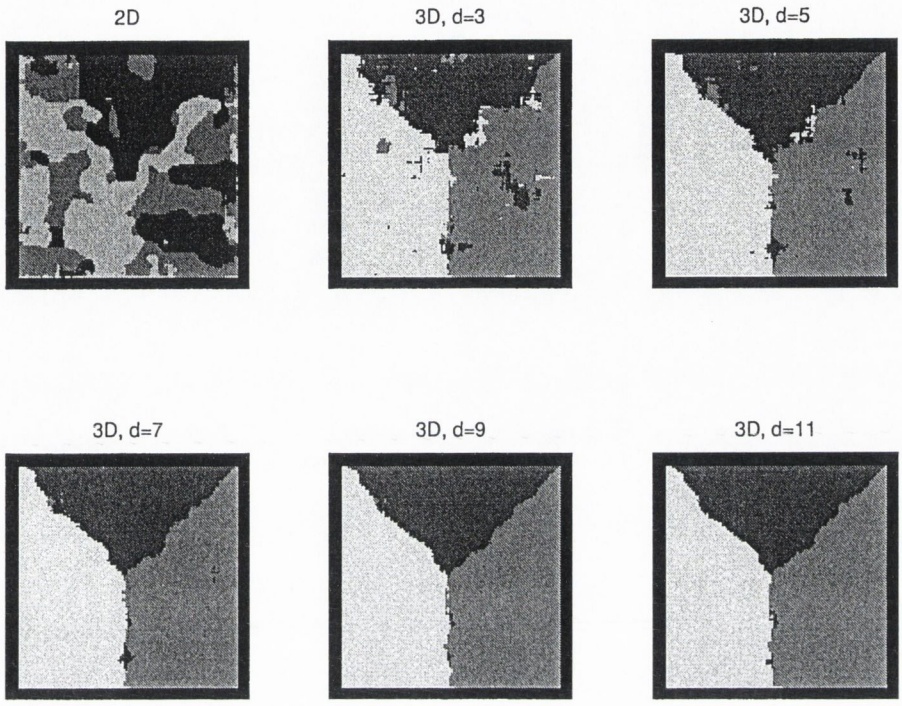

Figure 5.9: VBV segmentation results for the three texture problem: slice #6.

It can be noted that the perceptual quality of the segmentation maps increases with d.

The segmentation results for all slices are given in Figures D.4 -D.6.

5.4.3 VBV limitations

The same general trends have been observed in many simulation experiments of the kind reported above. To summarize, the main conclusions from the simulations were:

1. The 3-D VBV scheme is superior to the 2-D independent PBP segmentation of slices from 3-D data. This is evident in the case where the key discriminant between different classes are the $\boldsymbol{\theta}_v$ parameters. This is a good motivation for 3-D modelling of volumetric images, where it is supposed that there are similar $\boldsymbol{\theta}_h$ and different $\boldsymbol{\theta}_v$.
2. When the GMRF stationarity is preserved within ω_t, the classification accuracy increases by using deeper computational volumes.

3. The VBV scheme is much slower than the PBP one, and its computational burden increases with greater depth.

The three principal limitations of the VBV scheme are:

1. The texture stationarity assumption within ω_t. It does not often hold especially when the volume overlaps borders between textures.

2. The absence of regularization to encourage *connectivity* in the inferred label field.

3. The large computational load of obtaining the LS estimates at every voxel in the volume.

The first two limitations can be illustrated by comparing the visual quality of the classification results. Some label maps for different slices and for depths $d = 3$ and $d = 5$ are illustrated in Fig. 5.10. It can be seen that despite the relatively small values of δ (Table 5.5), the segmentation at the changepoint areas (borders between textures) is not satisfactory, even though it improves with greater depths. Note, however, that seeking to exploit this improvement by increasing d makes the computational burden even more prohibitive (Tables 5.4 and 5.5).

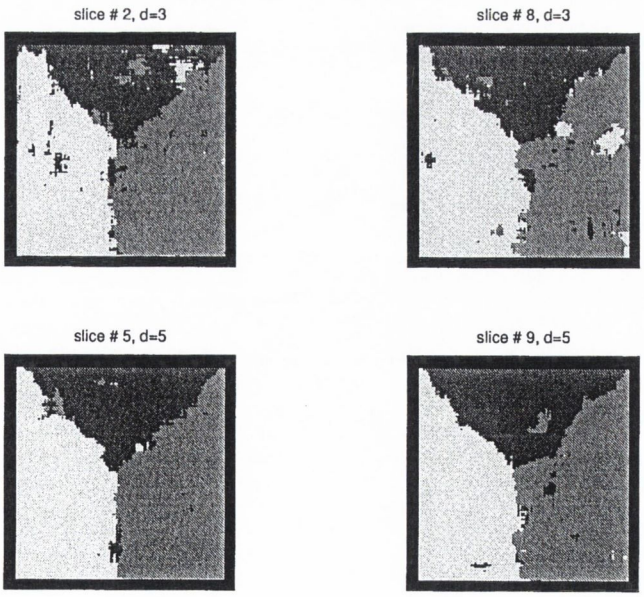

Figure 5.10: Poor VBV segmentation results for the three texture problem.

5.5 CONCLUSIONS

In this chapter a generalization of the LS parameter estimation technique has been developed for 3-D GMRFs. A simple computational volume-based approach has been proposed. The supervised segmentation is performed in two stages: the off-line codebook generation and the classification of nodes in VBV manner. The proposed method had shown superior performance in comparison to independent PBP segmentation of slices of a GMRF mixture data. This has been specifically pointed out for the case when the $x - y$ plane correlations were not enough to discriminate between the textures and use of the z-going correlations is the key requirement for obtaining meaningful segmentation.

The VBV algorithm, however, has several limitations and thus the resulting label fields do not satisfy the criteria for good segmentation. Because no constraint encouraging connectivity and smoothness in $\mathbf{L}$ has been imposed *a priori*, in comparison to the Bayesian methods, these have to be applied *a posteriori*. Methods for overcoming these limitations are considered in the following chapters.

Chapter 6

SEGMENTATION ON DECIMATED LATTICES

6.1 INTRODUCTION

The task of finding homogeneous texture regions in non-homogeneous 3-D data has been defined as labelling of nodes from a mixture of stationary GMRFs. This is the approach taken in this work to the difficult problem of texture segmentation of volumetric data. The VBV scheme (Section 5.4) presents a solution to this problem, but due to its limitation produces label field of unsatisfactory quality with high computational cost. Simultaneous quality and speed improvements can be achieved via filtering using the majority principle over decimated lattices geometry.

In Section 6.2 some criteria for evaluation of what is a 'good' segmentation are given. Different segmentation algorithms based on the VBV scheme followed by a majority filtering are presented in Section 6.3. Results from segmenting synthetic and more realistic 3-D data are reported. A comparison between the performance of the segmentation algorithms is the subject of Section 6.4.

In most real images the acquisition process and the nature of the 3-D object of interest, are sources of rapid change of the borders between textures along the z-

going dimension, therefore it requires special attention. The effect of this phenomena on the performance of the segmentation algorithms is studied in Section 6.5. Finally, the main conclusions of the chapter are given in Section 6.6.

6.2 3-D TEXTURE SEGMENTATION

As explained in Section 2.3 there are formal definitions of segmentation and some criteria for segmentation evaluation in the literature. Pavlidis [63] states that the segmentation process partitions Λ into disjoint nonempty subsets of nodes. These labelled subsets are the segmented regions. Image features such as grey level intensity or colour (multispectral data), motion, texture, etc. are used to form these regions. This work is concerned with segmentation of data, existing on 3-D lattices, based on their textural properties and will be referred to as *3-D texture segmentation*. The problem of segmentation is the inference of the hidden label field $\mathbf{L}$ for each node given the original grey-level data $\mathbf{X}$. The result of the segmentation tessellates the lattice into labelled regions. It is consistent with the definition above as all the conditions are satisfied.

Because of the lack of strict evaluation criteria and a unifying segmentation theory, the human observer must be the final judge of the quality of the segmentation result. Haralick and Shapiro [36] have established guidelines for a 'good' segmentation result in their survey of image segmentation techniques. According to them the segmented regions should obey the following requirements:

1. They should be uniform and homogeneous with respect to some characteristic such as grey tone or texture.
2. Region interiors should be simple and without many small holes.
3. Adjacent regions should have significantly different values with respect to the characteristic on which they are uniform.
4. Their boundaries should be simple and must be accurate.

In summary, these rules require the label field to be smooth and connected.

The label field produced by the VBV algorithm (Section 5.4) do not satisfy these requirements. Therefore, to have a proper segmentation algorithm, the inference of the label field should be *regularized*, i.e. spatial connectivity constraints should be imposed on it. This can be done *a priori* or *a posteriori*.

Very often the MRF-based segmentation is tackled as a Bayesian inference problem (Section 2.2.6). *A priori* knowledge about the image and assumptions about label field smoothness are encoded in the prior. The maximization of the posterior probability, i.e. the MAP estimation of $\mathbf{L}$ is achieved using algorithms such as Stimulated Annealing (SA) [49,59], Iterated Conditional Modes [6] and Expectation Maximization [29,84].

In this work an alternative approach is used. The connectivity of the label field realization, is not achieved *a priori*, but after the voxel classification (the Majority decision (MD) algorithm presented in Section 6.3.1) or incorporated within the classification process (the Unanimity Rule (UR) and the Recursive Unanimity Rule (RUR) algorithms presented in Sections 6.3.2 and 6.3.3). These algorithms impose connectivity while improving computational efficiency by performing calculations only at selected nodes from Λ.

6.3 ALGORITHMS ON DECIMATED LATTICES

Several algorithms based on the VBV scheme are presented. These algorithms elicit connectivity on $\mathbf{L}$ regardless of the labelling algorithm applied at each processed node. This is achieved by exploiting the decimated lattice geometry (Section 3.2.2). It leads to computational speedup as the number of nodes to be processed reduces greatly.

They are an extension of the methods for 2-D texture segmentation presented in [77]. For the purpose of 3-D texture segmentation 3-D versions were implemented with some modifications taking into account the specifics of the 3-D lattice geometry.

6.3.1 Majority Decision (MD) Segmentation

In this scheme, a simple nonlinear majority decision filter is applied to the label field inferred by the VBV scheme.

The decimated MD lattice $\mathbf{\Lambda}_{MD}$ is specified by the parameters n^d_{MD} and d^d_{MD} (Fig. 3.8). To reassign a label to the current node t, consider six nodes, four of which are located in the same slice as $t = (i, j, k)$, i.e. with coordinates $(i \pm n^d_{MD}/2, j \pm n^d_{MD}/2, k)$, and two located at chosen neighbouring slices, namely, the nodes $(i, j, k \pm d^d_{MD}/2)$. These nodes are denoted in grey and t in black in Fig. 6.1.

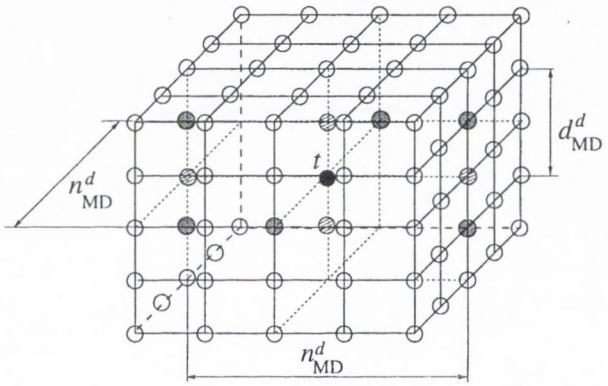

Figure 6.1: The MD lattice.

The label l_t takes the value of the majority of the labels at these six nodes. If there is ambiguity about the value of l_t, four additional nodes at locations $(i \pm n^d_{MD}/2, j, k)$ and $(i, j \pm n^d_{MD}/2, k)$ are considered. They are denoted by hashed circles in Fig. 6.1. If there is still some ambiguity, the final label decision is taken by choosing randomly between the most frequent labels among all nodes from $\mathbf{\Lambda}_{MD}$. The algorithm can be summarized as follows:

Algorithm :

1. For every node $t = (i, j, k)$ consider 6 nodes, located at $i \pm n^d_{MD}/2, j \pm n^d_{MD}/2, k)$ and at $(i, j, k \pm d^d_{MD}/2)$.
2. Assign to l_t the value of the majority of the labels at those nodes.
If no majority, consider 4 additional nodes $(i \pm n^d_{MD}/2, j, k), (i, j \pm n^d_{MD}/2, k)$.

Repeat step 2.

3. If no majority, assign l_t randomly from among the jointly most frequently inferred labels from these 10 nodes.

The MD algorithm strongly encourages spatial connectivity in the label field and reduces the imprecision caused by the presence of a border between two textures in the same computational volume ω_t, as simulations will now verify.

Consider MD segmentation of two slices of the volume from Fig. C.7, Appendix C. These are compared to the respective VBV results in Fig. 6.2. The results for the whole data set are given in Section D.2, Appendix D.

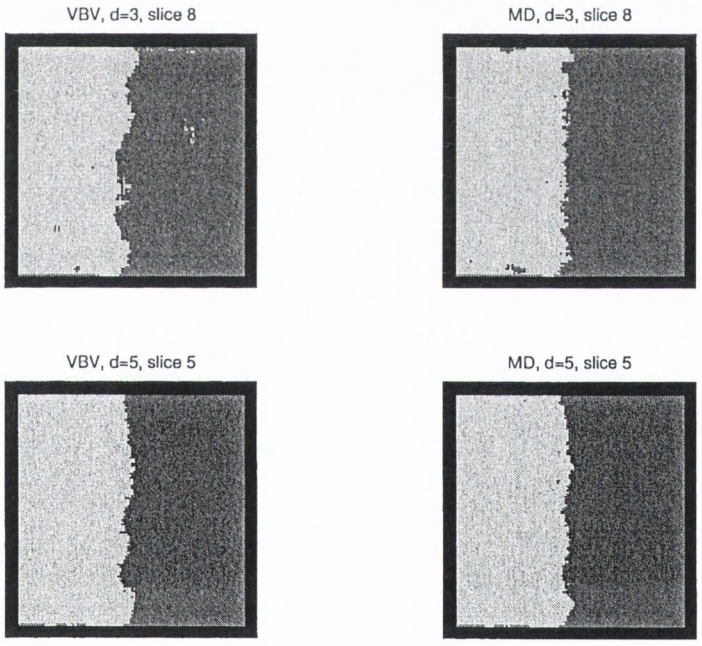

Figure 6.2: MD vs. VBV segmentation for the two texture problem.

The MD algorithm produced better region boundaries and less isolated voxels in the label maps than the VBV.

6.3.2 Unanimity Rule (UR) Segmentation

This strategy is closely related to the MD approach. Here, the unanimity of the labels on the decimated UR lattice Λ_{UR} is checked at the time of classification,

rather than as a post-processing procedure as with the MD. Λ_{UR} is specified by the parameters n_{UR}^d and d_{UR}^d as illustrated by the schematic in Fig. 6.3. The processed node t is denoted in black and the corners of the UR unit volume in grey.

Algorithm :

1. Evaluate the labels at nodes located at the corners of a volume of size $n_{UR}^d \times n_{UR}^d \times d_{UR}^d$ centred around the currently processed node t.
2. If all 8 labels are unanimous, assign the same label to *all* interior nodes for that volume.
3. Else use the VBV scheme (Section 5.4).

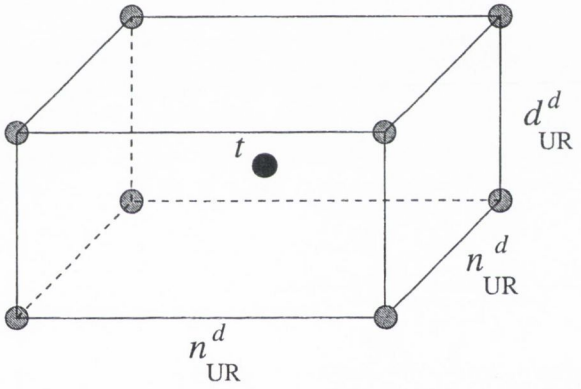

Figure 6.3: The UR lattice.

The UR algorithm segments homogeneous regions quickly, since unanimity will often be observed at the vertex nodes in such regions. In such cases, all further voxel classification is bypassed by the algorithm, resulting in a major computation saving.

Simulation results

To illustrate the performance of the algorithm the same data sets from Section 5.4 pertaining to the VBV scheme were segmented. The Λ_{UR} parameters used were $n_{UR}^d = 16$ and $d_{UR}^d = 4$. The performance parameters, namely the misclassification, computed via (5.10), and the elapsed times, for the two texture problem are reported in Table 6.1.

Table 6.1: UR segmentation: misclassification δ [%] and elapsed time [s], two texture problem.

Depth	mean	Slice #									el.time	
	δ	2	3	4	5	6	7	8	9	10	Tot.	Per sl.
$d = 3$	**2.15**	2.04	2.24	2.29	3.11	2.02	2.23	2.62	1.61	1.2	33.77	**3.75**
$d = 5$	**1.46**	-	1.67	1.72	1.6	1.49	1.55	0.98	1.24	-	125.84	**17.98**
$d = 7$	**1.21**	-	-	1.32	1.29	1.42	1.09	0.95	-	-	54.99	**10.99**
$d = 9$	**1.11**	-	-	-	1.18	1.09	1.06	-	-	-	266.55	**88.65**
$d = 11$	**1**	-	-	-	-	1	-	-	-	-	111.37	**111.37**

The same conclusions about the effect of the computational volume depth can be made as with the VBV algorithm, because in some cases they are equivalent (see step 2. of the UR algorithm).

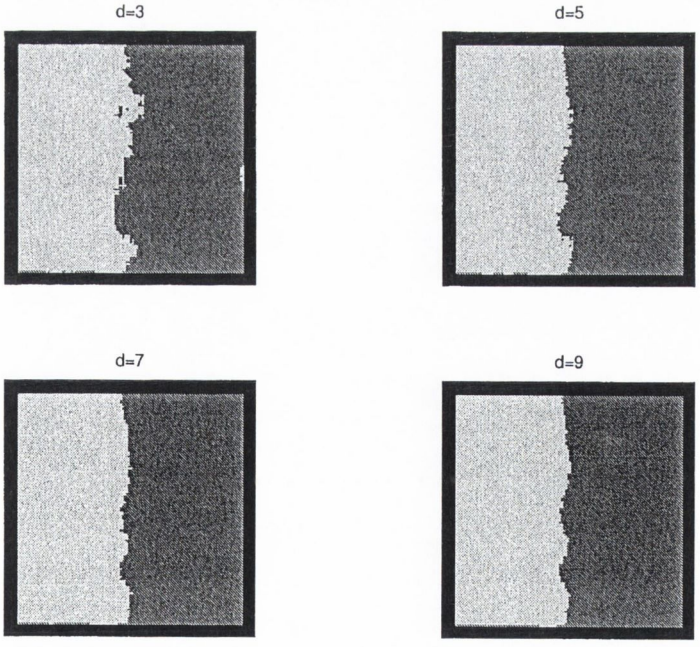

Figure 6.4: UR segmentation for the two texture problem: slice #5.

The deeper the computational volume, the better the segmentation, but the onger the computational time is. The visual improvement of the segmentation

result with depth is illustrated in Fig. 6.4. The results for the whole volume are given in Section D.3.

The expected dependence on d has been observed for the three texture problem as well. The performance parameters are given in Table 6.2. This is further illustrated in Fig. 6.5 for the fifth slice of the volume.

Table 6.2: UR segmentation: misclassification δ [%] and elapsed time [s], three texture problem.

Depth	mean	Slice #									el.time	
	δ	2	3	4	5	6	7	8	9	10	Tot.	Per sl.
$d=3$	**7.47**	7.72	6.93	6.14	7.21	7.7	6.69	4.58	7.38	12.85	123.25	**13.7**
$d=5$	**3.84**	-	4.45	3.61	3.2	3.43	3.52	3.52	5.12	-	239.75	**34.25**
$d=7$	**2.44**	-	-	2.6	2.49	2.31	2.11	2.7	-	-	95.29	**19.06**
$d=9$	**2.11**	-	-	-	2.11	2.06	2.16	-	-	-	443.2	**147.73**
$d=11$	**2.23**	-	-	-	-	2.23	-	-	-	-	185.02	**185.02**

The poor performance in the case $d = 3$ both for the two and three class problems is due to the minimal ω_t depth, which in this case is even smaller than d^d_{UR}.

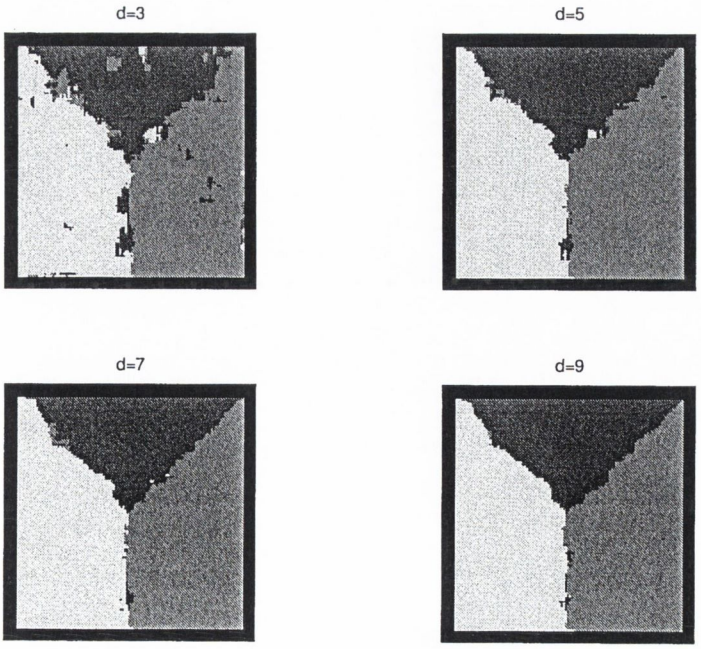

Figure 6.5: UR segmentation for the three texture problem: slice #5.

6.3.3 Recursive Unanimity Rule (RUR) Segmentation

A further improvement of the UR idea has resulted in the development of a fast recursive algorithm, called the Recursive Unanimity Rule (RUR). The initial lattice Λ_{RUR} is determined by the parameters n^d_{RUR} and d^d_{RUR} as shown in Fig. 6.6. These must each be radix-2, because on each iteration there is a division of the dimensions of the RUR volume by a factor of 2.

Initially the eight vertices of the RUR unit volume are tested for unanimity, in which case the UR policy is adopted. If unanimity fails, the volume is subdivided into eight sub-volumes by splitting each dimension in 2. Then, the RUR scheme is applied recursively for each subvolume, until unanimity is reached or a single node is left. In many real applications, such as in medical imagery, the sampling rate along the z-going dimension is smaller than within the $x-y$ plane. This is why the choice of parameters satisfies $d^d_{RUR} < n^d_{RUR}$. If any RUR dimension reaches 1, the RUR scheme is applied in the remaining two dimensions.

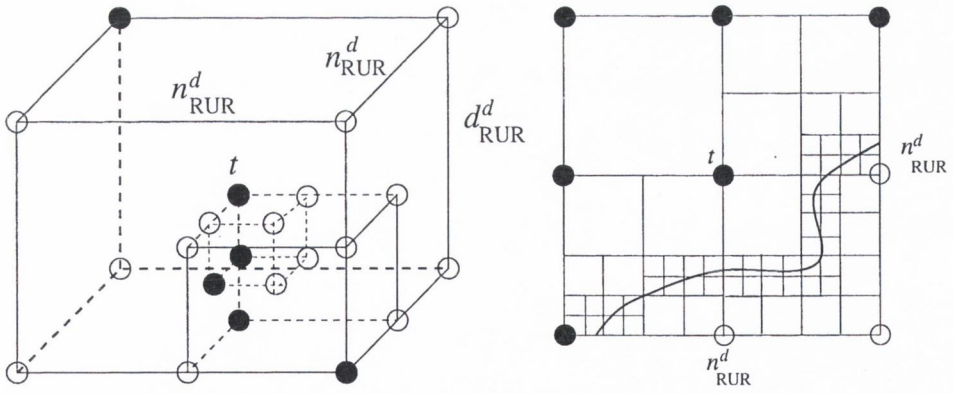

Figure 6.6: RUR algorithm in 3-D and 2-D respectively.

The algorithm is illustrated in Fig. 6.6 where the 'black' nodes have different labels than the 'white' nodes. This is the typical case when a recursion is needed, since the border between two 3-D textures falls within the RUR volume. The 3-D recursion is shown on the left of the figure, while the 2-D recursion is illustrated on the right.

If the n^d_{RUR} and d^d_{RUR} parameters are selected too large, there is a danger of missing some small 3-D structures. Therefore, if *a priori* knowledge is not available to assist the selection of these parameters, the issue can be tackled by small-step shifts of the entire RUR computational grid, and comparison of the segmentations yielded for each such grid. This, however, significantly increases the computational expense of the algorithm, and is found to be unnecessary in many practical instances.

Algorithm :

1. If $n^d_{RUR} = 1$ and $d^d_{RUR} = 1$ classify the single node as in VBV (Section 5.4). Stop.
2. Classify the 8 vertices of the current RUR volume of size $n^d_{RUR} \times n^d_{RUR} \times d^d_{RUR}$. If all 8 labels are the same, assign to *all* internal nodes within the volume the same label.
3. Set $n^d_{RUR} = n^d_{RUR}/2$. If $d^d_{RUR} \neq 1$ set $d^d_{RUR} = d^d_{RUR}/2$. Goto step 1.

This algorithm achieves significant speedup as homogeneous regions are processed very quickly (step 3.). The recursion is invoked only around the borders between homogeneous regions, therefore the lattice decimation rates adapt dynamically. The algorithm improves the spatial connectivity in the label field and the region borders better than the UR scheme because it invokes the VBV scheme at a finer scale or at single nodes.

Simulation results

Again, the performance of the RUR algorithm has been tested on the datasets from Section 5.4 as for the VBV and UR schemes. The Λ_{RUR} parameters used were $n^d_{RUR} = 16$ and $d^d_{RUR} = 4$. For the two texture problem, the performance parameters for each slice and for different depths of ω_t are given in Table 6.3. As before, increasing d decreases δ and increases the elapsed time.

The label maps for different depths for the seventh slice of the data volume are given in Fig. 6.7. All segmentation results are presented in Section D.4.

The performance parameters for the three class problem, for all slices and depths,

are given in Table 6.4.

Table 6.3: RUR segmentation: misclassification δ [%] and elapsed time [s], two texture class problem.

Depth	mean	Slice #									el.time	
	δ	2	3	4	5	6	7	8	9	10	Tot.	Per sl.
$d = 3$	**2.11**	2.04	2.24	2.27	3.1	2.08	2.19	2.67	1.35	1.1	9.51	**1.06**
$d = 5$	**1.49**	-	1.61	1.77	1.65	1.55	1.65	1.02	1.22	-	13.69	**1.96**
$d = 7$	**1.2**	-	-	1.32	1.28	1.44	1.07	0.92	-	-	13.92	**2.78**
$d = 9$	**1.09**	-	-	-	1.17	1.06	1.04	-	-	-	13.61	**4.54**
$d = 11$	**0.98**	-	-	-	-	0.98	-	-	-	-	29.61	**29.68**

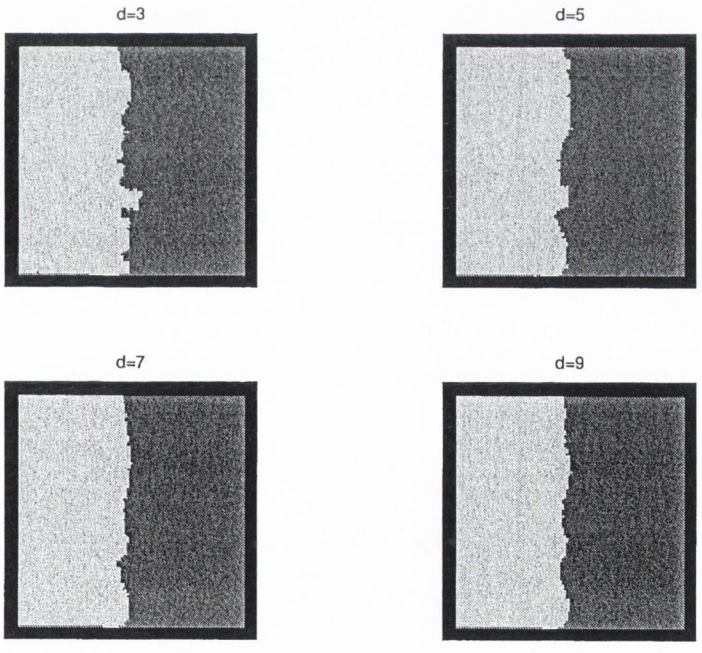

Figure 6.7: RUR segmentation for the two texture problem: slice #7.

In a similar manner, the segmentation results for the three texture problem for different depths of the seventh slice are given in Fig. 6.8.

The same observations about the impact of the ω_t depth on the quality of the label field can be made as for the two texture problem.

Table 6.4: RUR segmentation: misclassification δ [%] and elapsed time [s], three texture problem.

Depth	mean	Slice #									el.time	
	δ	2	3	4	5	6	7	8	9	10	Tot.	Per sl.
$d = 3$	**7.25**	6.89	6.53	6.12	6.84	6.26	7.49	7.12	7.51	10.48	41.21	**4.58**
$d = 5$	**3.73**	-	4.17	3.85	3.58	3.65	3.1	3.2	4.56	-	44.87	**6.41**
$d = 7$	**2.44**	-	-	2.56	2.59	2.34	2.13	2.56	-	-	39.86	**7.97**
$d = 9$	**2.08**	-	-	-	2.06	2.06	2.14	-	-	-	32.94	**10.98**
$d = 11$	**2.21**	-	-	-	-	2.21	-	-	-	-	54.851	**54.85**

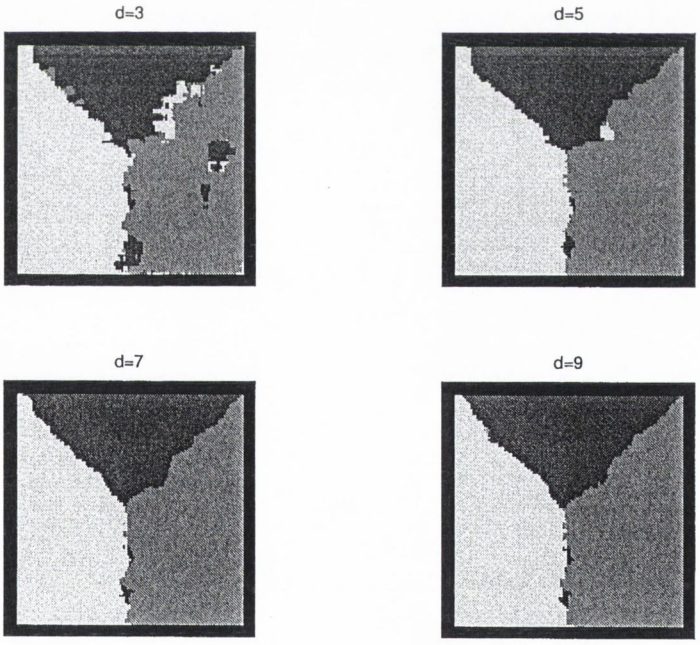

Figure 6.8: RUR segmentation for the three texture problem: slice #7.

6.4 SIMULATIONS

The MD segmentation obtained slightly better label maps than the VBV as illustrated in Section 6.3.1. It is a post-processing step, hence, the overall processing

time increased. Therefore, it is relevant to compare the UR and RUR algorithms with the VBV. Numerous experiments were designed and carried out with synthetic and more realistic data.

6.4.1 Synthetic Data

For the two texture problem, the UR and RUR algorithms improve the classification accuracy slightly, when compared to the VBV technique, but also have the added benefit of achieving a significant speedup. This is shown in Figures 6.9 and 6.10, where the performance parameters for the middle slice of the data are depicted as functions of d. The RUR algorithm gives the best performance.

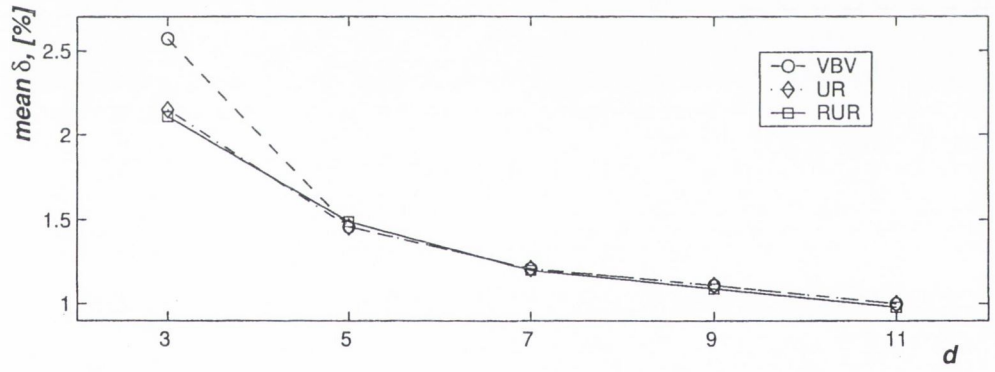

Figure 6.9: Comparison of the mean misclassification δ for VBV, UR and RUR, two texture problem.

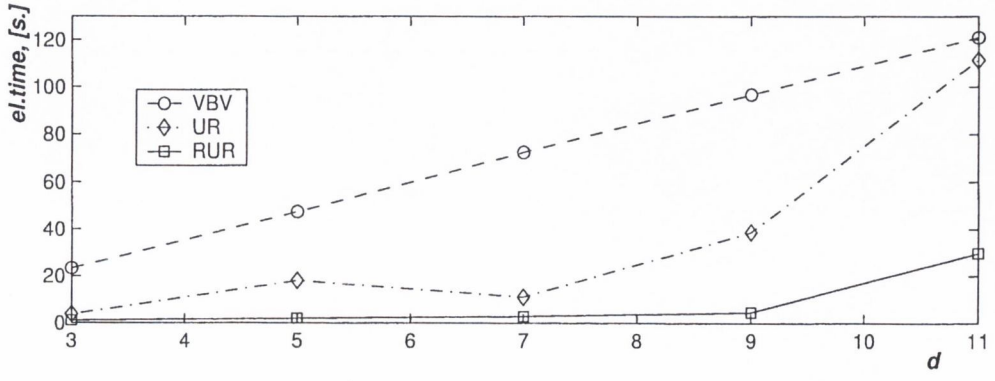

Figure 6.10: Comparison of the elapsed time per slice for VBV, UR and RUR, two texture problem.

For a visual illustration of how the limitations of the VBV scheme were overcome by the RUR scheme, consider Fig. 6.11 for the two texture and Fig. 6.14 for the three texture problem respectively. It can be seen how the amount of isolated 'holes' and the imprecision in the label maps were reduced by the usage of the RUR scheme.

The performance parameters for the three texture problem are illustrated in Figures 6.12 and 6.13, where the same conclusions can be drawn.

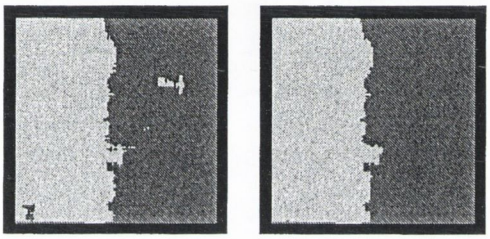

Figure 6.11: VBV vs. RUR segmentation, two texture problem, slice #7, $d = 3$.

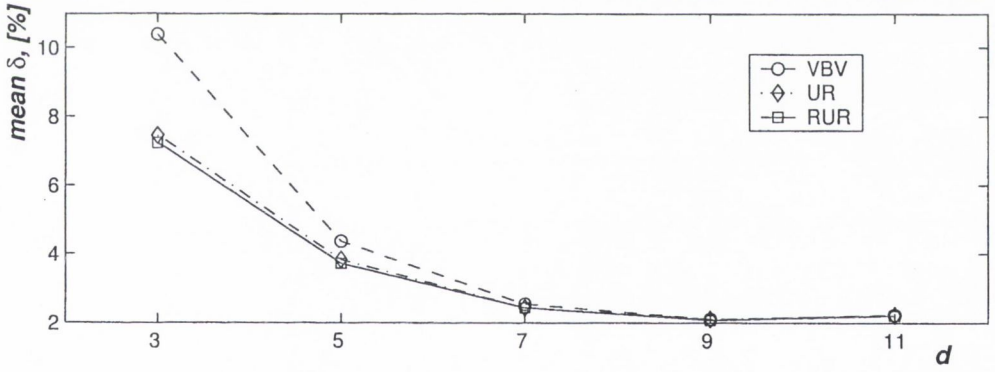

Figure 6.12: Comparison of the mean misclassification δ for VBV, UR and RUR, three texture problem.

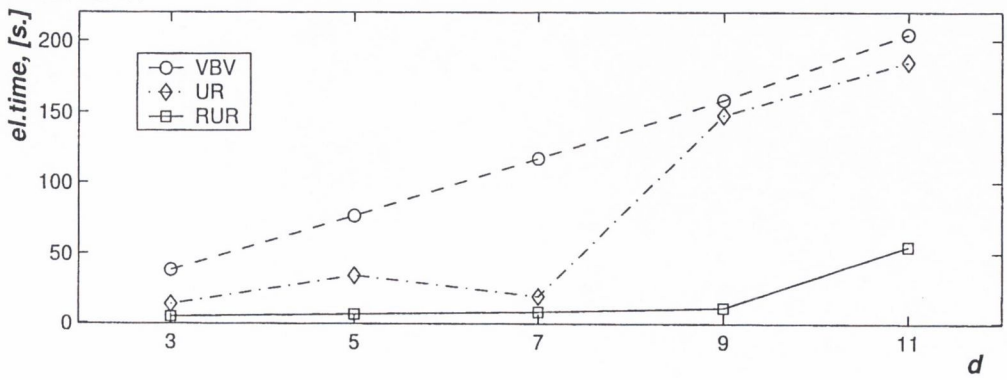

Figure 6.13: Comparison of the elapsed time per slice for VBV, UR and RUR, three texture problem.

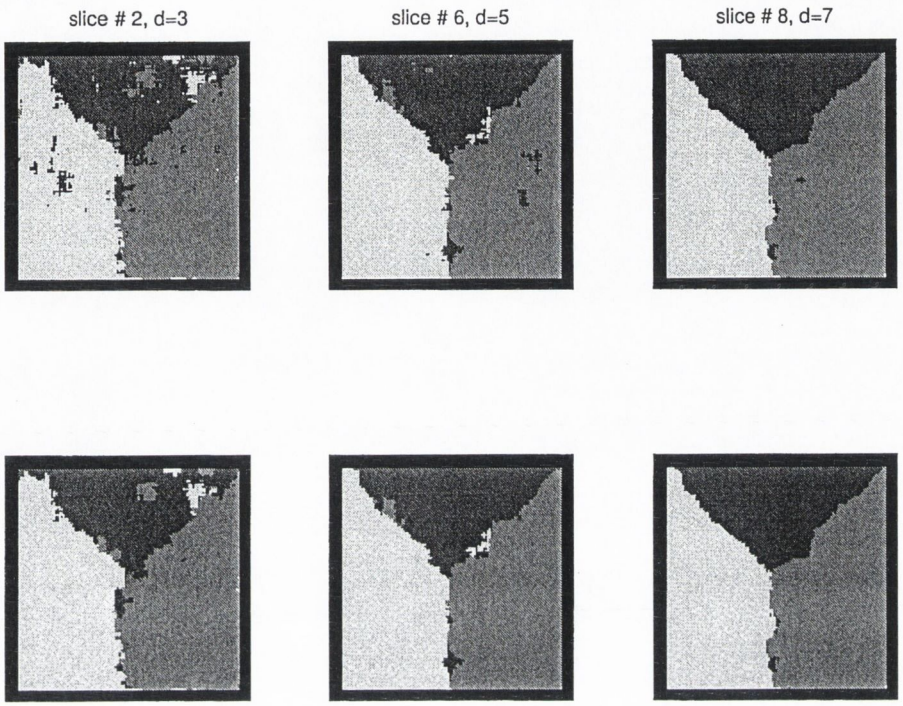

Figure 6.14: VBV (1st row) vs. RUR (2nd row) segmentation, three texture problem.

The experimental examples presented are illustrative of all simulations performed. The results confirmed the expected behaviour of the algorithms and the following conclusions can be made:

1. The 3-D segmentation gives superior classification accuracy in comparison to the independent slice by slice 2-D segmentation but incurs a greater computational load. Thus, the usage of a 3-D method is justified only when the 2-D segmentation is not performing satisfactorily.

2. The 3-D segmentation algorithms are the only reasonable methods to use when the stationary GMRFs possesses similar $\boldsymbol{\theta}_h$ (3.9) and differ in their correlations in the z-going dimension $\boldsymbol{\theta}_v$. In this case, the classification accuracy increases by increasing the depth of the computational volume.

3. The MD, UR and RUR segmentation algorithms are superior to the VBV scheme (Section 5.4). The label map produced by the RUR algorithm satisfies the criteria for 'good' segmentation and it is the fastest method. Hence, it is the main algorithm used in the remainder of this thesis.

6.4.2 Temporal Textures

More realistic datasets have been constructed using the MIT natural temporal textures database [79]. As explained in [83], temporal textures are image sequences of textures with motion, such as wavy water, rising steam and fire. They are also called dynamic textures. In this case, the third dimension of the data is the temporal dimension. Although in this respect they are similar to video sequences, there is no camera or object motion, therefore they can be treated as volumetric imagery.

The two temporal textures used were the 'river' ('wavy water from far') as a background (Fig. C.16) and the 'plastic'('plastic sheet waving in the wind') in the foreground (Fig. C.15). A synthetic mask is used to combine these temporal textures, to produce a sequence of size $84 \times 84 \times 11$, as shown in Fig. C.17.

After the 2-D parameters have been estimated independently for each slice (or sequence frame), 2-D RUR segmentation has been performed. Second order 2-D GMRF modelling has been used with computational window of size 15×15 and RUR parameter $n^d_{RUR} = 16$. The resulting label maps for all slices are given in Fig. D.22. It can be observed that the method was unable to distinguish between the two temporal textures. The poor performance can be explained by the fact that the textures are characterized by similar $\boldsymbol{\theta}_h$ (3.9) values as shown in Table 6.5. For the second order 3-D GMRFs these are the same set as $\boldsymbol{\theta}$ for the 2-D GMRF. The table gives all the parameter values $\boldsymbol{\theta}$ for the 3-D GMRF model.

Table 6.5: Estimated parameters for the temporal textures, second order 3-D GMRF.

texture	μ	σ	θ_0	θ_1	θ_2	θ_3	θ_4	θ_5	θ_6	θ_7	θ_8
'plastic'	129.3	3.36	0.212	0.485	0.376	−0.106	−0.76	−0.022	−0.025	−0.128	−0.226
'river'	132.52	6.07	0.23	0.524	0.17	−0.126	−0.14	0.015	0.0067	−0.04	−0.14

The distances between the textures are $D_{NE} = 0.1626$ when only the $\boldsymbol{\theta}_h$ were considered and $D_{NE} = 0.2259$ where the full set $\boldsymbol{\theta}$ was used. The latter value is greater due to the difference in the $\boldsymbol{\theta}_v$ parameters for the two textures.

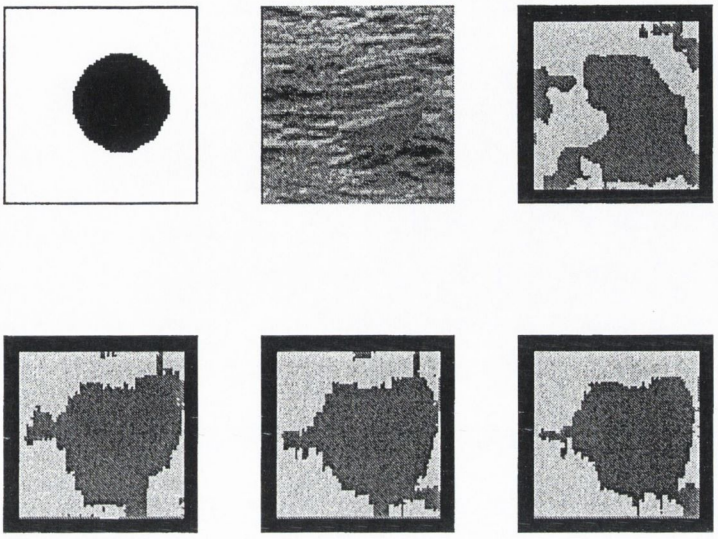

Figure 6.15: Ground truth mask, slice #8 from the sequence and 2-D RUR segmentation (1st row). 3-D RUR segmentation results for $d = 3$, 5 and 7 (2nd row).

The 3-D RUR segmentation with $n^d_{RUR} = 16$, $d^d_{RUR} = 4$ and $\omega_t = 15 \times 15 \times d$ is illustrated in Figures D.23, D.24 and D.25 for different computational volume depths. Although the results are not as good as the ones obtained with the synthetic textures, the 3-D segmentation algorithm is able to discriminate between the two textures unlike the 2-D independent segmentation. The dependence of the classification accuracy on the computational volume depth is confirmed as well (Fig.6.15).

6.5 SHAPE EVOLUTION ALONG THE z-GOING DIMENSION

So far, the synthetic examples used the same 2-D mask over each volumetric slice to produce the non-stationary data. This implies that the 2-D cross-section of a 3-D homogeneously textured object remains unchanged. However, this is not true with most of real volumetric or image sequence imagery. For instance, the sampling rates in MRI scans along the $x - y$ plane and the one along the z-going dimension

(perpendicular to the cross-sectional plane) are often different as explained in Section 2.5. When, furthermore, the anatomical peculiarities of the scanned object are considered, two consecutive data slices can differ significantly due to the shape evolution of some anatomical structures along the z-going dimension. In contrast, in the case of an image sequence, shape changes between consecutive frames are often due to camera or object motion.

The success of the 3-D segmentation methods above depends on the validity of the texture stationarity assumption within the computational volume. The experiments show that an important requirement for good parameter estimation which is the basis for good classification and segmentation is a data sample of sufficient size. It has been concluded that increasing the depth of the computational volume increases the segmentation accuracy by providing a sufficiently large stationary computational volume ω_t. However, the presence of rapidly evolving cross-sections in a homogeneously textured object greatly endangers these assumptions.

To illustrate the effect of the shape evolution along the z-going dimension, consider a data volume of size $128 \times 128 \times 11$, obtained using the first two stationary models in Table 5.2. A circular mask has been used to generate a 3-D 'skewed' cylinder embedded into a background. This effect was achieved by a constant displacement of the mask from slice to slice. In this simulation, the quantitative measurement of the *difference* between the data slices is known, and can be used to *compensate* for this effect. The segmentation results without and with the shape evolution compensation are shown in Figure 6.16.

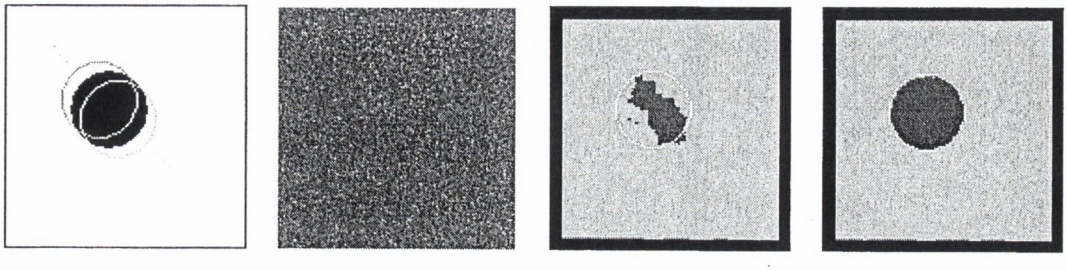

Figure 6.16: Ground truth generation mask, slice #6 from synthetic data set, 3-D segmentation results without and with compensation for shape evolution, $d = 11$.

6.6 CONCLUSIONS

The decimated lattices segmentation methods presented in this chapter are an alternative to the formal modelling of spatial dependency via an *a priori* probability density, which would be used in a formal Bayesian segmentation. The advantage is that the results are obtained for much smaller computational cost. The development of these algorithms constitutes a contribution to the collection of existing 3-D segmentation methods.

The 3-D RUR algorithm performs well in segmenting synthetic GMRF mixtures and more realistic 3-D data sets. This is evident especially when the use of $\boldsymbol{\theta}_v$ is of a key importance.

Finally, the big impact that the shape evolution along the z-going dimension on the 3-D segmentation algorithm performance was shown. It is very important to preserve the texture stationarity assumption for the computational volume, the segmentation is based on. Therefore, various novel techniques for estimating and compensating for this effect are the subject of the following chapters.

Chapter 7

COMPENSATION OF THE COMPUTATIONAL VOLUME VIA DIFFERENCE ESTIMATION

7.1 INTRODUCTION

As explained in Section 6.5, the shape evolution of a 3-D object cross-section along the z-going dimension induces non-stationarity in ω_t. It is manifested as differences between consecutive 2-D slices of the 3-D volume.

This chapter considers the problem of estimating and compensating for this effect. It is a problem similar to those of object tracking, motion estimation and image registration, as explained in Section 2.4. In these applications, typically, two 2-D frames or slices are used in the estimation of the difference. Such a method cannot be used directly for textural grey level slices as occur in textured volumetric imagery. The 3-D segmentation of minimum depth, however, produces a label field, which can be used as an input to a difference estimation procedure. Various known methods, modified to suit the properties of the label field data, can be applied to estimating the difference between a pair of 2-D slices. The methods presented in this

work, although related to existing ones, are a contribution to the area of difference analysis between slices in 3-D volumes.

In Section 7.2 the main notation needed for this chapter is introduced. What follows are various methods for solving the problem, based on estimation of the difference between two 2-D slices from the label field realization. These are: (i) Interest point detection and matching in Section 7.3, and (ii) the Cross-correlation method in Section 7.4. Finally, conclusions are drawn in Section 7.5.

7.2 MODELLING OF THE SHAPE EVOLUTION

In this section, a formal framework for modelling the shape evolution in the z-going dimension is introduced, followed by a simulation example.

7.2.1 Membership

In simulation, it is possible to measure the degree of violation of the stationarity assumption within ω_t. Consider the simple case of a homogeneously-textured 3-D object embedded in a textural background as illustrated in Fig. 7.1. Let t be the centre of the object cross-section in the middle frame, k. Denote by n_O the number of nodes within ω_t belonging to the 3-D object. Then the *membership* m_t of the 3-D object, within ω_t with total voxel count n_T is defined as:

$$m_t = \frac{n_O}{n_T}.100[\%]. \tag{7.1}$$

Ideally, the membership m_t should be 100%, i.e. the computational volume should consist entirely of nodes belonging to the same homogeneous texture object. In real data, however, this is often not satisfied even for 2-D images and the problem is more marked when dealing with 3-D images. For instance, this phenomena is well known to arise in MRI where the acquisition sampling rate between consecutive slices is bigger than the intra-slice sampling rate [10, 78]. The cross-section shape evolution of various anatomical structures decreases the membership for nodes of

a parallelepiped-shaped ω_t (Fig. 7.1). Such computational volume shape and the related segmentation will be referred to as *non-compensated.*

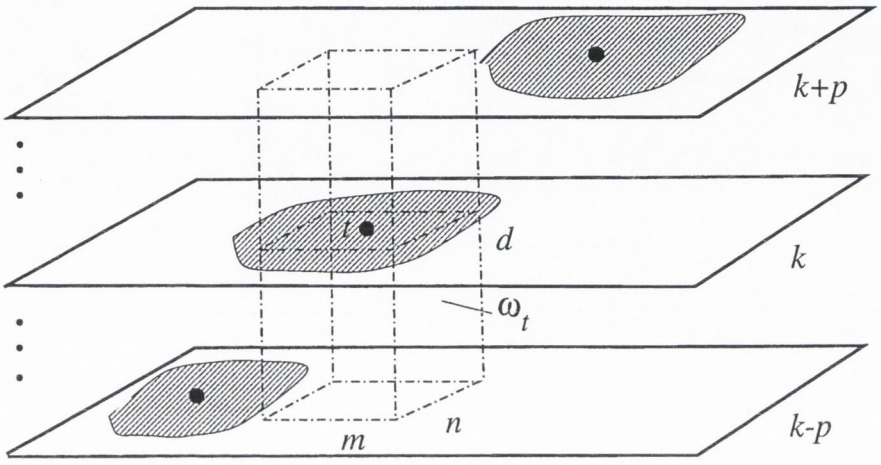

Figure 7.1: Non-compensated computational volume.

7.2.2 Difference Field

Shape evolution phenomena along the third axis can be modelled mathematically via a non-linear transformation g between the slices. Then, the node $t_z = (i_z, j_z, k_z)$, $k_z = k + z$ is the image of the node t after the transformation $t_z = g_z(t)$ for slices separated by z voxels. Many researchers have been investigating different models for the specification of g in relation to object tracking, motion compensation and registration, as summarized in Chapter 2. For example, the assumption of rigid body motion is often used in estimating object motion in an image sequence. This is not applicable for the medical imaging application. Instead of parameterization of g, non-parametric estimation of its effect is proposed in this work.

The shape evolution along the z-going dimension results in a *difference* between a pair of slices. Then, the problem of determining $g_z(t)$ can be formulated as finding the *difference vector* $\mathbf{d}_t(z)$ which maps the node t onto its image t_z: $t_z = t + \mathbf{d}_t(z)$. Therefore:

$$\mathbf{d}_t(z) = t_z - t. \tag{7.2}$$

Depending on the relative location of slices k and k_z according to the adopted ordering, the difference vector can be defined in a *forward* or *backward* manner. If $z > 0$, then the vector is defined as being forward and if $z < 0$, then it is a backward vector. If $z = 0$, then $i_z = i, j_z = j, k_z = k$, i.e. $t_0 = t$ and $\mathbf{d}_t(0) = 0$. If $1 > k \pm z > D$, i.e. if k_z goes beyond the lattice dimensions $\mathbf{d}_t(z) = \mathbf{0}$ as well. If $|z| = 1$, then z can be omitted from the notation for simplicity and the vector $\mathbf{d}_t$ is defined between a pair of consecutive slices. If the vector direction needs specification, $\mathbf{d}_t^f$ will stand for a forward vector and $\mathbf{d}_t^b$ for a backward vector (Fig. 7.2).

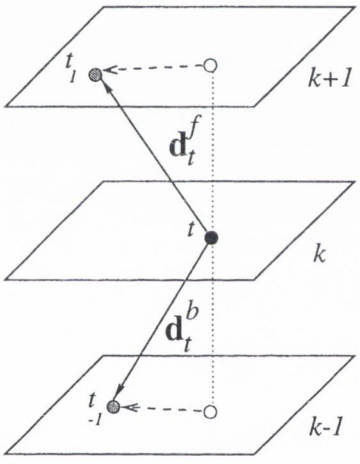

Figure 7.2: Forward and backward difference vectors.

The difference vector has three components (di, dj, dk),where $di = i_z - i$, $dj = j_z - i$ and $dk = k_z - k = z$. Usually $z = \pm 1$ or it is fixed in advance, so the difference vector for a node t is fully specified by its two-component projection (di, dj). These 2-D projections (denoted by dashed lines in Fig.7.2) will be used in schematics.

Difference vectors defined for all nodes of the lattice are elements of a *difference field* realization

$$\mathbf{D} = \{\mathbf{d}_t\}, \forall t \in \Lambda_I. \tag{7.3}$$

In an image sequence context, or in registering time-evolving processes in 3-D medical imagery, this corresponds to a *motion* or *displacement field* [48, 78, 85]. In registration of stereoscopic pair of images, it is called a *disparity field* [3].

The problem of evaluating the shape evolution along the z-going dimension can be formulated as the problem of estimating $\mathbf{D}$: given the grey-level data and possibly some prior information, one wishes to obtain the estimate $\widehat{\mathbf{D}} = \{\widehat{\mathbf{d}}_t\}, \forall t \in \mathbf{\Lambda}_I$.

Providing the difference field is known or estimated, a *difference-compensated* computational volume ω_t^c centred at node t can be constructed:

$$\omega_t^c = \bigcup_{i=0}^{p-1} \left\{ t + \sum_{j=0}^{i} \mathbf{d}_{t_j}(1) + (\Delta i, \Delta j, 0) \right\} + \bigcup_{i=0}^{p-1} \left\{ t + \sum_{j=0}^{i} \mathbf{d}_{t_{-j}}(-1) + (\Delta i, \Delta j, 0) \right\}$$
$$+ t + (\Delta i, \Delta j, 0), -l \leq \Delta i \leq l, -l \leq \Delta j \leq l. \tag{7.4}$$

Compared with the definition of the non-compensated computational volume (3.1) it can be seen that ω_t^c is a stack of $d = 2p + 1$ 2-D computational windows of size $n \times n$, $n = 2l + 1$ centred at the sequence of images of t located as indicated by the difference vectors.

An illustration of ω_t^c for the case when

$$\mathbf{d}_t(p) = \sum_{i=1}^{p} \mathbf{d}_{t_i} = p\ \mathbf{d}_t^f \tag{7.5}$$

and

$$\mathbf{d}_t(-p) = \sum_{i=-p}^{1} \mathbf{d}_{t_i} = p\ \mathbf{d}_t^b \tag{7.6}$$

is shown in Fig. 7.3.

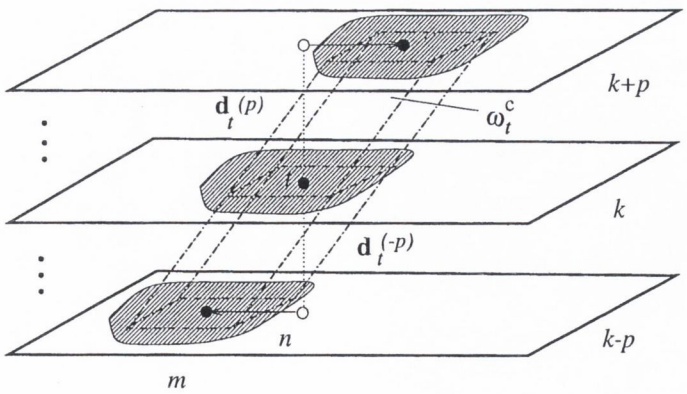

Figure 7.3: Difference-compensated computational volume.

Segmentation algorithms using compensated volumes will be referred to *difference-compensated* algorithms.

7.2.3 Simulation Example

Consider the example from Section 6.5. The ground truth mask used is a circle representing a 2-D object cross-section displaced between each pair of slices by $(+5, +5)$ pixels (Fig. C.10, Appendix C). Thus, the volume of a textured object in a shape of a 'skewed' cylinder within a textural background is created (Fig. C.11). The mask and the grey-level data for the middle slice of the volume are shown in Figure 6.16.

The deeper the non-compensated ω_t, the smaller the membership of the cylindrical texture as shown in Fig. 7.4. The membership is calculated at the centre of the circle for each middle slice for all possible computational volumes of size $15 \times 15 \times d$. The mean value for each depth is reported.

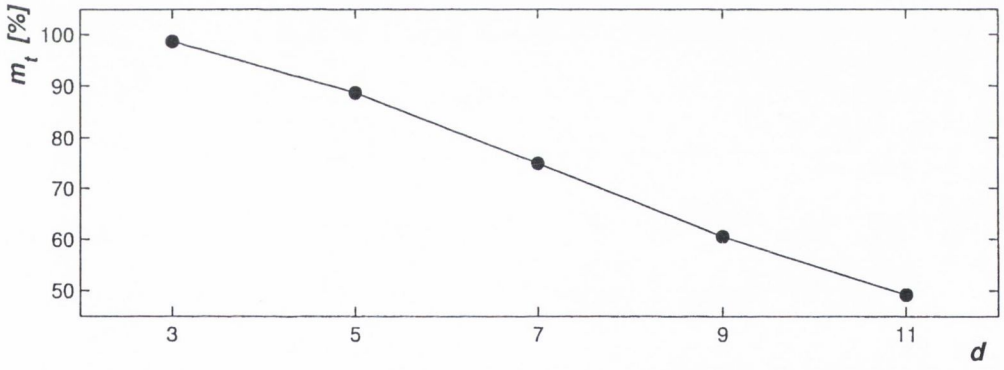

Figure 7.4: Mean membership of the texture in non-compensated ω_t for the 'skewed' cylinder volume.

The bigger the value of d, the stronger the effect of the shape evolution along the z-going dimension. This in turn reduces the accuracy of the parameter estimation (Section 5.2) as well as the segmentation accuracy, when the depth of a non-compensated ω_t increases as illustrated in Fig. 7.5 for the middle slice of the volume.

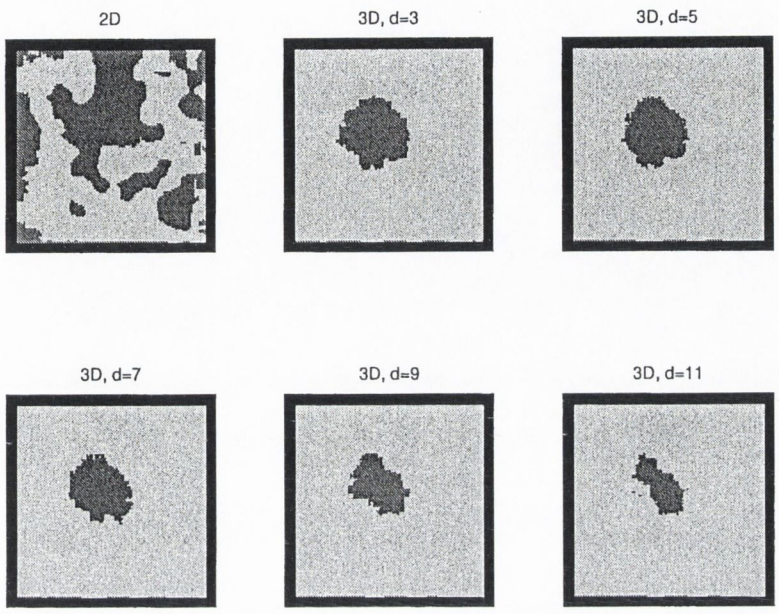

Figure 7.5: 2-D and 3-D RUR non-compensated segmentation with different depths of ω_t: slice #6 of the 'skewed' cylinder volume.

The result confirms the conclusion from Chapter 6 that the 3-D segmentation performs better than the 2-D. The latter ultimately fails because the $\boldsymbol{\theta}_h$ parameters were not enough to discriminate between the 2 textures while the former uses the whole vector $\boldsymbol{\theta}$ with $\boldsymbol{\theta}_v$ values being the key discriminants (see Table 5.2). The quality of the 3-D segmentation results, however, deteriorates with the use of deeper volumes. This contrasts to the opposite trend observed in previous simulations where no shape evolution was present.

To measure only the effect of the shape evolution, a volume of a textured 'straight' cylinder embedded within a textured background is created using the same two 3-D textures (Fig.C.9) as before. The same ground truth mask as for the middle slice of the 'skewed' cylinder volume is used for all data slices, i.e. in this case inter-slice differences are zero. The radius of the circle is 20 pixels, i.e. the membership m_t for the center of the cylinder is 100% for ω_t of size $15 \times 15 \times d$ for all d. The 3-D RUR non-compensated segmentation results with $n^d_{RUR} = 16$ and $\ell^d_{RUR} = 4$ for the middle slice of the volume are illustrated in Fig. 7.6.

This result is consistent with the conclusion from Chapter 6 about the effect of the deeper computational volume. Thus, the difference in the results for the 'skewed' and the 'straight' cylinder volumes is only due to the presence of a non-zero difference field.

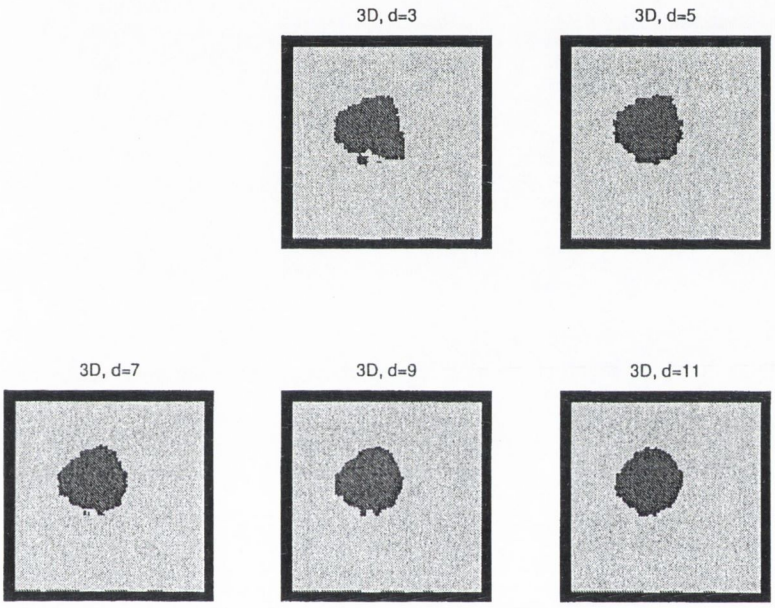

Figure 7.6: 3-D RUR non-compensated segmentation with different depths of ω_t: slice #6 of the 'straight' cylinder volume.

The two curves showing the dependence of the percentage of misclassification δ (eqn. (5.10)) on d for the middle slice of both test volumes, using non-compensated ω_t, are depicted in Fig. 7.7. For the 'straight' cylinder volume there is lack of difference for the central node and for the majority of the cylindrical texture nodes at each slice in respect to a non-compensated 3-D computational volume. The average membership for these nodes is constant and is close to 100%. Therefore, increasing the depth of ω_t decreases the classification error since it increases the estimation accuracy for the GMRF parameters. On the other hand, for the 'skewed' cylinder volume, increasing d increases the difference, (decreases m_t) as shown in Fig. 7.4. Decreasing the membership increases the misclassification.

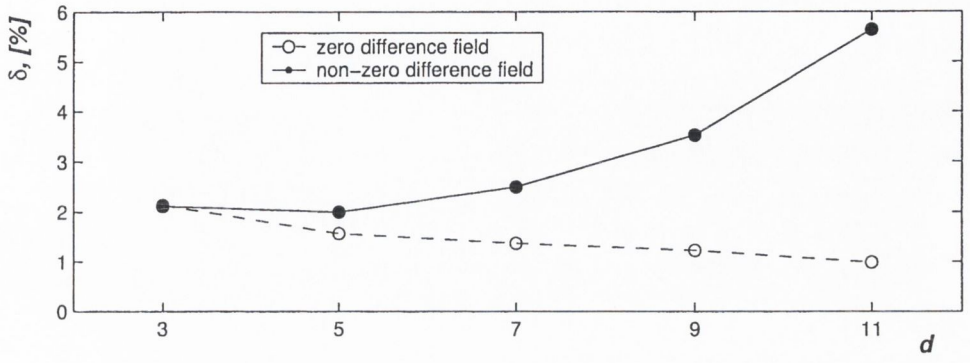

Figure 7.7: Effect of the difference field on the 3-D non-compensated segmentation accuracy. Mean misclassification for the 'skewed' and 'straight' cylinder volumes.

The same trend is observed for all slices of both test volumes as summarized in Tables 7.1 and 7.2.

Table 7.1: 3-D non-compensated RUR segmentation of the 'straight' cylinder volume: mean membership [%], misclassification [%], and elapsed time [s].

Depth	mean	mean	Slice # (misclassification)									el.time	
	m_t	δ	2	3	4	5	6	7	8	9	10	Tot.	Per sl.
$d=3$	**100**	**2.14**	2.5	2.46	2.2	2.15	2.25	1.97	2.48	2.22	1.06	10.83	**1.2**
$d=5$	**100**	**1.57**	-	2.02	1.76	1.43	1.28	1.34	1.87	1.31	-	15.41	**2.2**
$d=7$	**100**	**1.37**	-	-	1.45	1.16	1.38	1.66	1.2	-	-	16.72	**3.34**
$d=9$	**100**	**1.23**	-	-	-	1.2	1.36	1.13	-	-	-	14.67	**4.89**
$d=11$	**100**	**0.99**	-	-	-	-	0.99	-	-	-	-	30.94	**30.94**

Table 7.2: 3-D non-compensated RUR segmentation of the 'skewed' cylinder volume: mean membership [%], misclassification [%], and elapsed time [s].

Depth	mean	mean	Slice # (misclassification)									el.time	
	m_t	δ	2	3	4	5	6	7	8	9	10	Tot.	Per sl.
$d=3$	**98.67**	**2.12**	2.9	1.66	2.1	1.75	1.59	1.63	2.9	2.1	2.42	11.45	**1.27**
$d=5$	**88.78**	**2**	-	2.65	1.96	1.38	2.0	1.92	2.55	1.95	-	17.18	**2.45**
$d=7$	**75**	**2.5**	-	-	2.39	2.04	2.63	2.67	2.86	-	-	17.83	**3.56**
$d=9$	**60.5**	**3.53**	-	-	-	3.51	3.79	3.29	-	-	-	16.62	**3.37**
$d=11$	**49.2**	**5.64**	-	-	-	-	5.64	-	-	-	-	30.1	**30.1**

For the 'skewed' cylinder data the difference field satisfies equations (7.5) and (7.6), and the exact difference is known. Thus, a compensated computational volume ω_t^c of the type illustrated in Fig. 7.3 can be constructed using $\mathbf{d}_t = 0$ for the nodes belonging to the background and $\mathbf{d}_t(p) = (+5p, +5p, p)$, $\mathbf{d}_t(-p) = (-5p, -5p, -p)$ for the nodes belonging to the object mask.

The results from the 3-D compensated with the known difference field RUR segmentation with $n_{RUR}^d = 16$, $d_{RUR}^d = 4$ for the middle slice of the volume are illustrated in Fig. 7.8.

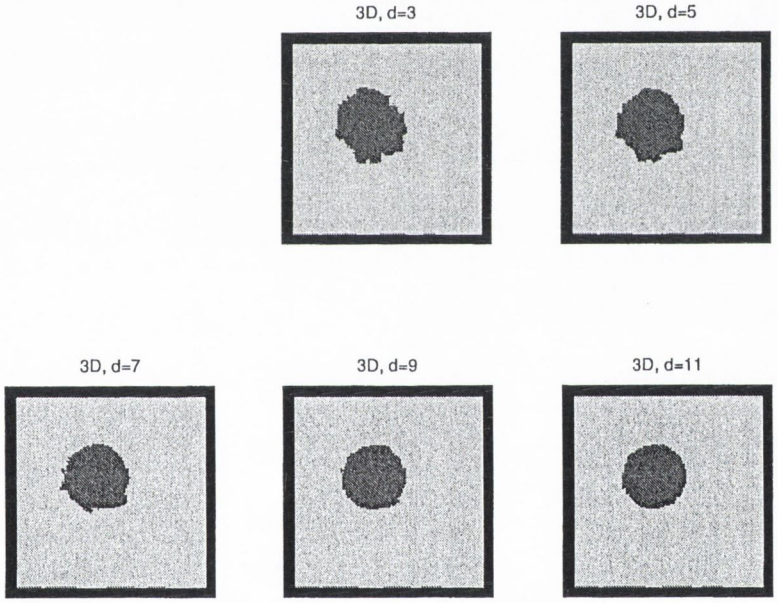

Figure 7.8: 3-D RUR compensated with the known difference field segmentation for different ω_t^c depths: slice #6 of the 'skewed' cylinder volume.

It can be clearly observed that increasing the depth of the compensated computational volume increases the segmentation accuracy. This is because ω_t^c follows the shape evolution along the z-going dimension, thus protecting the texture stationarity assumption by keeping the membership close to 100%. Comparing Figs. 7.5 and 7.8, the superiority of the compensated approach can be noted. Therefore, the same relation between the misclassification and the depth of the computational volume is observed as for the data where no difference is present. This is illustrated

by comparing Fig. 7.9 to Fig. 7.7.

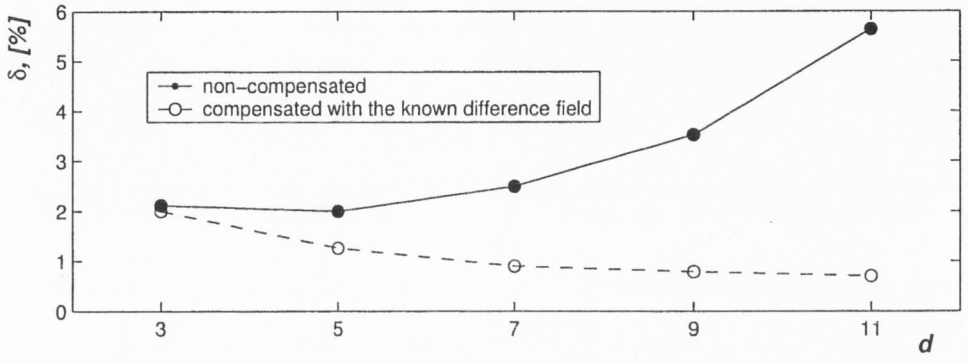

Figure 7.9: Compensating for the difference field effect on the 3-D segmentation accuracy. Mean misclassification for the 'skewed' cylinder volume with and without compensation.

Table 7.3: 3-D RUR segmentation of the of the 'skewed' cylinder volume, compensated with the exact difference field: mean membership [%], misclassification [%], and elapsed time [s].

Depth	mean	mean	Slice #									el.time	
	m_t	δ	2	3	4	5	6	7	8	9	10	Tot.	Per sl.
$d=3$	**100**	**2**	2.24	1.48	1.4	1.74	1.85	1.73	2.72	2.36	3.18	11.24	**0.59**
$d=5$	**100**	**1.26**	-	1.42	0.76	0.72	1.18	1.42	1.7	1.65	-	21.05	**3.007**
$d=7$	**100**	**0.9**	-	-	0.61	0.42	0.9	1.12	1.46	-	-	23.49	**4.69**
$d=9$	**100**	**0.79**	-	-	-	0.83	0.68	0.88	-	-	-	21.9	**7.3**
$d=11$	**100**	**0.7**	-	-	-	-	0.7	-	-	-	-	30.1	**31.21**

The performance parameters of the 3-D RUR compensated segmentation for all slices of the 'skewed' cylinder volume are given in Table 7.3.

The difference-compensated approach takes the same time to execute (excluding the time for estimating the difference field if unknown) as the non-compensated one, while the mean classification accuracy increases noticeably (Tables 7.2 and 7.3).

In conclusion, there is a definite demand for difference estimation and compensation. The following sections present various solutions to this problem.

7.3 INTEREST POINTS DETECTION AND MATCHING

The problem of difference field estimation is closely related to the *correspondence problem* arising in motion analysis, as explained in Section 2.4.2. Estimating the difference between two slices of a volumetric image can be considered as finding the correspondence between 2-D object cross-sections in these slices.

The process of finding correspondence is known as matching. Direct matching between slices from 3-D mixture of stationary fields with similar $\boldsymbol{\theta}_h$, is not a feasible task. In such data, it is extremely difficult to determine suitable correspondence tokens, hence, a simplification is needed.

Approximate texture segmentation, which transforms the raw data into label field realization (map), can be used as a pre-processing step. Different grey levels are used for different labels, thus a label map consists of regions of uniform grey level (the label encoding a texture class). Then, the matching between such regions would be based only on the similarity between their shapes. As concluded from the 3-D texture analysis so far, the 2-D segmentation cannot be used for this purpose. A 3-D non-compensated segmentation of minimum depth $d = 3$ ('thin-plate' segmentation), however, is least affected by the difference present in the original data. Hence, it can be used to produce the initial label maps, which can be used as source for selection of the correspondence tokens. A subsequent match between these tokens can produce an estimate of $\mathbf{D}$. This estimate $\widehat{\mathbf{D}}$ can be used for construction of a deeper ω_t^c, which will give the final enhanced result of the difference-compensated segmentation algorithm.

7.3.1 Moravec Operator (MO)

The first step of such an approach is to find correspondence tokens in the slices for which the correspondence is sought. Significant or *interest points* IPs are often used

for that purpose. IPs are centres of highly variable areas, which are likely to be found in both slices. In this work the IPs are located on the border between two regions in the label map.

More formally, an interest point is a node from Λ^{2D}. The IP corresponding to the label l_t at node t, is denoted by $x = (i, j)$, $x \in \Lambda^{2D}$.

The Moravec, Kitchen-Rosenfeld or Zuniga-Haralick operators are based on a local vicinity and are able to locate corners or vertices in images. A *corner* is defined as a junction point of two or more straight lines or as an intersection of two edges, oriented in different directions, which are not 180° apart [81]. The interest operators find excessive number of corners if applied to raw grey-level textural data, but are expected to work very well in an image consisting of several uniform grey level regions such as a label map.

In this work a modified colour-blind *Moravec operator* (MO) is used as an interest-point detector. The original MO is the simplest interest point operator which takes an image as input and gives as an output the likelihood that a pixel is a corner.

Consider the k-th 2-D slice $\mathbf{X}_k$ of the grey-level volume $\mathbf{X}$. Let the corresponding label field slice obtained after a 3-D 'thin-plate' non-compensated segmentation of $\mathbf{X}_k$ be $\mathbf{L}_k$. The MO works on a local 8-pixel area, producing a probability map where the maximum values correspond to the corners in the processed image:

$$MO(i,j) = \frac{1}{8} \sum_{\mathrm{i}=i-1}^{i+1} \sum_{\mathrm{j}=j-1}^{j+1} |\mathbf{L}_k(\mathrm{i},\mathrm{j}) - \mathbf{L}_k(i,j)|. \tag{7.7}$$

Because it is applied to a label map several modifications of the original MO were implemented, namely:

1. There are only few unique grey levels in a label map and it is assumed that an interest point lies on the border between two regions. Then, the modified MO detects a difference between two labels, rather than the actual values, i.e. it is colour-blind.
2. Not all label map pixels are considered, but only some from a decimated Λ^{2D}. If the four corners of a block from such a lattice have the same value, the interior of

such a block is left unprocessed.

3. Only a percentage of all corners above a certain probability threshold thresh are selected. Thus, x is chosen as an IP if $MO(i,j) >$ thresh. To ensure an even distribution of interest points over the segmentation classes, this percentage is divided equally between the classes.

7.3.2 Matching Algorithm

The second step in the procedure of estimating $\mathbf{D}$ is the matching of the sets of IPs detected independently for the two slices of interest $\mathbf{L}_k$ and $\mathbf{L}_{k_z}$ in the first step.

A modification of Barnard and Thompson's algorithm [3] for computing a velocity field between frames in an image sequence via matching of interest points is proposed. The modifications are necessary to reflect the specifics of matching IPs from slices of the label field, obtained after 3-D 'thin-plate' non-compensated segmentation.

Let $\mathcal{X} = \{x_\mathsf{m}\}$ be the set of all IPs in the first slice $\mathbf{L}_k$ and $\mathcal{Y} = \{y_\mathsf{n}\}$ be the set of all IP in the second slice of interest $\mathbf{L}_{k_z}$. Generally $|\mathcal{X}| \neq |\mathcal{Y}|$. Let $\mathbf{d}_{\mathsf{m},\mathsf{n}}$ be a 2-D vector connecting $x_\mathsf{m} = (i_\mathsf{m}, j_\mathsf{m})$ with $y_\mathsf{n} = (i_\mathsf{n}, j_\mathsf{n})$, i.e. $y_\mathsf{n} = x_\mathsf{m} + \mathbf{d}_{\mathsf{m},\mathsf{n}} \Rightarrow$ $\mathbf{d}_{\mathsf{m},\mathsf{n}} = (i_\mathsf{n} - i_\mathsf{m}, j_\mathsf{n} - j_\mathsf{m})$.

The two points x_m and y_n can be considered potentially corresponding if their absolute distance satisfies the assumptions of maximum possible difference

$$|x_\mathsf{m} - y_\mathsf{n}| \leq d_{\max}, \tag{7.8}$$

where $d_{\max}$ is the maximum distance at which a point can have its image within the slice. The correspondence between the points is based on a local similarity within a 2-D vicinity V. The similarity is measured by the sum of the squared differences between the nodes within the vicinity of the two interest points to be matched:

$$s_{\mathsf{m},\mathsf{n}} = \sum_{\substack{x \in V_{x_\mathsf{m}} \\ y \in V_{y_\mathsf{n}}}} [\mathbf{L}_k(x) - \mathbf{L}_{k_z}(y)]^2 \tag{7.9}$$

A weight is associated to each correspondence by:

$$w_{\mathsf{m,n}} = \frac{1}{1 + K s_{\mathsf{m,n}}}, \tag{7.10}$$

where K is a predefined constant and $s_{\mathsf{m,n}}$ is computed via (7.9). Let $[x_{\mathsf{m}}, y_{\mathsf{n}}]$ denote that x_{m} and y_{n} form a correspondence pair and $p_{\mathsf{m,n}}$ be the probability of $[x_{\mathsf{m}}, y_{\mathsf{n}}]$. The correspondence with the biggest weight should be the correct one, hence the initial probability that x_{m} is not matchable (has no correspondence with any y_{n}) is given by:

$$p^{*(0)}_{\mathsf{m,n}} = 1 - \max_{\mathsf{n}} w_{\mathsf{m,n}}, [x_{\mathsf{m}}, y_{\mathsf{n}}] \neq v^*, \tag{7.11}$$

where v^* is a special indicator that no potential correspondence was found and p^* is the probability for that.

Two correspondences of points $[x_{\mathsf{m}}, y_{\mathsf{n}}]$ and $[x_{\mathsf{k}}, y_{\mathsf{l}}]$ are said to be *consistent* if

$$|\mathbf{d}_{\mathsf{m,n}} - \mathbf{d}_{\mathsf{k,l}}| \leq d_{\mathrm{dif}}, \tag{7.12}$$

where d_{dif} is a preset constant.

The algorithm is iterative, starting from an initial correspondence between all possible pairs of IPs. A probability that the correspondence is correct is assigned to each pair. These probabilities are then updated iteratively to get the globally optimum set of correspondences, subject to the local smoothness assumption (7.12).

The modifications of the original algorithm [3] are as follows:

1. The algorithm is adapted to work on label maps rather than grey levels as intended originally. The similarity evaluation is based not on the actual grey levels, but on the observation if these labels are different or not. Instead of (7.9) the following expression is used:

$$s_{\mathsf{m,n}} = \mathsf{p}, \tag{7.13}$$

where $V_{x_{\mathsf{m}}}$ and $V_{y_{\mathsf{n}}}$ differ in p pixels.

2. To determine $d_{\max}$ the differential image analysis principle [42, 43] is used, i.e. $d_{\max}$ is selected to be equal to the width of the largest region in the difference image

$|\mathbf{L}_k - \mathbf{L}_{k_z}|$.

$|\mathbf{L}_3$. To avoid the 'rigid motion' constraint the value of d_{dif} is chosen to be different

3. To avoid the 'rigid motion' constraint the value of d_{dif} is chosen to be different than 1. Correspondences are considered to be consistent, only if the two grey levels in V_{x_m} are equal to the same grey levels from V_{y_n}. The search space for comparison of consistency is determined by a large value R.

Algorithm :

1. Detect all potential correspondences $[x_\mathsf{m}, y_\mathsf{n}]$, $\forall x_\mathsf{m} \in \mathcal{X}$ and $\forall y_\mathsf{n} \in \mathcal{Y}$.

2. For each point $x_\mathsf{m} \in \mathcal{X}$ create a data structure containing the vectors $\mathbf{d}_{\mathsf{m,n}}$ and the probabilities $p_{\mathsf{m,n}}$ for all $y_\mathsf{n} \in \mathcal{Y}$:

$$x_\mathsf{m} : \{(\mathbf{d}_{\mathsf{m},1}, p_{\mathsf{m},1}), (\mathbf{d}_{\mathsf{m},2}, p_{\mathsf{m},2}), \ldots, (\mathbf{d}_{\mathsf{m,n}}, p_{\mathsf{m,n}}), (v^*, p^*_{\mathsf{m,n}})\}. \tag{7.14}$$

3. Compute the initial probabilities $p^{(0)}_{\mathsf{m,n}}$ of each correspondence $[x_\mathsf{m}, y_\mathsf{n}]$:

$$p^{(0)}_{\mathsf{m,n}} = \frac{w_{\mathsf{m,n}}}{\sum_{\mathsf{l}\neq\mathsf{n}} w_{\mathsf{m,l}}}(1 - p^{*(0)}_{\mathsf{m,n}}), \tag{7.15}$$

using (7.10) and (7.11). Fill the data structure from step 2.

4. Iterate N times updating the probabilities by:

$$\tilde{p}^{(it)}_{\mathsf{m,n}} = p^{(it-1)}_{\mathsf{m,n}}(A + Bq^{(it-1)}_{\mathsf{m,n}}) \tag{7.16}$$

$$\tilde{p}^{*(it)}_{\mathsf{m,n}} = p^{*(it-1)}_{\mathsf{m,n}}, \tag{7.17}$$

where A and B are constants.

The quality of the correspondence $[x_\mathsf{m}, y_\mathsf{n}]$ is computed by:

$$q^{(it-1)}_{\mathsf{m,n}} = \sum_{\mathsf{m}\neq\mathsf{k}} \sum_{|x_\mathsf{m} - x_\mathsf{k}| \leq R} p^{(it-1)}_{\mathsf{k,l}} \tag{7.18}$$

for all correspondences $[x_\mathsf{k}, y_\mathsf{l}]$ consistent with $[x_\mathsf{m}, y_\mathsf{n}]$.

Normalize:

$$p^{(it)}_{\mathsf{m,n}} = \frac{\tilde{p}^{(it)}_{\mathsf{m,n}}}{\sum_{\forall \mathsf{l}} \tilde{p}^{(it)}_{\mathsf{m,l}}} \tag{7.19}$$

The algorithm finds the optimum set of correspondences, defined via the $\mathbf{d}_{m,n}$ vectors. These vectors are used to construct a *sparse* (only for the points in $\mathcal{X}$) estimate $\widehat{\mathbf{D}}$. $\mathbf{d}_{m,n}$ is the 2-D projection of the difference vector estimate $\widehat{\mathbf{d}}_{t_m}(z)$ for the Λ^{3D} node $t_m = (i_m, j_m, k)$, whose image is $t_n = (i_n, j_n, k_z)$.

The advantage of the proposed algorithm is that it can find difference vectors generated by very general intra-slice transform g. Only a sparse estimate of the difference field is produced as a result, which fits well within the decimated lattices philosophy of algorithms such as RUR. A disadvantage is the large number of parameters and constants, which have to be set in advance. An additional task of finding a difference vector at a specific node, given the sparsity of $\widehat{\mathbf{D}}$, has also to be solved in order to perform the compensated segmentation.

7.3.3 Simulation Results

Modified MO

The performance of the modified MO (Section 7.3.1) is illustrated in Fig. 7.10. Two 128×128 artificially generated label field realizations of a three class problem with curved boundaries are selected for testing.

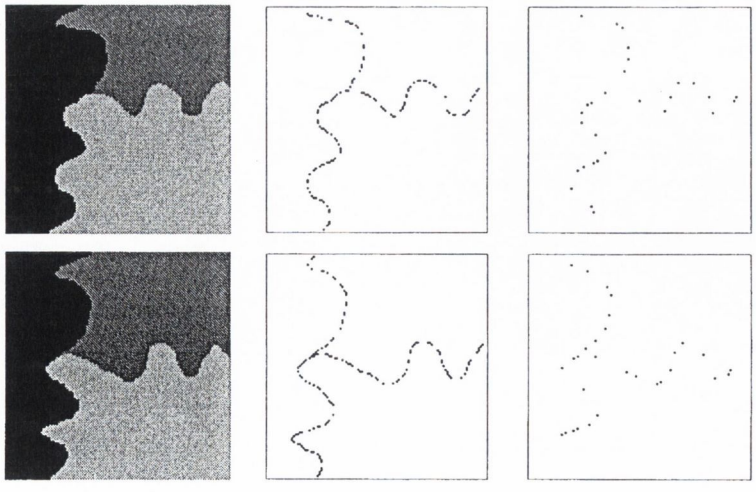

Figure 7.10: Modified Moravec operator. Original images (1st column), all corners detected (2nd column) and the selected IPs (3rd column).

For the first test image, the MO detected initially 126 corners within 0.67 s, and then a threshold was set to select the top 20% (29) of the them as IPs. For the second test image, the MO detected initially 147 corners within 0.8s and again the top 20% (30) of them were selected as IPs.

More results illustrating the MO performance are given in Section E.1.1.

Matching algorithm

The matching algorithm (Section 7.3.2) is able to detect large differences generated by an arbitrary image transformation g.

This is illustrated in Fig. 7.11 for the masks from Fig. 7.10 where the sparse $\widehat{\mathbf{D}}$ is shown in relation to the IPs detected in both images.

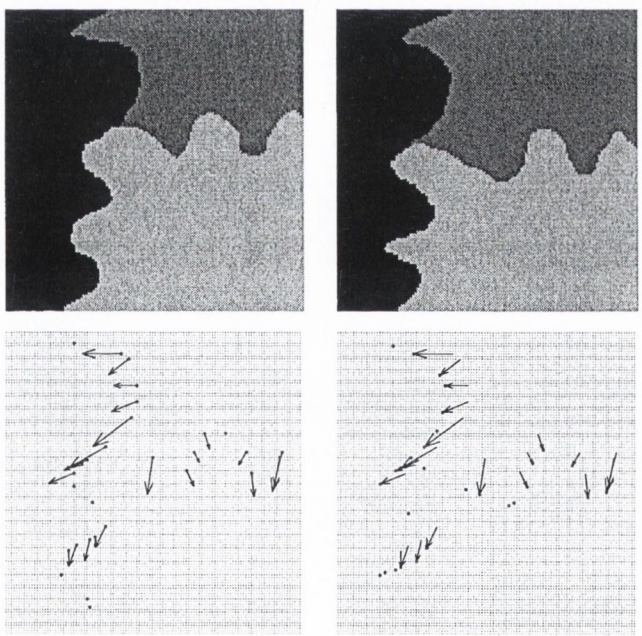

Figure 7.11: Matching algorithm. Original images (1st row) and the difference vectors with the respective IP sets (2nd row).

The algorithm parameters used were as follows: $d_{\max} = 15$ allowing up to 15 pixels displacement, $d_{\text{dif}} = 2$, in the consistency condition (7.12) allows for non-rigid displacements. The values of the constants $K = 10$, $R = 10$, $A = 0.3$, and $B = 3$

vere proposed in [3]. The evaluation of the difference vectors took 6.64 s. for N = 10 iterations.

More results of the performance of the matching algorithm are given in Section E.1.2.

Difference estimation and compensation

The IP detection and matching approach was tested on the 'skewed' cylinder data volume (Fig. C.11). The label maps obtained by 3-D non-compensated 'thin-plate' segmentation of the original data were used as a source of correspondence tokens. The modified MO was used to find sets of IPs in the slices #2, #6 and #10 of $\mathbf{L}$. The prior information that ω_t^c obeys (7.5) and (7.6) and is of the shape illustrated in Fig. 7.3 was used by considering only these three slices of $\mathbf{L}$, instead of all possible pairs of consecutive slices. The matching algorithm was able to estimate the difference between $\mathbf{L}_2$ and $\mathbf{L}_6$ for some IPs (Fig. 7.12).

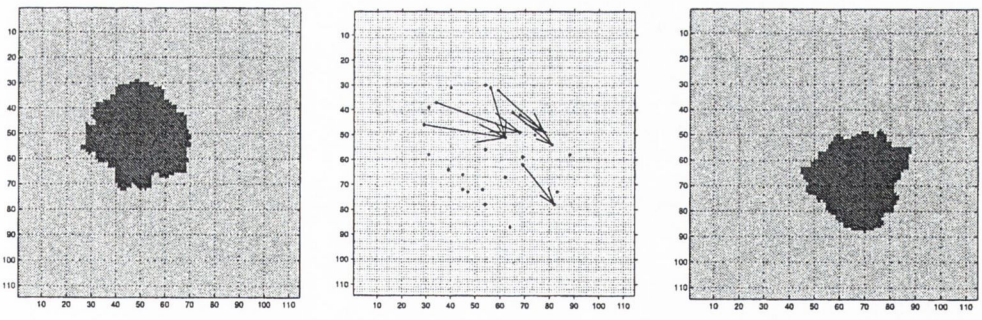

Figure 7.12: Original images and superimposed IPs (in white): 3-D non-compensated 'thin-plate' segmentation of slices #6 and #10 and the estimated sparse difference field between them.

The parameters used were: $d_{\max} = 40$, $d_{\text{dif}} = 2$, $K = 10$, $R = 10$, $A = 0.3$ and $B = 3$ and it took 3.21 s. to execute. The estimate of the difference vectors between $\mathbf{L}_2$ and $\mathbf{L}_6$ is given in Fig. E.7.

To obtain a *dense* (for all nodes of the slice) difference field estimate from the sparse $\widehat{\mathbf{D}}$, interpolation of the missing vectors for the pixels, which are not among

the set $\mathcal{X}$ of IPs is needed. It is assumed that nodes in the vicinity of each IP undergo the same local transform g. The interpolation is illustrated in Fig. 7.13.

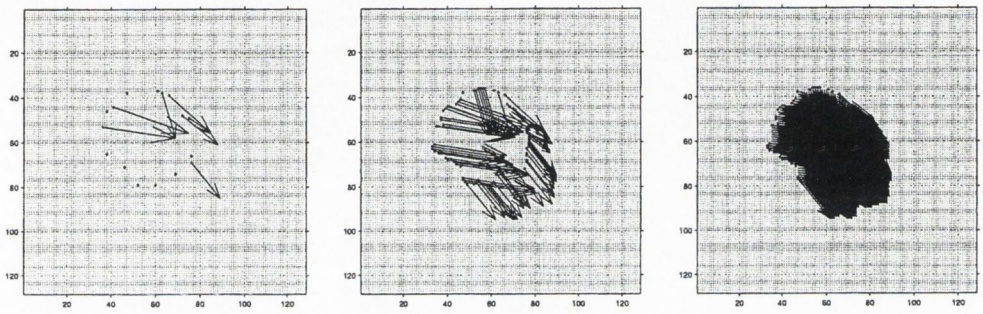

Figure 7.13: Construction of the dense $\widehat{\mathrm{D}}$ from the sparse between slices #6 and # 10.

The total elapsed time was 9.33 s for all IPs. The interpolation of the remaining vectors within the whole volume was performed assuming that (7.5) and (7.6) hold. The elapsed time for the interpolation between $\mathbf{L}_2$ and $\mathbf{L}_6$ was 9.98 s Thus, the dense $\widehat{\mathrm{D}}$ was obtained and was used to construct ω_t^c for segmentation of the original grey level volume.

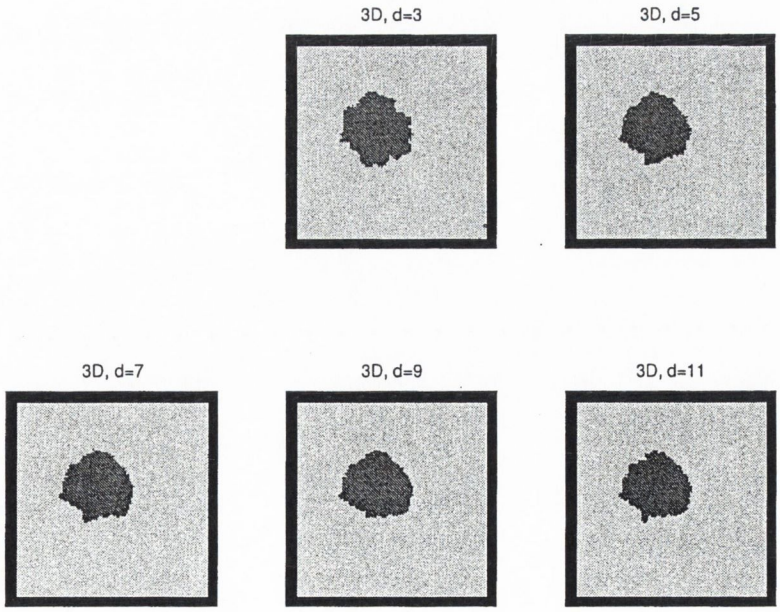

Figure 7.14: 3-D RUR compensated segmentation for different ω_t^c depths: slice #6 of the 'skewed' cylinder volume. IP detection and matching for estimation of the difference field.

The object shape has been preserved as a result of the compensated segmentation with different depths as shown in Fig. 7.14.

The performance of the compensated segmentation using $\widehat{\mathbf{D}}$ follows the trend of the theoretical result obtained using the known $\mathbf{D}$ as shown in Fig. 7.15.

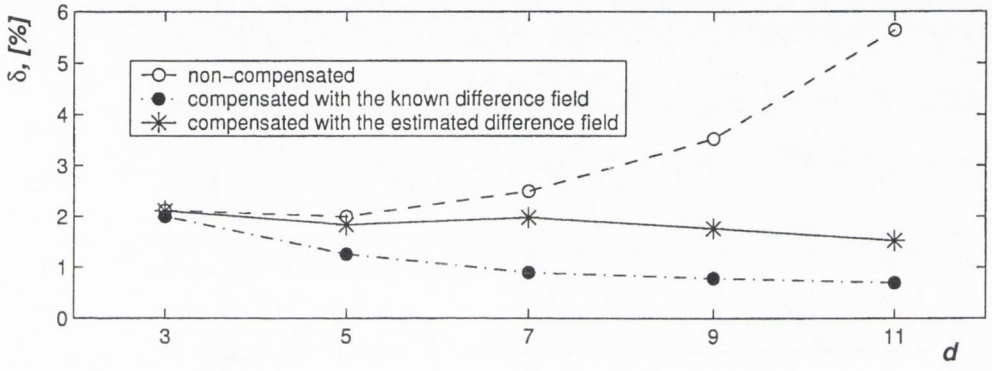

Figure 7.15: Mean misclassification for the 'skewed' cylinder volume with and without compensation. Difference field estimation via IP detection and matching.

The mean misclassification is bigger than with the known difference field, but the esimation procedure was able to produce $\widehat{\mathbf{D}}$ which enabled ω_t^c to follow the shape evolution. This can be noted by comparison of Figs. 7.5, 7.8 and 7.14.

The algorithm performance parameters for all slices and depths are given in Table 7.4 and the segmentation results in Section E.1.3.

Table 7.4: 3-D compensated with the estimated difference field RUR segmentation of the 'skewed' cylinder volume: misclassification, [%] and elapsed time, [s].

Depth	mean	Slice #									el.time	
	δ	2	3	4	5	6	7	8	9	10	Tot.	Per sl.
$d = 3$	**2.11**	2.58	1.8	1.69	1.79	1.69	1.59	2.92	2.15	2.79	9.58	**1.06**
$d = 5$	**1.84**	-	2.32	1.73	1.16	1.2	1.96	2.39	2.09	-	16.904	**2.41**
$d = 7$	**1.98**	-	-	2.55	1.62	1.35	1.84	2.55	-	-	18.74	**3.74**
$d = 9$	**1.77**	-	-	-	2.09	1.39	1.82	-	-	-	18.25	**6.08**
$d = 11$	**1.53**	-	-	-	-	1.53	-	-	-	-	36.01	**36.01**

Some more results from the IP detection and matching approach were published in [71].

7.4 CROSS-CORRELATION

As explained in Chapter 2, there is a close relation between the difference field estimation problem and motion estimation in video sequences, template matching or 2-D registration. An important technique in these applications is the so called *block-based* approach [48, 78, 90]. In this approach, the assumption is that the difference (motion, displacement) is the same within a region of the image called block. Then, a block centred around the pixel of interest in the first image is compared with candidate blocks centred at possible candidate pixels for matching in the second image within a search area. A measure of similarity between the two blocks is computed and the candidate block which maximizes the similarity is considered to be the best match. The difference vector is constructed between the centres of the original block and the best match. The difference with the previous approach is that there is no IP detection step, hence the dense field can be obtained at once if applied to all pixels of the processed slice.

7.4.1 2-D Block Matching

Denote by B_{ij} a 2-D block of size $N_b \times N_b$, $N_b = 2L_b + 1$ centred at node t from $\mathbf{L}$. A candidate block for matching of the same size is denoted by B'_{uv} centred at (u, v) in slice k_z, and the best match is the block $B^*_{i_z j_z}$ centred around t_z. Define the blocks B_{ij} and B'_{uv} as follows:

$$\begin{aligned} B_{ij} &= \mathbf{L}_k(i + \mathsf{i}, j + \mathsf{j}), \\ B'_{uv} &= \mathbf{L}_{k_z}(u + \mathsf{i}, v + \mathsf{j}), \\ -L_b \leq \mathsf{i} \leq L_b, &-L_b \leq \mathsf{j} \leq L_b \end{aligned} \tag{7.20}$$

The candidate locations (u, v) are limited by a search space specified by the maximum difference $d_{\max}$. The search accuracy is controlled by the resolution parameter Δ. If $\Delta = 1$ this is equivalent to an exhaustive search. This is illustrated in Fig. 7.16.

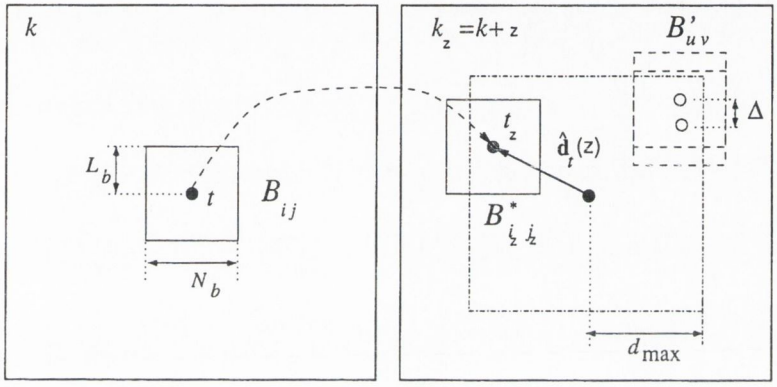

Figure 7.16: 2-D Block-based matching.

The best match is the one which maximizes a similarity measure $S_{ij}^{uv} = S(B_{ij}, B'_{uv})$ between blocks B_{ij} and B'_{uv}:

$$B^*_{i_z j_z} = \arg\max_{B'_{uv}} S(B_{ij}, B'_{uv}), \tag{7.21}$$

$$u = i + \Delta i, v = j + \Delta j,$$

$$-d_{\max} \leq \Delta i \leq d_{\max}, -d_{\max} \leq \Delta j \leq d_{\max}.$$

Then, the components of the 2-D projection of $\mathbf{d}_t(z)$ are the values $di = \Delta i$ and $dj = \Delta j$ for the best match $B^*_{i_z j_z}$.

This method does not limit the generality of the transformation between the candidate blocks. Rather than looking for an exact pattern matching, textural similarity is sought between blocks centred on texture border nodes.

The similarity measure can be defined in different ways: sum of squares differences of grey levels, cross-correlation, mutual information [81, 90]. The normalized cross-correlation known as *correlation coefficient* has been shown to be a good choice [62].

7.4.2 2-D Correlation Coefficient

The correlation coefficient $C_{ab}^{pq} = C(B_{ab}, B'_{pq})$ between two 2-D blocks B_{ab} and B'_{pq} is given by:

$$C_{ab}^{pq} = \frac{1}{\sigma_{ab}\sigma'_{pq}} \sum_{i=-L_b}^{L_b} \sum_{j=-L_b}^{L_b} [B_{ab}(i,j) - \bar{B}_{ab}][B'_{pq}(i,j) - \bar{B'}_{pq}] \tag{7.22}$$

where $\bar{B}_{ab}$ and $\bar{B'}_{pq}$ are the mean grey level values and σ_{ab} and σ'_{pq} are the standard deviations of the data within the blocks B_{ab} and B'_{pq}, i.e.:

$$\bar{B}_{ab} = \frac{1}{N_b^2} \sum_i \sum_j B_{a,b}(i,j) \tag{7.23}$$

and

$$\sigma_{ab} = \frac{1}{N_b} \sqrt{\sum_i \sum_j [B_{ab}(i,j) - \bar{B}_{ab}]^2} \tag{7.24}$$

If one expands the expressions for σ_{ab} and σ'_{pq} in (7.22) the following formula applies:

$$C_{ab}^{pq} = \frac{N_b^2 \sum_i \sum_j [B_{ab}(i,j) - \bar{B}_{ab}][B'_{pq}(i,j) - \bar{B'}_{pq}]}{\sqrt{\left(\sum_i \sum_j [B_{ab}(i,j) - \bar{B}_{ab}]^2\right)\left(\sum_i \sum_j [B'_{pq}(i,j) - \bar{B'}_{pq}]^2\right)}} \tag{7.25}$$

The correlation coefficient is known as well as Zero-mean Normalized Cross-Correlation (ZNCC).

7.4.3 Fast Computation

Although, ZNCC is faster than other similarity measures [62], it is calculated at great number of nodes, especially when dense $\widehat{\mathbf{D}}$ is required. Therefore, it is important to investigate ways for computational speedup.

In this work an even faster implementation of the calculation on the ZNCC is

proposed. If the variance σ_{ab}^2 from (7.24) is considered it follows:

$$\begin{aligned} N_b^2\sigma_{ab}^2 &= \sum_{\mathrm{i}}\sum_{\mathrm{j}}[B_{ab}(\mathrm{i},\mathrm{j}) - \bar{B}_{ab}]^2 \\ &= \sum_{\mathrm{i}}\sum_{\mathrm{j}}[B_{ab}^2(\mathrm{i},\mathrm{j}) - 2B_{ab}(\mathrm{i},\mathrm{j})\bar{B}_{ab} + \bar{B}_{ab}^2] \\ &= \sum_{\mathrm{i}}\sum_{\mathrm{j}} B_{ab}^2(\mathrm{i},\mathrm{j}) - \bar{B}_{ab}[2\sum_{\mathrm{i}}\sum_{\mathrm{j}} B_{ab}(\mathrm{i},\mathrm{j}) - N_b^2\bar{B}_{ab}] \end{aligned} \tag{7.26}$$

Then, substituting (7.23) into (7.26):

$$\begin{aligned} N_b^2\sigma_{ab}^2 &= \sum_{\mathrm{i}}\sum_{\mathrm{j}} B_{ab}^2(\mathrm{i},\mathrm{j}) - \bar{B}_{ab}[2N_b^2\bar{B}_{ab} - N_b^2\bar{B}_{ab}] \\ &= \sum_{\mathrm{i}}\sum_{\mathrm{j}} B_{ab}^2(\mathrm{i},\mathrm{j}) - N_b^2\bar{B}_{ab}^2 \end{aligned} \tag{7.27}$$

The same can be applied to σ_{pq} leading to the final expression for the ZNCC:

$$C_{ab}^{pq} = \frac{N_b^2\sum_{\mathrm{i}}\sum_{\mathrm{j}}[B_{ab}(\mathrm{i},\mathrm{j}) - \bar{B}_{ab}][B'_{pq}(\mathrm{i},\mathrm{j}) - \bar{B}'_{pq}]}{\sqrt{\left([\sum_{\mathrm{i}}\sum_{\mathrm{j}} B_{ab}^2(\mathrm{i},\mathrm{j}) - N_b^2\bar{B}_{ab}^2]\right)\left([\sum_{\mathrm{i}}\sum_{\mathrm{j}} {B'_{pq}}^2(\mathrm{i},\mathrm{j}) - N_b^2\bar{B}'^2_{pq}]\right)}}. \tag{7.28}$$

Comparing (7.25) with (7.28) it can be seen that the sums of squared grey-level values at the nodes within the block can be calculated at the same step as calculating the means for the latter.

7.4.4 Simulation Results

2-D Correlation coefficient

The ZNCC can produce good estimates of the difference between label maps with smooth boundaries as illustrated in Fig. 7.17. The difference field shown has been estimated on a decimated grid at each 5-th node of the absolute difference image $|\mathbf{L}_k - \mathbf{L}_{k_z}|$, in a computational time of 28.43 s. The parameters used were: $d_{\max} = 15$, $\Delta = 1$, $L_b = 5$.

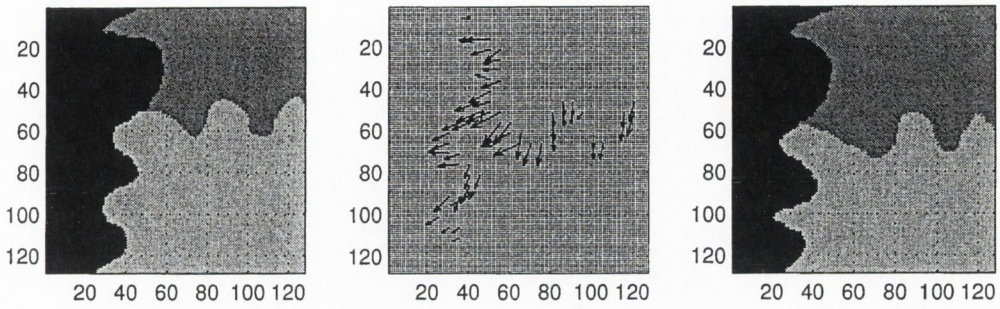

Figure 7.17: Grey-level masks and the estimated difference field between them.

When used in 3-D non-compensated 'thin-plate' segmentation, the ZNCC is affected by the region borders imperfections as illustrated in Fig. 7.18.

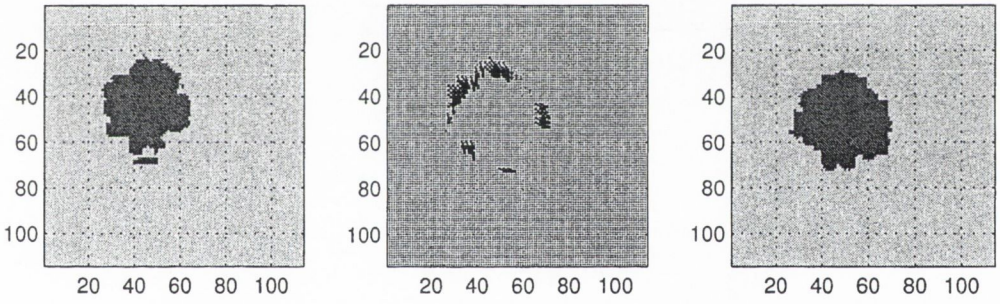

Figure 7.18: 3-D non-compensated 'thin-plate' RUR segmentation results for slices #5 and #6 of the 'skewed' cylinder volume and the estimated difference field between them.

The same parameters were used as before except $d_{\max} = 5$. More illustrations of the performance of the ZNCC are given in Section E.2.1.

Fast computation

To compare the performance of the fast (7.28) and the standard computations (7.25) of the ZNCC, the grey level masks shown in Fig. 7.17 were resampled to different sizes and both formulae were applied for the dense difference field estimation. The parameters used were $L_b = 5$, $\Delta = 1$ and various values of $d_{\max}$ as specified in Table 7.5. The absolute elapsed times are given in Table 7.5. The fast computation approach outperforms the standard one by an order of 10, which is an important factor when the size of the image increases.

Table 7.5: Elapsed times, [s] for the dense difference field estimation using standard and fast ZNCC computations for different data resolutions and $d_{\max}$ [pixels]

Resolution	$d_{\max}$	Standard	Fast
16×16	2	0.65	0.16
32×32	4	16.86	2.99
64×64	8	308.67	50.6
128×128	16	5826.7	857.76

Compensated segmentation results

The difference field for the 'skewed' cylinder volume (Fig. C.11) was estimated using the fast ZNCC technique between each pair of consecutive slices of the label field produced by the 3-D 'thin-plate' non-compensated segmentation.

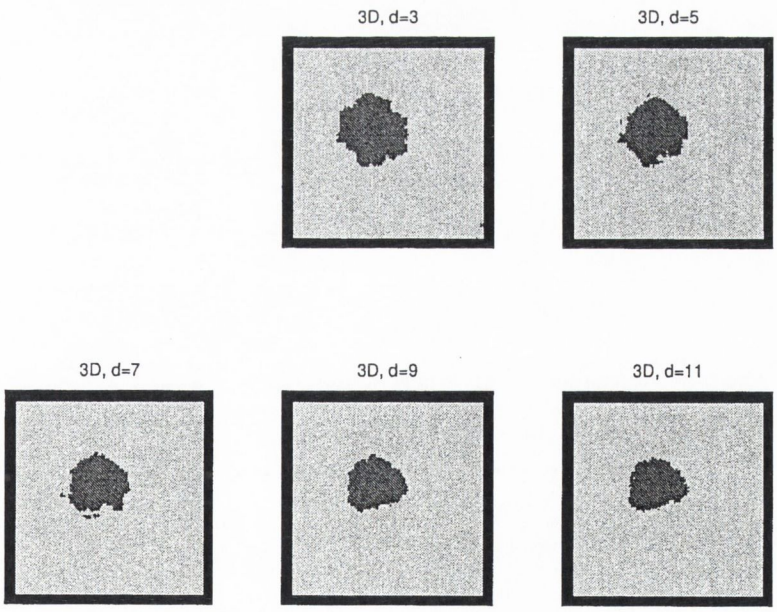

Figure 7.19: 3-D RUR compensated segmentation for different ω_t^c depths, slice #6 of the 'skewed' cylinder volume. Difference field estimation by ZNCC.

The 3-D RUR compensated segmentation using the obtained estimate $\widehat{\mathbf{D}}$ for the middle slice is visually better than the non-compensated one. Although the

tendency for the classification accuracy to decrease with increasing ω_t^c depth is still present (Fig. 7.19). The segmentation results for all slices of the volume are given in Section E.2.2.

The mean percentage misclassification reported in Table 7.6 is slightly better than the non-compensated case for small depths and is smaller for bigger depths, but the quality of the segmentation still follows the non-compensated trend as illustrated in Fig. 7.20.

Table 7.6: 3-D RUR segmentation of the 'skewed' cylinder volume compensated via ZNCC estimated difference field: misclassification [%], and elapsed time [s].

Depth	mean	Slice #									el.time	
	δ	2	3	4	5	6	7	8	9	10	Tot.	Per sl.
$d=3$	**2.11**	2.58	1.8	1.69	1.79	1.69	1.59	2.92	2.15	2.79	10.93	**1.22**
$d=5$	**2.15**	-	2.54	2.1	1.6	1.95	1.9	2.8	2.15	-	19.83	**2.83**
$d=7$	**2.53**	-	-	2.57	2.45	2.76	2	2.9	-	-	19.07	**3.81**
$d=9$	**2.88**	-	-	-	2.85	3.18	2.61	-	-	-	18.57	**6.19**
$d=11$	**3.53**	-	-	-	-	3.53	-	-	-	-	39.771	**39.77**

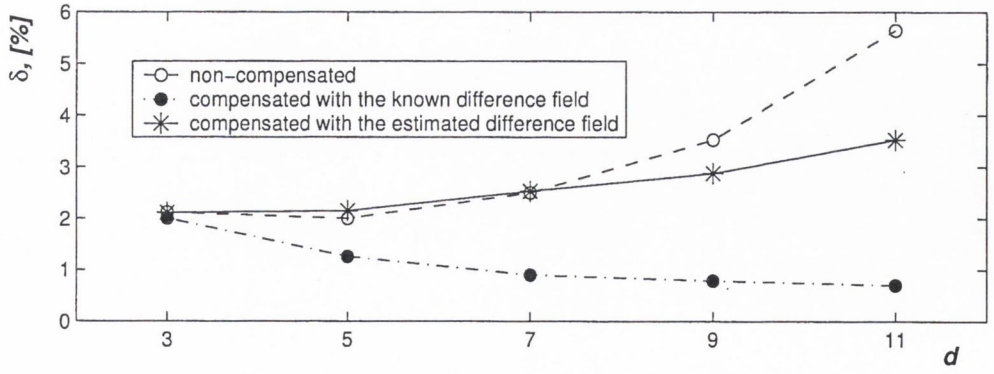

Figure 7.20: Mean misclassification for the 'skewed' cylinder volume with and without compensation. Difference field estimation by ZNCC.

This result can be explained by looking at the expression (7.28) for the ZNCC. The correlation coefficient performs well as a difference estimator providing the

ZNCC for the best block has a distinct maximum over other potential candidates. When working with $\mathbf{L}$, the block variances are greatly affected by the geometry of the region borders. Thus, a poor non-compensated result produces a poor $\widehat{\mathbf{D}}$. Hence the performance for $d > 3$ is poor because the inaccuracy of the estimate accumulates over depth.

In comparison with the IP detection and matching scheme, no extra information about $\mathbf{D}$ was provided *a priori* which also affected the final performance.

Sensitivity to the non-compensated segmentation accuracy

The dependence of the ZNCC performance on the quality of the 3-D 'thin-plate' non-compensated segmentation results can be illustrated using the results published in [72]. The same 'skewed' cylinder mask was used, but different stationary GMRFs to compose the GMRF mixture (Fig. C.12). The textures were synthesized with similar $\boldsymbol{\theta}_h$ and different $\boldsymbol{\theta}_v$ resulting in unsatisfactory 2-D RUR segmentation, as seen in Fig. 7.21.

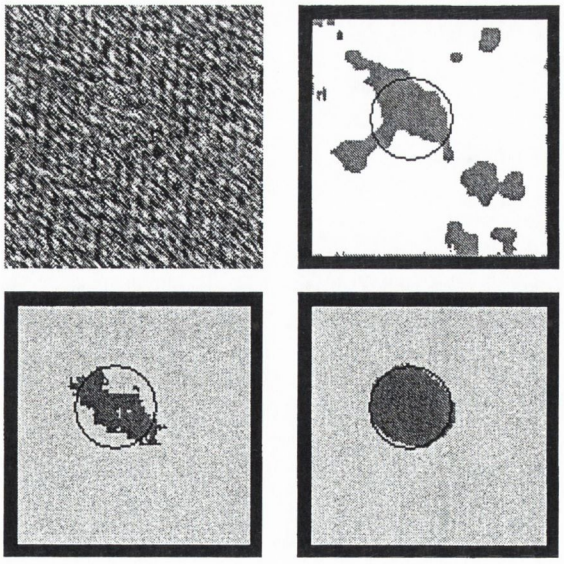

Figure 7.21: Slice # 6 from synthetic data set and 2-D RUR segmentation (1st row); 3-D segmentation results: non-compensated and compensated with the estimated difference field (2nd row).

The stationary fields properties, however, were different, which had influence on the better quality of the 3-D non-compensated 'thin-plate' segmentation result in comparison to the other dataset. This can be noted by comparing the first rows of Tables 7.2 and 7.7.

For this data set the compensated segmentation using $\widehat{\mathbf{D}}$ obtained by 2-D ZNCC block matching is closer to the result using the known $\mathbf{D}$. This is illustrated in Figs. 7.21 and 7.22 and Table 7.7.

The actual segmentation results for all depths are given in Section E.2.2.

Table 7.7: Supervised segmentation:[%] misclassification and elapsed time [s]

Algorithm												
Depth	Mean	Slice #									El.time	
	δ	2	3	4	5	6	7	8	9	10	Tot.	Per sl.
Non-compensated segmentation												
$d = 3$	**1.21**	1.32	0.9	1.85	1.45	1.29	1.32	1.38	1.18	1.24	40.43	**4.49**
$d = 5$	**1.45**	-	1.33	1.52	1.2	1.43	1.36	1.88	1.45	-	40.19	**5.74**
$d = 7$	**2.17**	-	-	2.14	1.64	2.2	2.4	2.47	-	-	37.77	**7.55**
$d = 9$	**3.17**	-	-	-	2.7	3.5	3.32	-	-	-	32.7	**10.9**
$d = 11$	**5.56**	-	-	-	-	5.56	-	-	-	-	44.83	**44.83**
Compensated with the known difference field												
$d = 3$	**1.27**	1.63	1.45	1.38	1.49	1.33	1.16	1.04	0.95	1.04	9.9	**1.1**
$d = 5$	**0.99**	-	1.09	1.02	1.22	1.07	0.96	0.8	0.74	-	19.58	**2.79**
$d = 7$	**0.75**	-	-	0.98	0.83	0.73	0.7	0.49	-	-	22.20	**4.44**
$d = 9$	**0.6**	-	-	-	0.71	0.66	0.44	-	-	-	20.61	**6.87**
$d = 11$	**0.69**	-	-	-	-	0.69	-	-	-	-	43.29	**43.29**
Compensated with the estimated difference field												
$d = 3$	**1.25**	-	1.4	1.39	1.5	1.34	1.16	0.98	0.95	-	29.59	**4.21**
$d = 5$	**1.14**	-	-	1.34	1.49	0.9	0.94	1.01	-	-	41.3	**8.26**
$d = 7$	**1.06**	-	-	-	1.3	0.98	0.89	-	-	-	43.29	**14.43**
$d = 9$	**1.16**	-	-	-	-	1.16	-	-	-	-	55.88	**55.88**

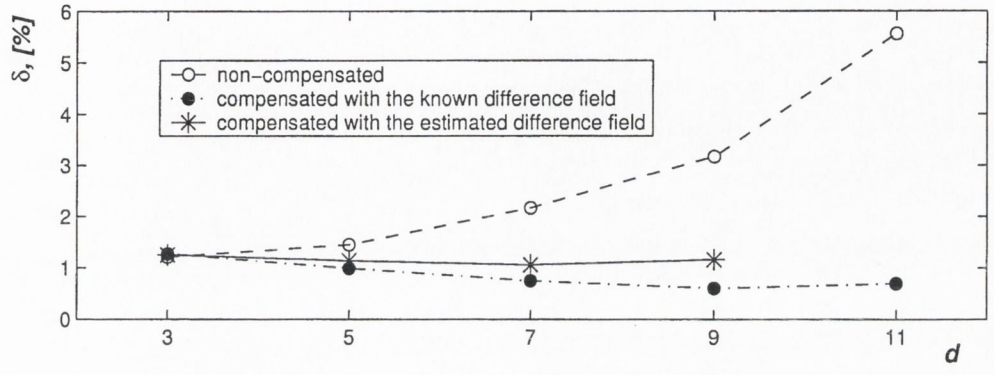

Figure 7.22: Mean misclassification for the 'skewed' cylinder volume with and without compensation.

Similar performance of the difference estimation methods have been observed on different synthetic datasets where various types of shape evolution were modelled.

7.5 CONCLUSIONS

The difference estimation methods, presented in this chapter, namely the IP detection and matching and the ZNCC, can model general slice-to-slice shape evolution. The estimated difference field obtained improved the quality of the compensated segmentation of synthetic data with shape evolution along the z-going dimension.

These methods, however, have the disadvantage of using 2-D data blocks and features drawn from $\mathbf{L}$ instead of $\mathbf{X}$ to be used as correspondence tokens. The information provided by the third z-going dimension, incorporated in $\mathbf{L}$, is obtained in the pre-processing stage by the 3-D 'thin-plate' non-compensated segmentation. Thus, the methods are dependant on the accuracy of this pre-processing stage and the total elapsed time is composed of the times necessary for the pre-segmentation, for the difference estimation and for the final compensated segmentation. A desirable characteristic of a difference estimation method would be to obtain $\widehat{\mathbf{D}}$ based directly on the original grey level data $\mathbf{X}$, and using a single unified procedure. Approaches for solving this problem are presented in the next chapter.

Chapter 8

3-D METHODS FOR DIFFERENCE ESTIMATION

8.1 INTRODUCTION

The techniques in Chapter 7 present a solution to the difference estimation problem based on the comparison of preliminary obtained slices of a label field realization. These techniques avail of the 3-D nature of the original grey-level data only to the extent available from the 'thin-plate' segmentations. The accuracy of the difference estimator depends strongly on the quality of this pre-segmentation. Since the total procedure consists of several stages, it is unattractive in real-time applications. The success of each stage depends on that of the previous one, and so error propagation is a problem.

This chapter presents novel approaches for difference estimation based on the original grey-level data. They involve fewer stages than the label-field-based methods, as illustrated in Fig. 8.1.

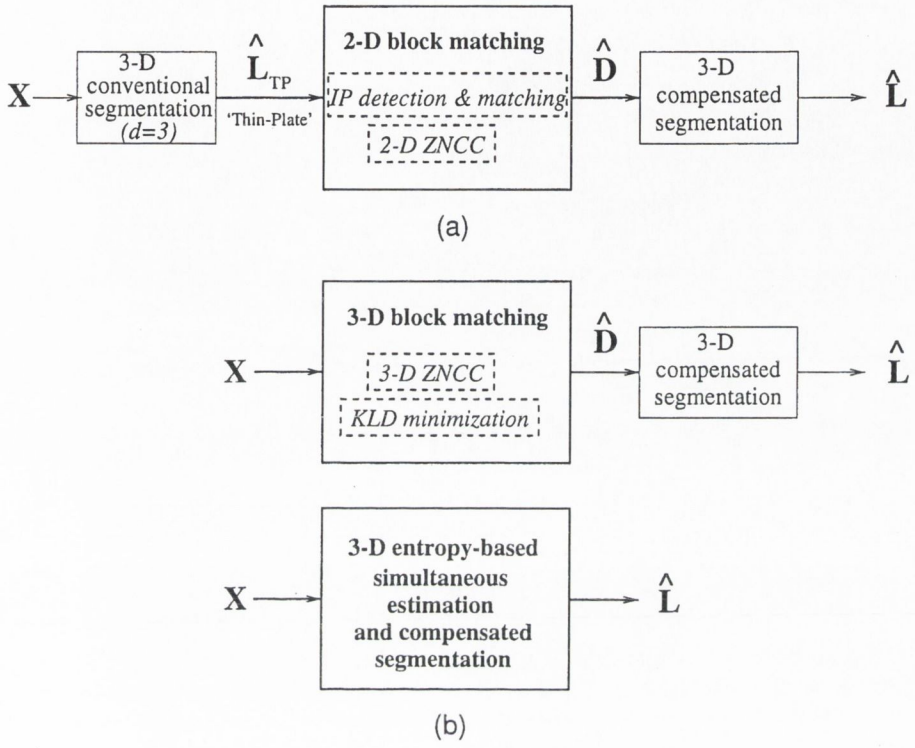

Figure 8.1: Label field (a) and grey level-based (b) difference estimation and compensation.

The 3-D cross-correlation (Section 8.2) and the Kullback-Leibler Distance (KLD) minimization (Section 8.3) methods are 3-D block matching techniques. They calculate and minimize a similarity measure between 3-D blocks of grey-level data in order to estimate the difference between nodes of interest. These methods are shown to be suitable for MRI segmentation (Section 8.4). Finally, an elegant framework for simultaneous difference estimation and compensated segmentation via entropy minimization is presented in Section 8.5.

8.2 3-D CROSS-CORRELATION

This method is a natural extension of the 2-D cross-correlation technique presented in Section 7.4. Here, 3-D blocks of data centred at the current and the candidate nodes are considered.

8.2.1 3-D Block Matching

Denote by B_t a 3-D block of size $N_b \times N_b \times d_b$, $N_b = 2L_b + 1$ centred at t. A 3-D candidate matching block (of the same size) is denoted by $B_{t'}$ centred at $t' = (i', j', k_z)$ and the best match is the block $B^*_{t_z}$ centred around t_z. Usually $d_b = 3$, i.e. 'thin-plate' blocks are used, as denoted in Fig. 8.2.

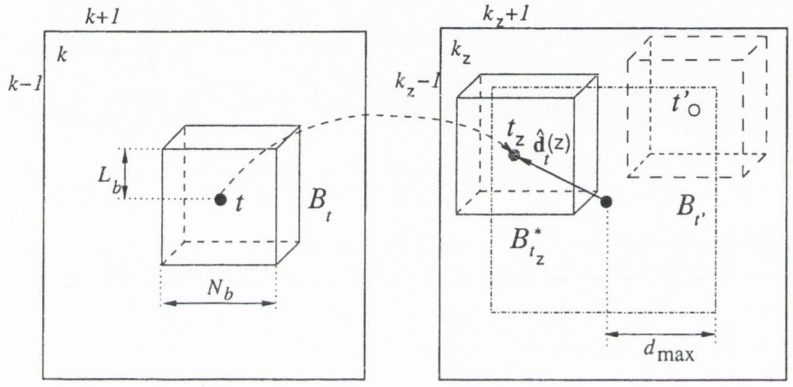

Figure 8.2: 3-D Block-based matching.

The blocks are formally defined as:

$$B_t = \mathbf{X}(i + \mathsf{i}, j + \mathsf{j}, k + \mathsf{k}), \tag{8.1}$$

$$B_{t'} = \mathbf{X}(i' + \mathsf{i}, j' + \mathsf{j}, k_z + \mathsf{k}), \tag{8.2}$$

$$-L_b \leq \mathsf{i} \leq L_b, -L_b \leq \mathsf{j} \leq L_b, -1 \leq \mathsf{k} \leq 1$$

The matching is performed by examining candidate blocks centred on nodes at slice $k_z = k + \mathsf{z}$ within some search space specified by $d_{\max}$ and Δ as in the 2-D case (Figs. 7.16, 8.2). The best match is the block, which maximizes the similarity measure $S(B_t, B_{t'})$:

$$B^*_{t_z} = \arg\max_{B_{t'}} S(B_t, B_{t'}), \tag{8.3}$$

$$i' = i + \Delta i, j' = j + \Delta j, k' = k_z$$

$$-d_{\max} \leq \Delta i \leq d_{\max}, -d_{\max} \leq \Delta j \leq d_{\max}.$$

Then, the difference vector estimate is $\widehat{\mathbf{d}}_t(\mathsf{z}) = t_z - t$.

8.2.2 3-D Correlation Coefficient

The 3-D correlation coefficient $C_r^s = C(B_r, B_s)$ between two 3-D blocks B_r and B_s, centred at nodes r and s, in analogy to (7.22) and using (8.1) is given by:

$$C_r^s = \frac{1}{\sigma_r \sigma_s} \sum_{\mathsf{k}=-1}^{1} \sum_{\mathsf{i}=-L_b}^{L_b} \sum_{\mathsf{j}=-L_b}^{L_b} [B_r(\mathsf{i},\mathsf{j},\mathsf{k}) - \bar{B}_r][B_s(\mathsf{i},\mathsf{j},\mathsf{k}) - \bar{B}_s] \tag{8.4}$$

where $\bar{B}_r$ and $\bar{B}_s$ are the mean grey level values and σ_r and σ_s are the standard deviations of the data within the blocks B_r and B_s respectively. d_b is assumed to be 3.

If the same idea as developed in Section 7.4.3 is used, the following formula for fast computation of the 3-D cross-correlation on image blocks applies:

$$C_r^s = \frac{\sum_{\mathsf{k}} \sum_{\mathsf{i}} \sum_{\mathsf{j}} [B_r(\mathsf{i},\mathsf{j},\mathsf{k}) - \bar{B}_r][B_s(\mathsf{i},\mathsf{j},\mathsf{k}) - \bar{B}_s]}{\sqrt{\left([\sum_{\mathsf{k}} \sum_{\mathsf{i}} \sum_{\mathsf{j}} B_r^2(\mathsf{i},\mathsf{j},\mathsf{k}) - M_b \bar{B}_r^2]\right)\left([\sum_{\mathsf{k}} \sum_{\mathsf{i}} \sum_{\mathsf{j}} B_s^2(\mathsf{i},\mathsf{j},\mathsf{k}) - M_b \bar{B}_s^2]\right)}}, \tag{8.5}$$

where $M_b = 3N_b^2$.

The 3-D correlation coefficient will be referred to as 3-D ZNCC. The performance of the fast method of computing the 3-D ZNCC is shown in Fig. 8.3. Two blocks of data of size $N_b \times N_b \times 3$ were generated with random entries and the 3-D cross correlation between them was computed using the equations (8.4) and (8.5). The corresponding elapsed times on a logarithmic scale are plotted versus N_b in Fig. 8.3.

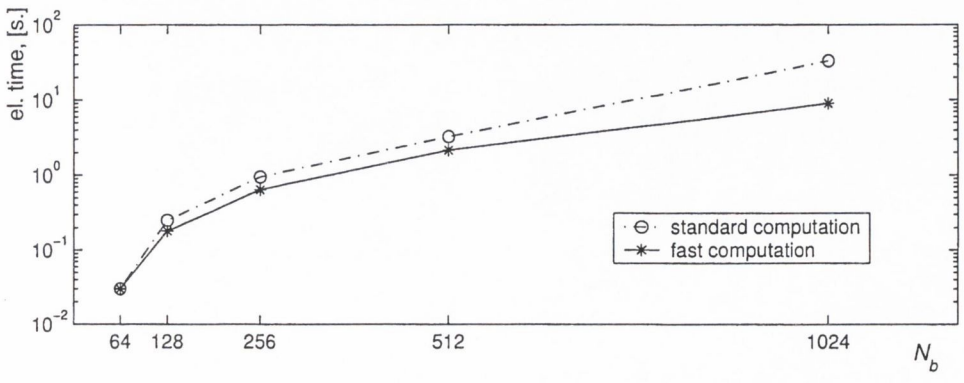

Figure 8.3: Standard vs. fast computation of the 3-D ZNCC.

It can be seen that the bigger the data dimensions, the bigger computational speedup when using the fast approach is achieved.

The advantage of using 3-D ZNCC instead of 2-D ZNCC is illustrated on Fig. 8.4.

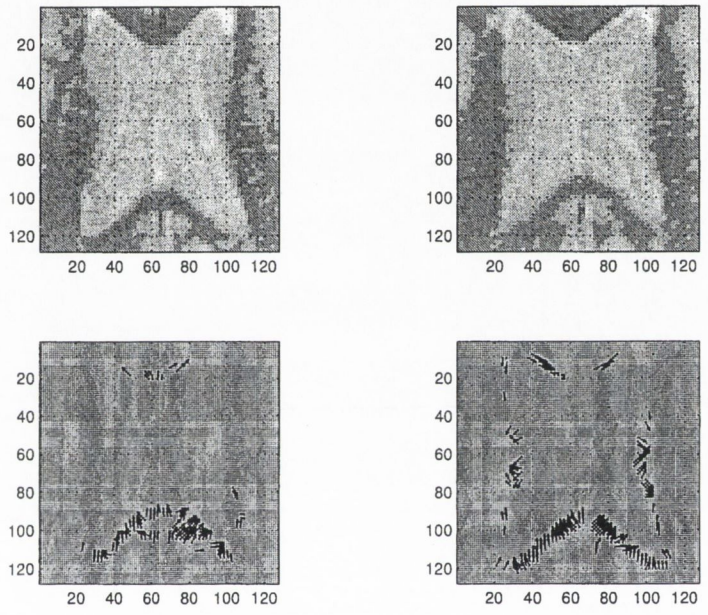

Figure 8.4: Slices #32 and #33 from the PD MRI volume (1st row) and the difference vectors estimates between them obtained by 2-D and 3-D ZNCC (2nd row.)

The grey-level data shown are two slices from the MRI PD volume given in Fig. C.21, Appendix C. When applied on the actual grey-level intensities, the 2-D ZNCC (7.28) although took only 2.54 s to compute did not provide good estimates of the difference between the slices. While the 3-D ZNCC (8.5), although slower (11.29 s.) produced better vector estimates. The parameters used were $N_b = 5$, $d_{\max} = 5$, $\Delta = 2$ and $d_b = 3$ for the 3-D blocks. The vectors in Fig. 8.4 are shown on a decimated lattice for every second node of the first image.

Block-based methods are widely used for motion estimation in video sequences. They are primarily based on two consecutive frames only [48,85]. While estimating differences between nodes from an intrinsically 3-D data based on their grey-level intensities benefits from the usage of 3-D 'thin-plate' ($d_b = 3$) blocks.

8.3 KULLBACK-LEIBLER DISTANCE (KLD) MINIMIZATION

All the difference estimation methods presented so far are unable to detect differences between slices of textural mixtures, especially when the $\boldsymbol{\theta}_h$ parameters of the stationary textures are similar. So far the *a priori* information available in a supervised segmentation context, namely the codebook, has not been referred to. It can be used, however, to calculate a probability mass function (p.m.f.), over the class variables for a grey-level 3-D block. Then, a measure of similarity between two p.m.f.s of two 3-D blocks, can be defined. In this way the difference estimation is based on the grey level data and the codebook.

Hence, a 3-D block-based method which uses the Kullback-Leibler Distance (KLD) as a similarity measure between the class p.m.f.s is proposed.

8.3.1 Class Probability Mass Function

As stated in Section 3.4.1 any block $\mathbf{X}$ of stationary GMRF can be specified by its joint probability (3.13). Given the codebook $\mathbf{C}$ (Section 3.5.2) the conditional probability of $\mathbf{X}$, given that it belongs to the homogeneous texture class c is given by:

$$P(\mathbf{X}|c) = \frac{\sqrt{\det(\mathbf{B}_c)}}{\sqrt{(2\pi\sigma_c^2)^{N_T}}} \exp\left[-\frac{(\mathbf{X}-\boldsymbol{\mu}_c)^T\mathbf{B}_c(\mathbf{X}-\boldsymbol{\mu}_c)}{2\sigma_c^2}\right], \tag{8.6}$$

where the c-th codebook entry is $[\mu_c, \sigma_c^2, \boldsymbol{\theta}_c^T]^T$ and $\mathbf{B}_c$, depending on $\boldsymbol{\theta}_c$, is the interaction matrix (3.14) for class c. Under the toroidal boundary assumption, the evaluation of the conditional probability (8.6) can be performed efficiently using the level three circulant algorithms given in Section 4.3. This is a very important computational saving in a block-matching technique where the similarity measure computation is needed for all candidates within a extensive search.

The conditional probability of c given the data, i.e. the *class p.m.f.*, $p(c|\mathbf{X})$, can

be defined from normalization of the $P(\mathbf{X}|c)$ quantities across all classes $c \in \{1, \ldots, T\}$:

$$p(c|\mathbf{X}) \triangleq \frac{P(\mathbf{X}|c)}{\sum_c P(\mathbf{X}|c)}. \tag{8.7}$$

A uniform prior on $\mathbf{C}$ is assumed.

8.3.2 Kullback-Leilbler Similarity Measure

The *Kullback-Leibler Distance* (KLD)- also known as the relative entropy- between two p.m.f.s, has been successfully used as a texture similarity measure [24]. The KLD between the p.m.f.s f and h is evaluated as [20]:

$$KLD(f\|h) = \sum_i f_i \log \frac{f_i}{h_i}. \tag{8.8}$$

The KLD is not commutative. For our purposes, f represents the current, and h- the candidate block p.m.f.s.

To find the best match t_z for the node of interest t among all candidates t' in slice $k_z = k + z$, the class p.m.f.s $p = \{p_c\}, c \in \{1, \ldots T\}$ and $p' = \{p'_c\}, c \in \{1, \ldots T\}$ for the 3-D blocks centred at t and t' respectively are computed for all classes via (8.7). The $KLD(p\|p')$ measures the similarity between the class p.m.f.s as shown in Fig. 8.5. For the example shown in the figure, the KLD will reach its minimum for the p.m.f. denoted by p^*.

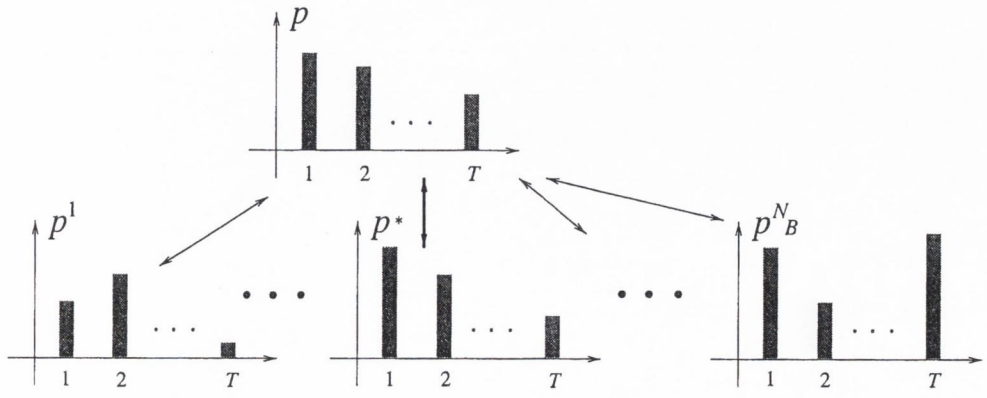

Figure 8.5: Comparing the class p.m.f.s

Therefore, the best match is selected to be the one which minimizes the KLD between p and the p.m.f.s for all N_B possible candidate blocks (Fig. 8.5). The best match for the node t is given by:

$$t_z = \arg\min_{t'} KLD(p||p'). \tag{8.9}$$

8.3.3 Simulation Results

Difference estimation

The KLD method is able to estimate the difference for the synthesized textured volumes directly by working on the grey level data, unlike the methods in Chapter 7. This is illustrated in Figs. 8.6, 8.7 and E.20 (Appendix E).

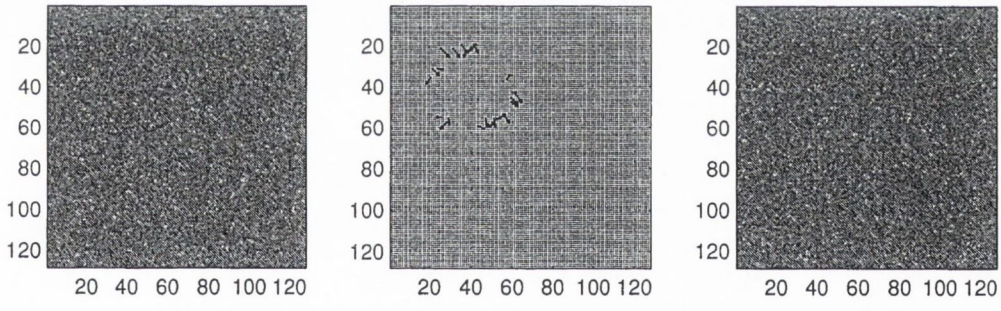

Figure 8.6: Grey level data and the estimated difference field between them, using KLD minimization. Slices #2 and # 3 from the 'skewed' cylinder volume (Fig.C.11).

The difference is estimated for every second node of the image lattice. The parameters used were $L_b = 7$ $d_b = 3$, $d_{\max} = 5$, $\Delta = 1$. Computational time was 23.77 s.

A volume that is quite difficult to segment was created using three classes masks given in Fig. C.13. The grey level volume of size $128 \times 128 \times 11$ obtained using third order GMRFs is given slice by slice in Fig. C.14. The parameters of the stationary textures used are given in Table 5.3. The estimated difference using the KLD minimization between slices #6 and #7 of the volumes is shown in Fig. 8.7.

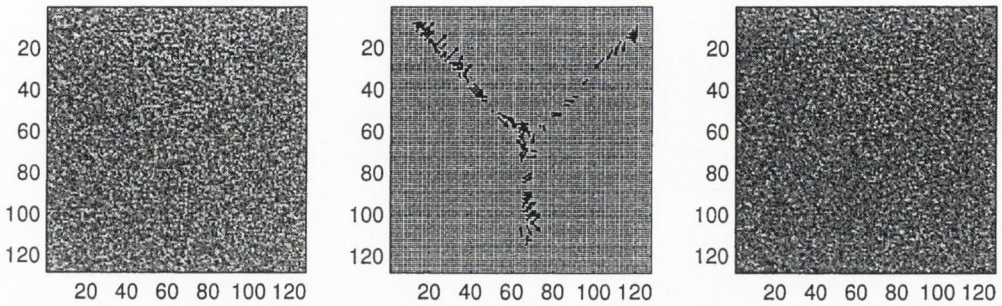

Figure 8.7: Grey level data and the estimated difference field between them, using KLD minimization. Slices #6 and #7 from the three textures volume.

The parameters used were as in the previous example and the computational time time was 47.09 s

Compensated segmentation results

The 3-D compensated RUR segmentation for the middle slice of the 'skewed' cylinder volume, composed of two second order textures with similar $\boldsymbol{\theta}_h$ and different $\boldsymbol{\theta}_v$ is shown in Fig. 8.8.

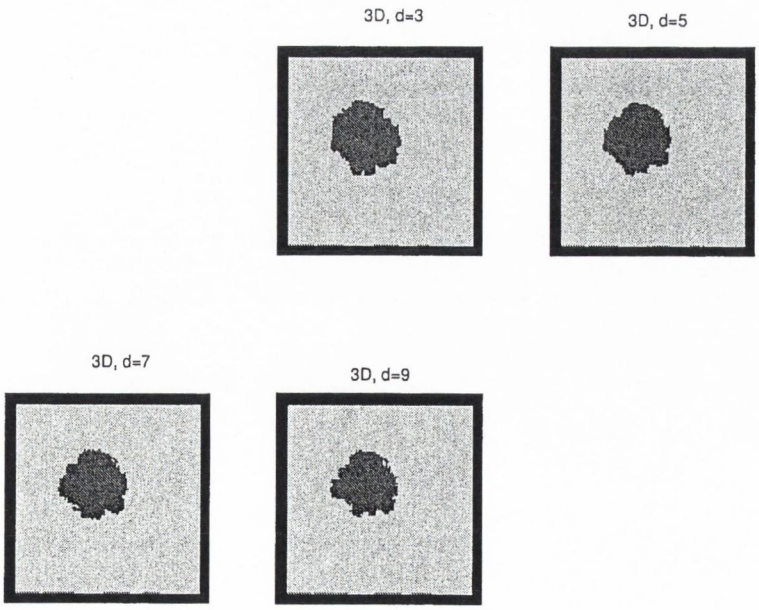

Figure 8.8: 3-D RUR compensated segmentation for different ω_t^c depths, slice #6 of the 'skewed' cylinder volume. KLD minimization-based difference field estimation.

The KLD minimization approach was used to produce $\widehat{\mathbf{D}}$. The difference vectors were estimated backwards and forwards using 3-D blocks of size $7 \times 7 \times 3$ for all possible pairs of slices k and k_z, $z = \pm 1$. Hence, the maximum possible depth for ω_t^c for slice #6 is $d = 9$. The quality of the compensated segmentation for the middle slice is similar for all depths and the object cross-section shape has been preserved as illustrated in Fig. 8.8. The compensated results using the KLD-based estimation $\widehat{\mathbf{D}}$ for all slices are given in Figs. E.21 -E.23. The performance parameters for all slices of the volume are listed in Table 8.1, confirming the superiority of the compensated RUR algorithm over the non-compensated one.

Table 8.1: 3-D compensated RUR segmentation of the 'skewed' cylinder volume: misclassification δ [%], and elapsed time [s]. Difference field estimation via KLD minimization.

Depth	mean	Slice #									el.time	
	δ	2	3	4	5	6	7	8	9	10	Tot.	Per sl.
$d = 3$	**2**	2.24	1.48	1.4	1.74	1.85	1.73	2.73	2.36	3.18	11.05	**1.23**
$d = 5$	**1.34**	-	1.37	0.98	0.99	1.12	1.3	1.84	1.75	-	19.73	**2.82**
$d = 7$	**1.28**	-	-	1.13	0.87	1.13	1.32	1.93	-	-	23.34	**3.81**
$d = 9$	**1.5**	-	-	-	1.16	1.57	1.78	-	-	-	22.24	**7.41**

The performance measured by the mean percentage misclassification for each depth of ω_t^c is close to the performance of the compensated segmentation using the exact difference field and marks an improvement over the non-compensated segmentation (Fig. 8.9).

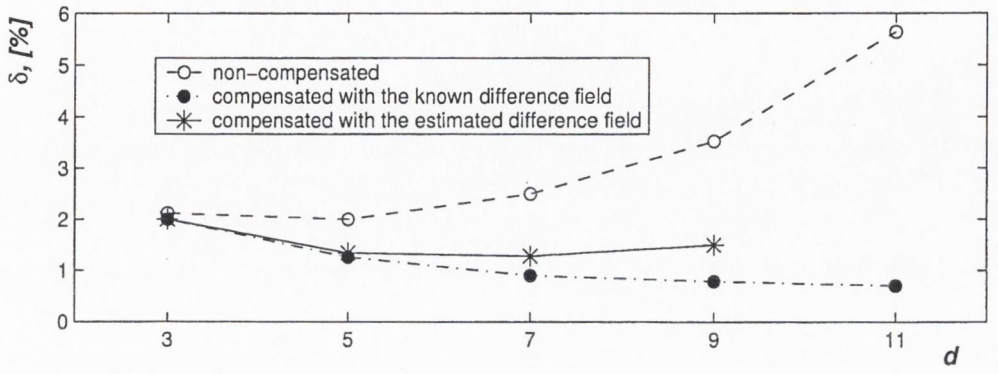

Figure 8.9: Mean misclassification for the 'skewed' cylinder volume with and without compensation. Difference field estimation via KLD minimization .

Next, the third order three textures volume was segmented via the non-compensated RUR approach, and then via compensated segmentation using the known and the estimated via KLD minimization difference field algorithms. The non-compensated segmentation fails to segment the data, particularly at the borders between the stationary textures due to the presence of inter-slice differences (Figs. E.24-E.26). The difference field estimated via the KLD minimization technique is used to compensate for these differences. It results in more accurate boundaries between the segmentation areas in $\mathbf{L}$ as illustrated in Fig. 8.10 for the middle slice of the volume.

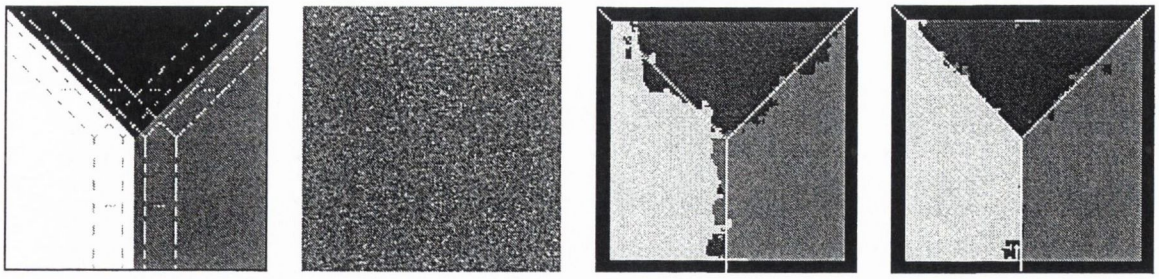

Figure 8.10: Ground truth generation mask, slice #6 from the data set, 3-D non-compensated (depth 11) and 3-D compensated segmentation. Difference estimation via KLD minimization.

The segmentation results for all slices and depths are given in Figures E.27 - E.29 using the known difference field and in Figures E.30 - E.32 using the estimated

difference field respectively. The elapsed times and the misclassification of each method and for each processed slice are listed in Table 8.2.

Table 8.2: RUR segmentation: misclassification δ [%], and elapsed time per slice [s]

Algorithm												
Depth	**mean**	**Slice #**									**el.time**	
	δ	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**Tot.**	**Per sl.**
Non-compensated segmentation												
$d=3$	**8**	9.14	6.49	8.37	8.7	6.28	9	8	8.5	8.2	55.39	**6.15**
$d=5$	**4.58**	-	5.1	4.4	4.7	4.25	5.23	4.76	3.6	-	57.1	**8.15**
$d=7$	**4.37**	-	-	4.28	4.45	5.16	4.25	3.7	-	-	58.54	**11.71**
$d=9$	**5.34**	-	-	-	4.95	5.56	5.5	-	-	-	53	**17.67**
$d=11$	**6.69**	-	-	-	-	6.69	-	-	-	-	63.5	**63.5**
Compensated with the known difference field												
$d=3$	**8**	8.9	8.59	8.12	8.2	5.6	7.9	7.15	8.9	8.7	42.1	**4.67**
$d=5$	**3.13**	-	4	2.9	2.47	2.35	3.6	3.56	3	-	55.9	**7.98**
$d=7$	**1.24**	-	-	1.12	0.98	1	1.58	1.52	-	-	54.7	**10.94**
$d=9$	**0.79**	-	-	-	0.7	0.78	0.9	-	-	-	46.9	**15.63**
$d=11$	**0.62**	-	-	-	-	0.62	-	-	-	-	64.7	**64.7**
Compensated with the estimated difference field												
$d=3$	**8.03**	8.98	8.58	8.12	8.15	5.64	7.96	7.15	8.9	8.76	40.51	**4.5**
$d=5$	**4.84**	-	5.39	3.85	4.64	3.57	4.67	5.99	5.74	-	63.5	**9.07**
$d=7$	**3.87**	-	-	3.1	3.67	3.48	3.9	5.22	-	-	61.38	**12.28**
$d=9$	**3.8**	-	-	-	3.58	3.24	4.6	-	-	-	55.26	**18.42**
$d=11$	**3.33**	-	-	-	-	3.33	-	-	-	-	67.84	**67.84**

The mean misclassification for each depth of ω_t^c (Fig. 8.11), using the presented method, decreases when using deeper volumes. Thus, it follows the trend of the compensated segmentation with the exact **D** unlike the deteriorating trend with the non-compensated algorithm. Note that in order to build a full $d = 11$ compensated computational volume, ω_t^c, the unavailable estimates for the difference from slice #1 to #2 and, again, from #10 to #11, were assumed zero.

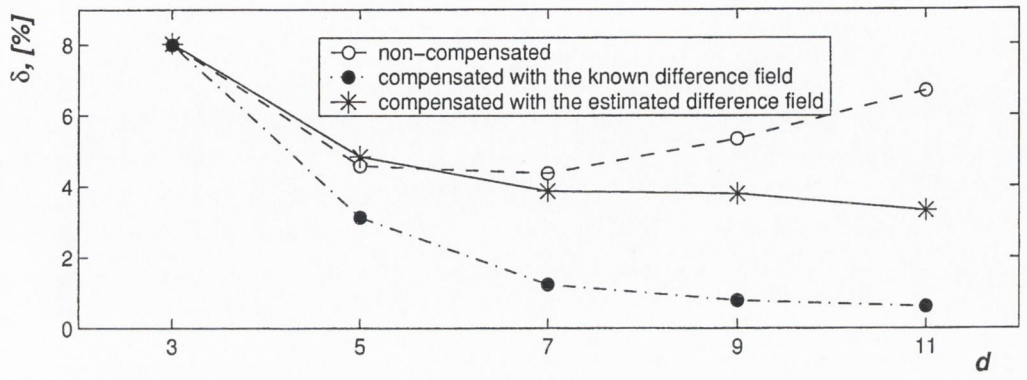

Figure 8.11: Mean misclassification for the three textures volume with and without compensation. Difference field estimation via KLD minimization.

The KLD minimization method was able to estimate and successfully compensate for the inter-slice variations in the texture mixture, using only $\mathbf{X}$ and $\mathbf{C}$. The KLD-based method was reported in [73].

8.4 CASE STUDY: MAGNETIC RESONANCE IMAGING (MRI)

3-D block-based methods for difference estimation were tested on real 3-D MRI volumetric images. As explained in Section 2.5, the T2-weighted and PD modalities present the most challenging data for the anatomical segmentation problem. The original T2 and PD volumes of size $512 \times 512 \times 9$, being transverse MRI scans of a normal human brain, were downloaded from the Whole Brain Atlas web site [80]. They are shown in Figs. C.18 and C.19 respectively, where the original slice numbering is preserved. The sampling resolution is 1 mm in each slice and the slice thickness is 5 mm. The poorer resolution in the third dimension encourages strong inter-slice non-stationarity, and therefore, a significant inter-slice difference. Subvolumes of size $128 \times 128 \times 9$ were selected (Figs. C.20 and C.21) and a two class problem defined: namely segmentation of the ventricles by discrimination between the white and grey matter tissues.

A second order 3-D GMRF model for each tissue was assumed and the codebook for the two test volumes were estimated (Section 5.2) from manually selected regions of size $16 \times 16 \times 5$ for the white and $32 \times 32 \times 9$ for the grey matter. Note that only relatively small samples of nominally stationary texture were available for the parameter estimation procedure, thus affecting the estimation accuracy.

The codebook entries listed in Tables 8.3 and 8.4 provide important information for the choice of a difference estimation and segmentation method.

Table 8.3: Codebook entries for the T2 MRI test volume.

μ	σ	θ_0	θ_1	θ_2	θ_3	θ_4	θ_5	θ_6	θ_7	θ_8
247.76	1.3015	0.4231	0.4974	0.06	−0.1889	−0.2218	−0.0047	−0.0216	−0.0450	−0.0165
100.02	1.7671	0.5029	0.4878	0.0339	−0.2376	−0.2529	−0.0171	−0.0127	−0.0083	−0.0041

Table 8.4: Codebook entries for the PD MRI test volume.

μ	σ	θ_0	θ_1	θ_2	θ_3	θ_4	θ_5	θ_6	θ_7	θ_8
197.82	1.3065	0.4847	0.4971	−0.0687	−0.2262	−0.2613	0.0298	0.0167	0.0303	0.0169
152.11	1.6103	0.4908	0.4959	0.0244	−0.2238	−0.2625	−0.0103	−0.0021	−0.0079	−0.0091

The normalized Euclidean distances between the codebook entries are as follows: for the T2 data $D_{NE} = 0.3616$ using the whole $\boldsymbol{\theta}$ and $D_{NE} = 0.3595$ when only the $\boldsymbol{\theta}_h$ were used. For the PD data these are $D_{NE} = 0.0656$ and $D_{NE} = 0.0551$ respectively. For the two tissue types using T2 modality, it can be observed (Table 8.3) that the most significant difference is contributed by the mean parameter. This suggests that a 2-D segmentation, or even a simple thresholding of single 2-D slices, will perform very well. On the other hand, in a 3-D non-compensated segmentation, the inter-slice difference would affect the quality of the result. To estimate this difference, a simple 2-D ZNCC will be able to track the ventricle cross-section shape evolution.

The codebook entries for the PD data (Table 8.4) also differ significantly in the mean value, but the difference is not as great as for the T2 data, which can

be observed by comparison of the grey-levels for both test volumes and also the respective D_{NE}s with the use of $\boldsymbol{\theta}$. Hence, segmenting this volume would be a more difficult task than the T2 one. This fact, combined with the observation that the two tissue models possess similar $\boldsymbol{\theta}_h$ and differ more markedly in their $\boldsymbol{\theta}_v$ parameters, suggests that a 3-D segmentation is worth considering in this case. The data properties suggest that a 3-D block matching should be favoured in comparison to a 2-D one.

To verify these hypotheses, the test volumes were segmented using 2-D and 3-D non-compensated approaches. 2-D, 3-D ZNCC and KLD methods were applied to difference estimation and the resulting $\widehat{\mathbf{D}}$ was used for 3-D compensated segmentation.

8.4.1 Non-compensated 2-D and 3-D RUR Segmentation

The T2 data were segmented slice by slice independently using the 2-D RUR algorithm with $n^d = 16$, a second order GMRF, and a computational window size of $n = 15$. The segmentation was quite satisfactory as illustrated by Fig. E.39, although some small areas were not segmented well, for example the left area of white matter in slices #28-#31 and #36.

The 3-D non-compensated 'thin-plate' RUR segmentation results capture the shape of the ventricle cross-sections better (Fig.E.41). The operational parameters used were: 2nd order 3-D GMRF, $n = 15$, $d = 3$, $n^d = 4$, $d^d = 4$. Increasing the depth of ω_t results in deterioration of the segmentation results (Figs. E.42 and E.43).

The results for slices #31 and #32 are illustrated in Fig. 8.12 and Fig. 8.13 respectively.

The PD data volume was segmented independently slice by slice, via the 2-D RUR algorithm with $n^d = 16$, $q = 2$ and $n = 15$. The segmentation results obtained were good (Fig.E.40), although small regions of the white matter were missing, for

instance in slices #29, #30, #33 and #36.

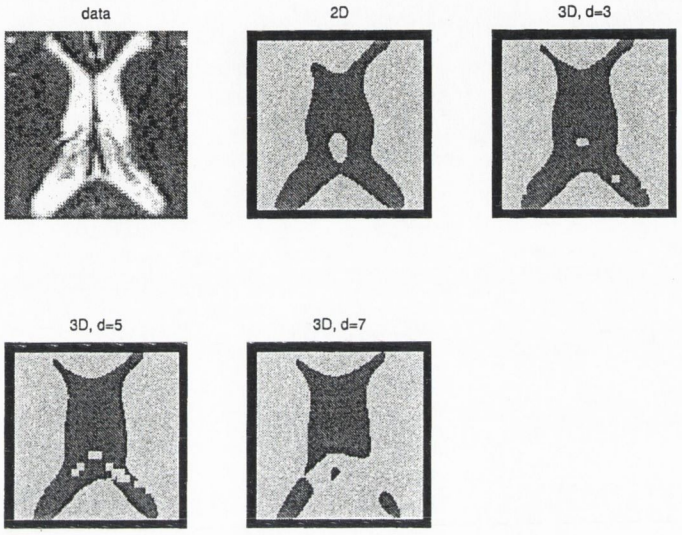

Figure 8.12: Non-compensated RUR segmentation of slice #31 from the T2 MRI volume.

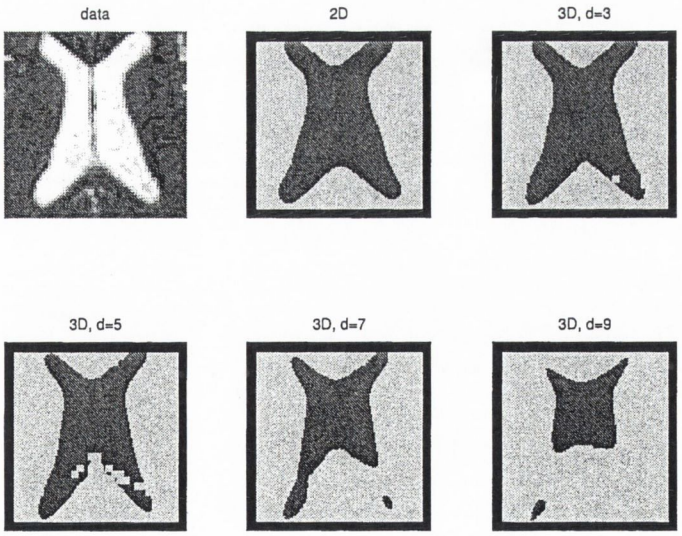

Figure 8.13: Non-compensated RUR segmentation of slice #32 from the T2 MRI volume.

The 3-D RUR non-compensated results with $q = 2$, a 3-D GMRF, $n = 15$, $d = 3$, $n^d = 16$ and $d^d = 4$ were able to detect correctly some missing regions, but

increasing d resulted in the appearance of some holes in the label maps (Fig.E.44 - E.46).

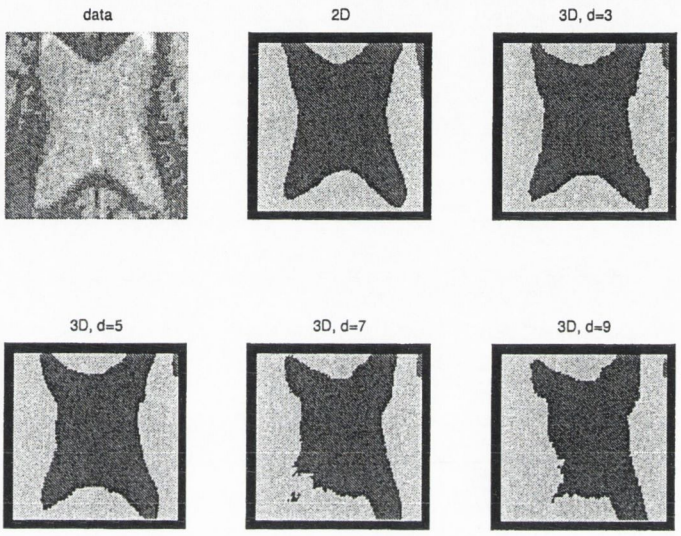

Figure 8.14: Non-compensated RUR segmentation of slice #32 from the PD MRI volume.

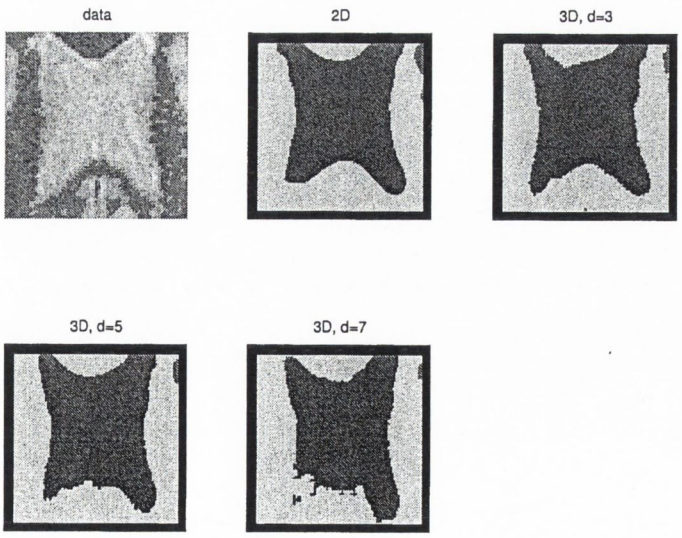

Figure 8.15: Non-compensated RUR segmentation of slice #33 from the PD MRI volume.

These observations are illustrated for slices # 32 and # 33 in Figs. 8.14 and 8.15 respectively.

8.4.2 Difference Estimation

To estimate the shape evolution along the third dimension, block matching techniques were used to obtain $\widehat{\mathbf{D}}$. For both the T2 and PD volumes, the 2-D and 3-D ZNCC method as well as the KLD technique, were tested. The results for the forward difference estimates between slices #32 and #33 are shown in Figs.8.16-8.18 for the T2 modality, and in Figs.8.4 and 8.19 for the PD modality. More results are given in Section E.4.1.

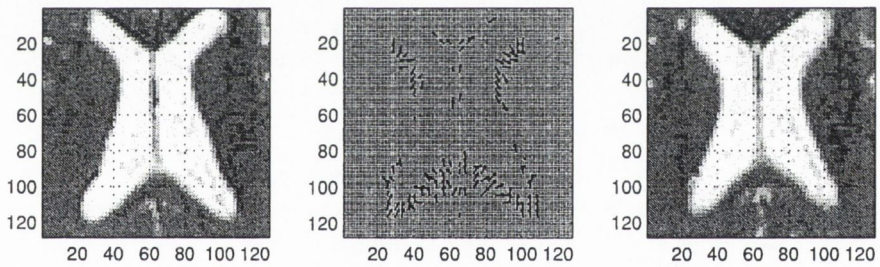

Figure 8.16: T2 MRI: slices 32 and 33 and the difference field between them, estimated via 2-D ZNCC.

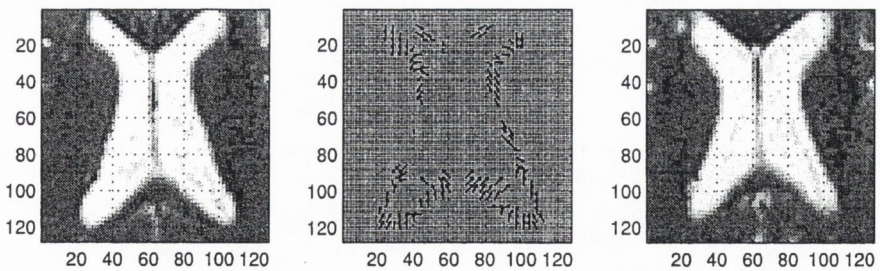

Figure 8.17: T2 MRI: slices 32 and 33 and the difference field between them, estimated via 3-D ZNCC.

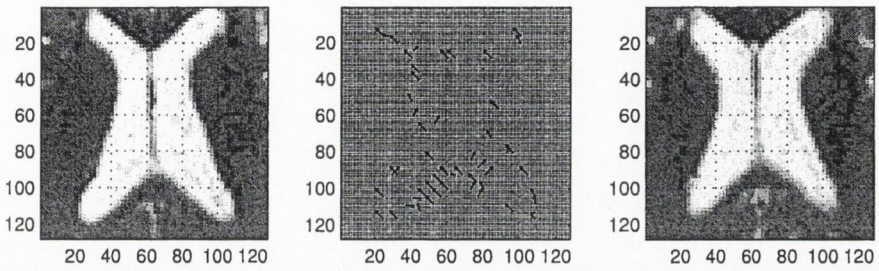

Figure 8.18: T2 MRI: slices 32 and 33 and the difference field between them, estimated via KLD minimization.

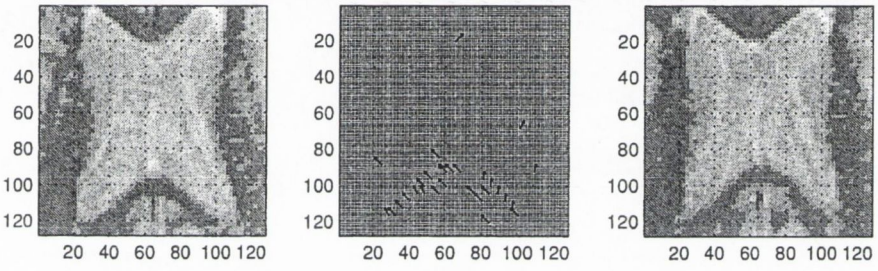

Figure 8.19: PD MRI: slices 32 and 33 and the difference field between them, estimated via KLD minimization.

Table 8.5: Difference field estimation between slices #32 and #33 from the MRI volume: elapsed time, [s].

Algorithm	**T2**	**PD**
2-D ZNCC	2.54	0.82
3-D ZNCC	11.29	5.2
KLD	10.64	8.30

The block parameters used were $L_b = 5$, $d_{\max} = 5$ and $\Delta = 2$ and every fourth node from $\mathbf{\Lambda}$ was considered. The elapsed times for this pair of slices for both modalities using the three algorithms are summarized in Table 8.5.

The 2-D ZNCC technique produces good quality results for the T2 data for the shortest computational time, therefore it was used to obtain $\widehat{\mathbf{D}}$ for the whole volume in the T2 modality. This method did not work very well for the PD data, while the 3-D ZNCC worked well, even for a higher computational cost.

8.4.3 Compensated Segmentation

The estimated difference field was used to construct ω_t^c and compensated RUR segmentation was performed. All label maps are given in Figs.E.47- E.49 for the T2 and in Figs. E.50- E.52 for the PD data. The results for slices # 31 and #32 from the T2 volume are given in Fig.8.20 and Fig.8.21 respectively.

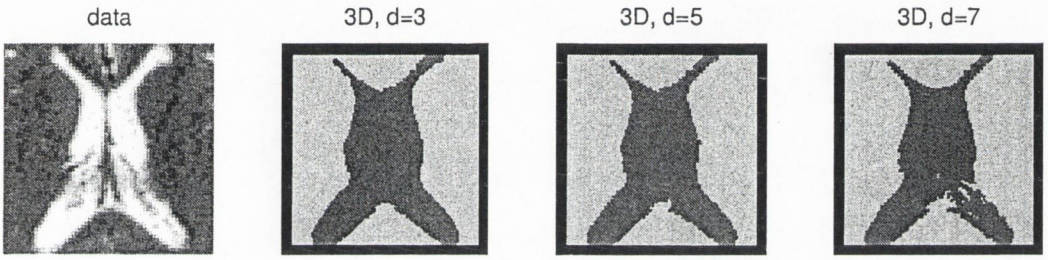

Figure 8.20: Compensated RUR segmentation of slice #31 from the T2 MRI volume. $\widehat{\mathrm{D}}$ estimated via the 2-D ZNCC.

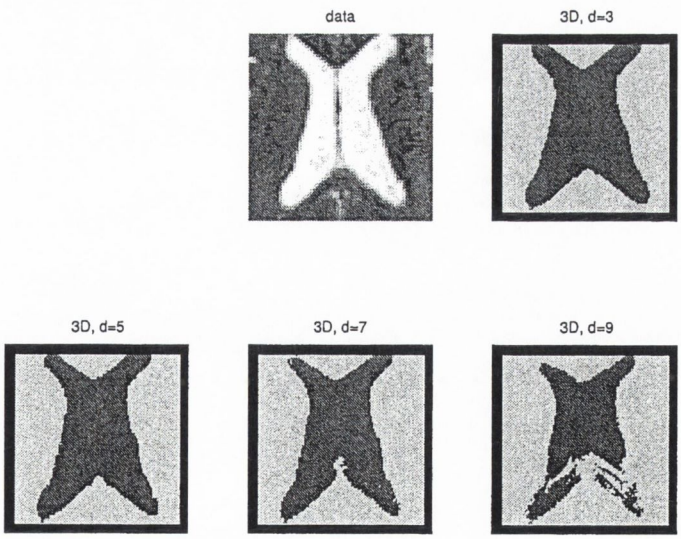

Figure 8.21: Compensated RUR segmentation of slice #32 from the T2 MRI volume. $\widehat{\mathrm{D}}$ estimated via the 2-D ZNCC.

Figures 8.22 and 8.23 illustrate the compensated results for slices #32 and #33 from the PD volume.

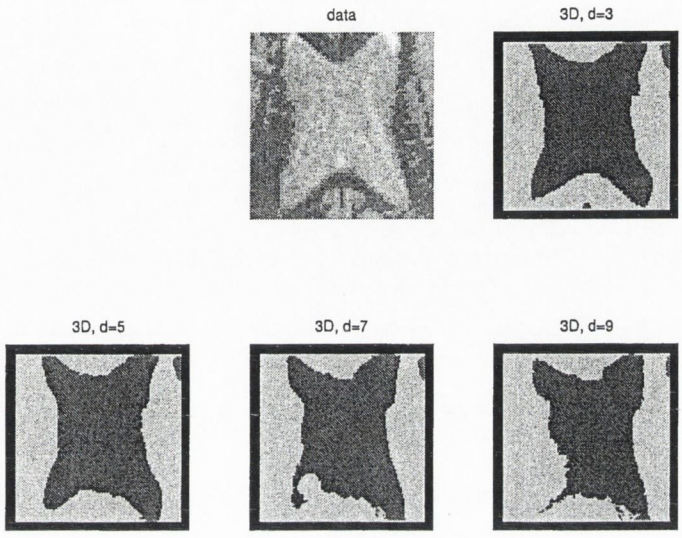

Figure 8.22: Compensated RUR segmentation of slice #32 from the PD MRI volume. $\widehat{\mathrm{D}}$ estimated via the KLD minimization.

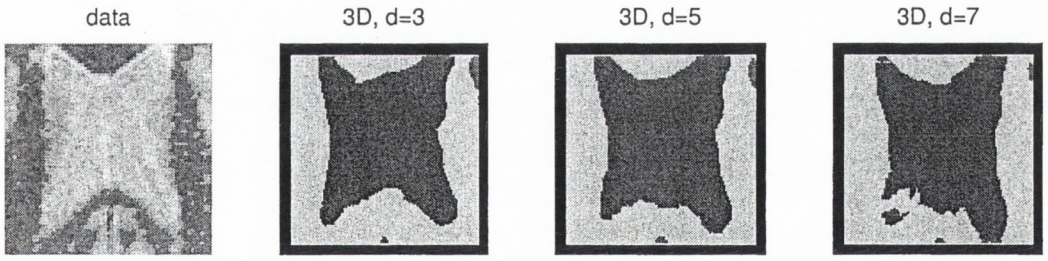

Figure 8.23: Compensated RUR segmentation of slice #33 from the PD MRI volume. $\widehat{\mathrm{D}}$ estimated via the KLD minimization.

The elapsed times for both non-compensated and compensated results are summarized in Table 8.6. Note that the times reported for the T2 include the difference estimation via the 2-D ZNCC, while the times for the PD are only for the compensated segmentation after $\widehat{\mathrm{D}}$ was obtained via the KLD minimization, which took about 15 s per pair of slices.

Table 8.6: Supervised segmentation of the MRI data. Elapsed times [s].

Algorithm				
Modality	**T2**		**PD**	
Depth	**el.time**		**el.time**	
	Tot.	**Per sl.**	**Tot.**	**Per sl.**
Non-compensated segmentation				
$d = 3$	48.05	**6.86**	20.53	**2.9**
$d = 5$	50	**10**	24.68	**4.9**
$d = 7$	69.91	**23.3**	24.17	**8.06**
$d = 9$	91.96	**91.96**	26.20	**26.20**
Compensated segmentation				
$d = 3$	143.28	**20.47**	23.24	**3.32**
$d = 5$	153.02	**30.6**	34.52	**6.9**
$d = 7$	155.63	**51.88**	34.48	**11.49**
$d = 9$	299.94	**299.94**	36.58	**36.58**

These results lead to the following conclusions:

1. The label maps, after segmenting PD MRI data, are worse than those for the T2 data. The task of segmenting PD is more difficult than T2 data as underlined by the respective D_{NE} using $\boldsymbol{\theta}$.

2. The 2-D segmentation results are already of good quality especially for the T2 data, where the contrast between the two tissues is bigger. There are some small areas from the images, however, which are not segmented well.

3. 3-D segmentation performs better for segmenting these small areas due to the extra data available, which increases the parameter estimation accuracy. The benefit of using 3-D processing is realized at small depths, while for greater the segmentation accuracy deteriorates.

4. The 3-D non-compensated segmentation is affected substantially by the presence of 3-D shape evolution effects.

5. Fast 2-D block-based techniques (ZNCC) for difference estimation work well for the T2 data, but give unsatisfactory results for the PD data. Computationally more

expensive 3-D methods (KLD minimization) are applicable for the latter.

6. The compensated segmentation is able to preserve the quality of the label field obtained using deeper ω_t^c. Thus, it outperforms the non-compensated segmentation at greater depths, especially in segmenting small anatomical structures. It provides more stationary data for the GMRF parameter estimation step by successfully tracking the ventricles' shape evolution.

8.5 GUIDED COMPUTATIONAL VOLUMES VIA ENTROPY MINIMIZATION

A disadvantage of the difference estimation and compensation methods presented so far is the inability to determine optimal depth for ω_t^c. It has been concluded that generally the bigger d, the better. The protection of the texture stationarity, however, is necessary only for those non-compensated computational volumes ω_t, where it is not valid. For some nodes, stationary 'thin-plate' ω_t is enough to obtain reliable GMRF estimates, which leads to a satisfactory non-compensated segmentation. The $\widehat{\mathbf{D}}$ obtained by these methods had many zero entries for such nodes belonging to large homogeneous texture volumes, but this sparsity has not been exploited. The algorithms considered so far use fixed d for all ω_c^t, while in reality the depth should be responsive to the local changepoint activity. It should be generally greater in areas of high activity, while minimal ($d = 3$) elsewhere. Using ω_t^c with the optimal depth for each node can lead to significant computational speedup.

We propose a block-matching scheme for estimating of the optimal d and construction of ω_t^c, which embraces this principle.

8.5.1 Entropy

The method is based on the requirement to keep the entropy associated with the class p.m.f. of a growing 3-D block to a minimum.

The (Shannon) entropy of a discrete random variable X is defined as:

$$H(X) = -\sum_{x \in X} p(x) \log_2[p(x)], \tag{8.10}$$

where $p(x)$ is the probability that X is in the state (realization) x. The entropy of the class p.m.f. (8.7) is given by:

$$H(C|\mathbf{X}) = -\sum_{c=1}^{T} p(c|\mathbf{X}) \log_2[p(c|\mathbf{X})]. \tag{8.11}$$

8.5.2 Difference Compensation

Within the proposed entropy minimization framework, difference estimation and compensation are performed simultaneously.

Consider a block B_t of size $N_b \times N_b \times d_b$, $N_b = 2L_b + 1$ centred at t defined by (see eqn.(7.4)):

$$\begin{aligned} B_t(d_b) = & \bigcup_{i=0}^{(d_b-1)/2-1} \left\{ t + \sum_{j=0}^{i} \widehat{\mathbf{d}}_{t_j}(1) + (\Delta i, \Delta j, 0) \right\} \\ & + \bigcup_{i=0}^{(d_b-1)/2-1} \left\{ t + \sum_{j=0}^{i} \widehat{\mathbf{d}}_{t_{-j}}(-1) + (\Delta i, \Delta j, 0) \right\} \\ & + t + (\Delta i, \Delta j, 0), -L_b \leq \Delta i \leq L_b, -L_b \leq \Delta j \leq L_b, \end{aligned} \tag{8.12}$$

where $\widehat{\mathbf{d}}_{t_i}(1)$ is the difference vector at each center of the 'stacked' 2-D blocks of size $N_b \times N_b$. It is equivalent to the difference vector estimate used in previous methods. Note that this estimate does not need to be obtained *a priori* to the construction of the block. Instead, it can be computed from the minimum entropy trajectory as shown next. $t_i = (i_i, j_i, k_i)$ and $i_i = i + \delta i$, $j_i = j + \delta j$, $k_i = k \pm 1$.

This block is the current best block of depth d_b for t. Let the class p.m.f. entropy for t based on the data from the block $B_t(d_b)$ be $\widehat{H}_{d_b}(C|B_t(d_b))$. Initially $d_b = 3$,

and $\widehat{\mathbf{d}}_t$ are zero., i.e. the initial block $B_t^{(0)}(3)$ is a 'thin-plate' non-compensated. The difference compensation is achieved in two stages:

1. Search for a better block $B_t^{(1)}(3)$ with a smaller entropy

$$H^{(1)}(C|B_t^{(1)}(3)) < H^{(0)}(C|B_t^{(0)}(3)).$$

If no better block is found, stop. Else:

2. Find the best block

$$B_t^* = \arg\min_{B_t^{(it)}} H^{(it)}(C|B_t^{(it)})$$

at depth d_b^*.

The block at each iteration $B_t^{(it)}$ is obtained from the block from the previous iteration $B_t^{(it-1)}$ via incrementing the depth by two ($d_b^{(it)} = d_b^{(it-1)} + 2$). This is done by adding 2-D blocks of size $N_b \times N_b$ in alternating forward and backward manner. The centres of these 2-D blocks lie within the search area, i.e. $-d_{\max} \leq \delta i \leq d_{\max}$, $-d_{\max} \leq \delta j \leq d_{\max}$ and $\delta k = k \pm (d_b - 1)/2$. Each 3-D block $B_t^{(it)}$ is constructed to give smaller class p.m.f. entropy than $B_t^{(it-1)}$, i.e. $H^{(it)}(C|B_t^{(it)}) \leq H^{(it-1)}(C|B_t^{(it-1)})$. This is equivalent to the estimation of the difference vectors $\widehat{\mathbf{d}}_{t^{(it-1)}} = t^{(it)} - t^{(it-1)}$, but is derived from the obtained minimum entropy trajectory $\tau = t^{(0)}, t^{(1)}, \ldots, t^{(it)}$. The procedure stops when the maximum possible depth being limited by the data dimension has been explored. The resulting best block B_t^* with depth d_d^* is the compensated computational volume ω_t^c for the node t, where $m = n = N_b$ and $d = d_b^*$.

Stage 1. detects the presence of difference and necessity for compensation and stage 2. provides the optimum depth ω_t^c for a successful compensated segmentation. This procedure is being invoked only for the nodes of the Λ_{RUR}, ensuring texture stationarity protection with the minimum of computation. Fewer nodes implies faster execution of the overall procedure.

8.5.3 Simulation Results

The entropy minimization technique is mostly suitable for highly textured data, where no clear 3-D object structure is present and all other methods for difference estimation when applied to the grey level data fail. Therefore, it has been tested on the 'skewed' cylinder and the three texture data volumes.

The performance of the 3-D entropy-based RUR algorithm is illustrated in Fig.8.24 for the middle slice of the 'skewed' cylinder volume. It is compared with the non-compensated and compensated versions with given **D** segmentation results with maximum depth. The algorithm parameters were $n = 15$, $q = 2$, $n^d = 16$, $d^d = 4$ and the depth of ω_t^c has been estimated automatically for each Λ_{RUR} node, thus adapting dynamically in order to maximize the membership (7.1) of the dominant texture class.

To compare the entropy-based method with compensated algorithm with a fixed d, an averaging along all possible depths for each slice is performed. The mean misclassification for each slice is compared with the values obtained from a single execution of the entropy method in Fig.8.25. Segmentation of all slices of the volume are given in Fig. E.53.

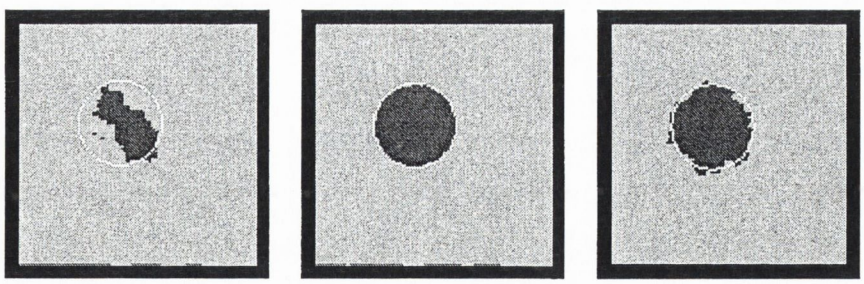

Figure 8.24: 3-D RUR segmentation of the two texture problem: slice #6. Non-compensated, compensated with the exact difference field ($d = 11$) and with the entropy minimization (variable d).

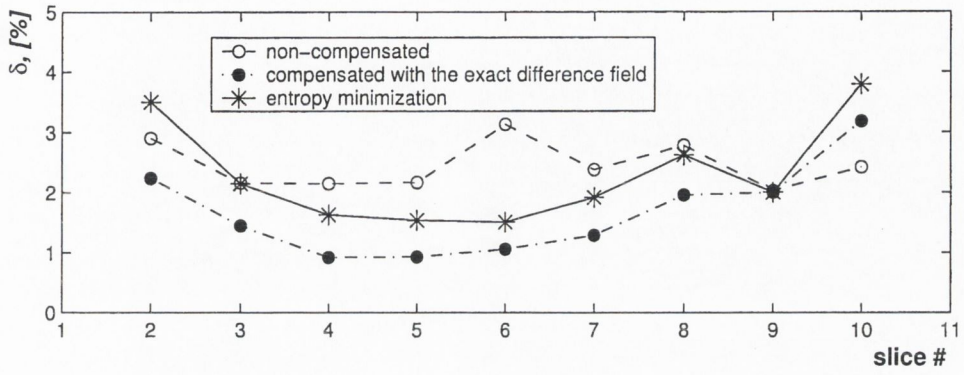

Figure 8.25: Average misclassification for the 'skewed cylinder'volume with and without compensation. Entropy-based construction of ω_t^c.

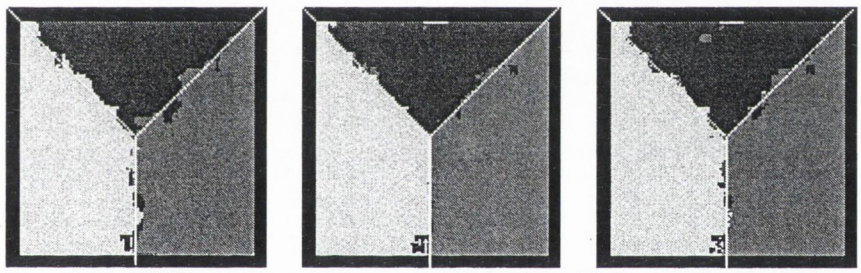

Figure 8.26: 3-D RUR segmentation of slice #6 with superimposed mask: non-compensated, compensated with the exact difference field ($d = 5$) and with the entropy minimization (variable d).

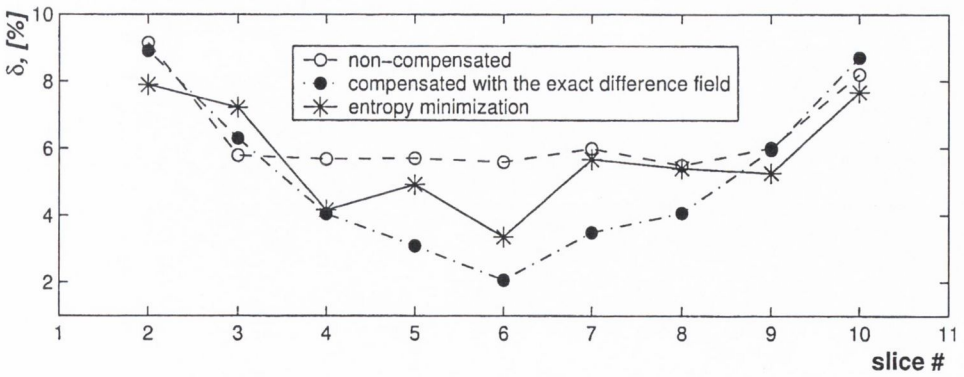

Figure 8.27: Average misclassification for the three textures volume with and without compensation. Entropy-based construction of ω_t^c.

It can be seen that the entropy method provides good quality results. Similar performance was reported in [74] for the task of segmenting the three texture third

order synthetic data. An illustration of the middle slice of the volume is given in Fig.8.26 and the misclassification curves are given in Fig.8.27. Segmentation of all slices of the volume are given in Fig. E.54.

The entropy-minimization method presents a unifying framework for simultaneous compensation and provides an automatic robust specification of the optimum depth d based only on the original grey-level data. It achieves good segmentation in highly textured volumes, which are extremely hard to segment using either 2-D or 3-D non-compensated segmentation. A disadvantage of the method is its relatively high computational complexity, even when the fast 3-D FFT based techniques for computations with circulants of level three (Section 4.3) are used. It requires computing the joint p.d.f of a 3-D block of data with progressively increasing dimensions.

8.6 CONCLUSIONS

This chapter presented novel methods for difference estimation and compensation of textured volumes. They use the properties of the original 3-D grey level data, the underlying GMRF model, and the codebook available in a supervised framework. These models are shown to segment the data without the preliminary 'thin-plate' segmentation used by the label-field-based methods (Chapter 7). Their performance has been demonstrated on synthetic volumes and real MRI data. Finally, a unifying framework which uses optimally compensated computational volumes without the explicit estimation of the difference field was presented. It is based on finding a trajectory through the node to be labelled, which has the property of minimal entropy. This framework has the ability to segment robustly textured volumes where the 3-D object structures cannot be captured by other methods.

Chapter 9

CONCLUSIONS AND FUTURE WORK

9.1 MAIN CONCLUSIONS

The development of supervised segmentation algorithms for textured images on 3-D lattices was the central theme of this thesis. Results demonstrating their effectiveness in segmenting a variety of computer generated and real MRI images were presented.

The overview of the methods for texture segmentation of volumetric imagery presented in Chapter 2 identified the areas where research efforts were needed. The task of segmentation is too complex and diverse and therefore, no universal solution is found yet. The area of 3-D segmentation, specifically, is a new and dynamically developing area with demands in various areas such as biomedicine, geology, material science, etc. Much research has been done in the area of image sequence processing: segmentation, restoration, retrieval etc. for many years, while the processing of 3-D still imagery has been developing only since recent years. Specifically, the segmentation of textured images on 3-D lattices was identified as an important task.

The advantages of using a model for homogeneous texture were shown. The problem of difference estimation in volumetric imagery was identified. While its relation to the problems of motion estimation, correspondence, pattern recognition and image registration is evident, the need for developing specific methods was concluded. Looking at potential applications, the nature of MRI data sets recommended them as a good real-world context for the methods.

A model-based approach was chosen for the segmentation algorithms in this work. The model for images on 3-D lattices, namely the 3-D GMRF, was presented in Chapter 3 as a systematic extension of the 2-D GMRF [14]. The flexibility for defining an MRF model via the local or global properties, established by the H-C theorem, enabled the selection of model with simple specification. The compact representation of a stationary 3-D GMRF via a parameter vector has been shown to suit the representation of a homogeneous texture. The model depends on a simple 3-D symmetric neighbourhood geometry within the image lattice. Selecting only pair-wise cliques, while restricting the generality of the model, has the ability to analyze and synthesize a large class of textures combined with the desirable property of simple and fast algorithmic implementation.

An elegant algorithm for synthesis of 3-D stationary GMRFs was developed in Chapter 4. The thyroidal boundary assumption and the 3-D GMRF neighbourhood topological properties yielded in a special structure of the joint interaction matrix of the model. The main theoretical results for circulant matrices of level three were presented. In particular, the diagonalization theorem has provided a basis for the development of the fast synthesis algorithm via the 3-D FFT. It has allowed the generation of many 3-D textures used for testing of the algorithms throughout the thesis. Synthesizing various textures with similar $\boldsymbol{\theta}_h$ and different $\boldsymbol{\theta}_v$ parameters was of a special benefit when studying the potential gain of a 3-D processing of such volumetric data in comparison to 2-D independent slice by slice one. While it had led to a very efficient method for generating samples from 3-D GMRFs, it has the disadvantage of being able to produce textures, which are not necessarily valid fields.

Therefore special attention had to be paid when selecting the GMRF parameters, so that the generated model had valid interaction matrix, necessary for the analysis.

In Chapter 5 the estimation of the parameters of a 3-D stationary GMRF via a LS technique, within a computational volume, was derived. The estimation accuracy was studied, and the applicability of the method for the task of classifying lattice nodes belonging to a predefined texture class was concluded. A 3-D segmentation method based on parameter estimation followed by a classification for each node of the lattice, namely the voxel-by-voxel (VBV) scheme, was proposed. The simple idea of using computational volume for that purpose has shown the benefit of an easy implementation producing good results on a smaller computational price in comparison to the popular Bayesian methods. The algorithm was shown to successfully segment volumetric images, for which the 2-D independent slice by slice segmentation failed. The main algorithm drawbacks, however, namely the increasing computational load with the data dimensions and the lack of connectivity in the resulting label field, has led to the development of improved methods.

Segmentation methods based on the VBV scheme, but working on decimated lattices was proposed in Chapter 6. These were the Majority Decision (MD), Unanimity Rule (UR) and the recursive UR (RUR). The simple idea of imposing connectivity in the label field *a posteriori* independently of the way it was produced has manifested itself as a successful alternative to the MRF priors or double MRF models used within the Bayesian framework [58]. The decimated lattice methods had overcome the VBV limitations and were able to produce better connected label maps for a smaller computational cost. The methods were tested on 3-D synthetic and more realistic temporal textures. It was concluded that the RUR method was the best method in respect to classification accuracy and speed. Hence, it was used in all later work.

A disadvantage of using simple parallelepiped shaped computational volume, however, in the case when third dimension shape evolution is present was identified. It has led to the motivation and development of compensated computational volume

methods.

These methods for evaluation of this phenomena and compensation of the computational volume were presented in Chapters 7 and 8. A formal framework of modelling the third dimension shape evolution by introducing the difference field was developed in Chapter 7. Approaches originating from the related areas was modified and used for enhanced segmentation. These are the interest points detection and matching, and 2-D cross-correlation of blocks from an initial label field obtained after a 'thin-plate' non-compensated RUR segmentation. These methods, although based on ideas from related areas, were adapted to reflect the specifics of 3-D label field input data. The resulting compensated segmentation was shown to improve the quality of the results obtained by the non-compensated approaches in Chapter 6. The initial 'thin-plate' segmentation and estimation stages were shown to be sources of accumulating error. Hence, new 3-D methods based solely on the image data and the pre-determined texture codebook were developed in Chapter 8. 3-D cross-correlation and Kullback-Leibler Distance (KLD) minimization methods were shown to work well on both synthetic and real MRI images. Finally, a unified approach for simultaneous difference estimation and compensation via entropy minimization was presented and applied.

9.2 DIRECTIONS FOR FUTURE WORK

The GMRF model-based algorithms discussed in this work involve the selection of some parameters in advance of the processing, such as the GMRF order and the computational volume size. The automatic choice of the depth parameter was considered in the entropy-minimization approach, but automatic selection is an unsolved task for the other parameters. For instance, investigation of the optimum model order, depending on the application, is an important task. The GMRF synthesis algorithm's potential for synthesizing natural looking 3-D textures (the temporal textures, for instance) via selecting the right order is worth investigation.

The main concern addressed in this work was to protect the texture stationarity assumption within the computational volume via adapting it in the third dimension. The non-stationarity existing within each slice was addressed by the decimated lattice approaches in methods like the RUR scheme, but the 2-D cross-section of the computational volume (the computational window) is restricted to be a square of a given size. Therefore, it would be worthwhile to implement the automatic selection of another important parameter, namely the size or even the shape of the computational window in 2-D. This would be suitable for incorporation within the adaptive volume construction in the entropy-based minimization method. Such a method would be extremely powerful in segmenting small 3-D structures, where correct parameter estimation can be obtained only via the accumulation of more stationary data along the third dimension. This is particularly true in cases where a fixed-size stationary computational window within a 2-D slice is hard to find. Another step in this direction, would be to incorporate more prior knowledge on the shape of the 2-D cross-sections into the computational window shape to encourage particular shape evolutions. For example, such an approach would be useful in identifying blood vessels in MRI or other image modalities.

More work can be done on pre-processing, based either on the data or using analysis information. For example, the experiments with the MRI showed the sensitivity of the methods to the image modality. Also, previous work on 2-D window-based segmentation [77], suggests that a post-processing techniques such as the ICM algorithm can be used to enhance the segmentation result.

The entropy minimization method offers the flexibility of determining the optimal computational volume depth, but the current implementation is based on exhaustive search and entropy computations for all possible depths. An immediate improvement would be to develop an automatic stopping rule based on the entropy variations dependance on depth.

Supervised segmentation finds application in medical image analysis, for instance, where anatomical knowledge is a source for codebook generation. In many

real applications, however, an automatic unsupervised segmentation is the task to be solved. Methods based on 2-D computational windows have been extended to the unsupervised case. Therefore, a natural step forward would be the extension of the developed 3-D methods to serve the unsupervised segmentation demands.

The methods were shown to give impressive results for synthetic 3-D textures where the z-going correlations are the key discriminant between texture classes. These methods, however, are shown to be unnecessary for segmenting real images like T2 MRI, where simpler 2-D approaches give satisfactory results. It is important to investigate real-world applications where the abilities of the compensated methods are vital for success. One such application could be 3-D electron or confocal microscopy [61]. MRI datasets of high resolution would also probably be amenable to the 3-D techniques presented in this thesis, since they would manifest a sufficient number of textured voxels per tissue type.

The entropy minimization framework has been a step towards a parameter-free one-step segmentation method. A drawback is its computational cost. The local repetitive nature of the computations involving the GMRF neighbourhood, parameter estimation on the RUR grid and recomputing of the entropy of a growing matrix suggest a parallel implementation would be suitable. Some initial experiments [69] have shown that a High Performance Computing approach is a direction for future research.

Bibliography

[1] T. Aach and H. Dawid, "Region Oriented 3-D segmentation of NMR Datasets," in *SPIE Visual Communications and Image Processing*, pp. 691–701, 1990.

[2] F. Argenty, L. Alparone, and G. Benelli, "Fast Algorithms for Texture Analysis using Co-occurrence Matrices," in *IEE Proceedings, Part F: Radar and Signal Processing*, vol. 137(6), pp. 443–448, 1990.

[3] S. Barnard and W. Thompson, "Disparity Analysis of Images," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 2:4, pp. 333–340, 1980.

[4] J. Besag, "Spatial Interaction and the Statistical Analysis of Lattice Systems. (with discussions)," *Journal of the Royal Statistical Society, Series B*, vol. 36, pp. 192–236, 1974.

[5] J. Besag, "Statistical Analysis of Non-lattice Data.," *The Statistician*, vol. 24, pp. 179–195, 1975.

[6] J. Besag, "On the Statistical Analysis of Dirty Pictures," *Journal of the Royal Statistical Society, Series B*, vol. 48, pp. 259–302, 1986.

[7] A. Borcs and I. Pitas, "Object Segmentation in 3-D Images Based on Alpha-Trimmed Mean Radial Basis Function Network," in *Proceedings of the European Conference on Signal and Image Processing*, 1998.

[8] A. Bovik, M. Clarke, and W. Geisler, "Multichannel Texture Analysis using Localized Spatial Filters," *Pattern Analysis and Machine Intelligence*, vol. 12, pp. 55–73, 1990.

[9] M. Chang, M. Tekalp, and M.Sezan, "Simultaneous Motion Estimation and Segmentation," *IEEE Transactions on Image Processing*, vol. 6(9), pp. 1326–1333, 1997.

[10] M. Chang, M. Tekalp, and I. Sezan, "Bayesian Segmentation of MR Images using 3-D Gibssian Priors," in *1993 Image and Video Processing*, pp. 122–133, SPIE Conference, 1993.

[11] T. Chang and C. Kuo, "Texture Analysis and Classification with Tree-structured Wavelet Transform," *IEEE Transactions on Image Processing*, vol. 2, pp. 429–441, 1993.

[12] B. Chaudhuri, N. Sarkar, and P. Kundu, "Improved Fractal Geometry Based Texture Segmentation Technique," *IEE Proceedings, Part E*, vol. 140, pp. 233–241, 1993.

[13] R. Chellappa and S. Chatterjee, "Classification of Textures using Gaussian Markov Random Fields," *IEEE Transactions on Acoustic Speech and Signal Processing*, vol. 33(4), pp. 959–963, 1985.

[14] R. Chellappa, "Two-dimensional Discrete Gaussian Markov Random Field Models for Image Processing," *Progress in Pattern Recognition 2*, pp. 79–112, 1985.

[15] S. Choi and J. E. L. et al, "Volumetric Object Reconstruction using 3-D MRF Model-Based Segmentation," *IEEE Transactions on Medical Imaging*, vol. 16, no. 16, pp. 997–892, 1997.

[16] E. Clark and A. Quinn, "A Data-Driven Bayesian Sampling Scheme for Unsupervised Image Segmentation," in *International Conference on Acoustic, Speech and Signal Processing*, 1999.

[17] T. Clifford, *Markov Random Fields in Statistics.* Clarendon, Oxford, 1990.

[18] L. Cohen and I. Cohen, "Finite-element Methods for Active Contour Models and Balloons for 2-D and 3-D Images," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 15, no. 11, pp. 1131–1147, 1993.

[19] W. Cooley and J. Tukey, "An Algorithm for the Machine Calculation of Complex Fourier Series.," *Mathematics of computation*, vol. 19, pp. 297–301, 1965.

[20] T. Cover and J. Thomas, *Elements of Information Theory.* Wiley Interscience, New York, 1991.

[21] G. Cross and A. Jain, "Markov Random Field Texture Models," *Pattern Analysis and Machine Intelligence*, vol. 5, pp. 25–39, 1983.

[22] P. Davis, *Circulant Matrices.* John Wiley and sons, 1979.

[23] H. Derin and H. Elliot, "Modelling and Segmentation of Noisy and Textured Images using Gibbs Random Fields," *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, vol. PAMI-9, pp. 39–55, 1987.

[24] M. Do and M. Vetterli, "Texture Similarity Measurements using Kullback-Leibler Distance on Wavelet Subbands," in *Proceedings of the International Conference on Image Processing*, (Vancouver, Canada), 2000.

[25] E. Dougherty, E. Kraus, and J.Pelz, "Image Segmentation by Local Morphological Granulometries," in *Proceedings of IGARSS'89 and Canadian Symposium on Remote Sensing*, pp. 1220–1223, 1989.

[26] R. Dubes and A. Jain, "Random Field Models in Image Analysis," *Journal of Applied Statistics*, vol. 16(2), pp. 131–164, 1989.

[27] R. Duda and P. Hart, *Pattern Classification and Scene Analysis* . Wiley-Interscience, 1973.

[28] J. E. D. Fisher and S. Collins, "Automated detection of noninvasive magnetic resonance markers," in *Computers in Cardiology*, pp. 493–496, 1991.

[29] J. Fwu and P. Djurić, "EM Algorithm for Image Segmentation Initialized by a Tree Structure Scheme," *IEEE Transactions on Image Processing*, vol. 6(2), pp. 349–352, 1997.

[30] M. Galloway, "Texture Classification using Grey Level Run Length," *Computer Graphics and Image Processing*, vol. 4, pp. 172–179, 1975.

[31] S. Geman and D. Geman, "Stochastic Relaxation, Gibbs Distribution, and Bayesian Restoration of Images," *IEEE Transactions Pattern Analysis Machine Intelligence*, vol. 6, pp. 721–741, 1984.

[32] Gregson, "Automatic Segmentation of a Heart in 3-D MR images," in *Proceedings of the Canadian Conference on Electrical and Computer Engineering*, pp. 584–587, 1994.

[33] I. Grinias and G. Tziritas, "Motion Segmentation and Tracking using a Seeded Region Growing Method," in *Proceedings of the European Conference on Signal and Image Processing*, 1998.

[34] M. Haindl, "Texture synthesis," Technical Report CS-R9139, Amsterdam, The Netherlands, 1991.

[35] R. Haralick, K. Shanmugam, and I. Dinstein, "Textural Features for Image Classification," *Transactions on SMC*, vol. 3, no. 6, pp. 610–621, 1973.

[36] R. Haralick and L. Shapiro, "Survey: Image Segmentation Techniques," *Computer Vision, Graphics and Image processing*, vol. 29, pp. 100–132, 1985.

[37] K. Held and E. R. K. et al, "Markov Random Field Segmentation of Brain MR Images," *IEEE Transactions on Medical Imaging*, vol. 16, no. 6, pp. 878–886, 1997.

[38] B. Horn and B. Schunk, "Determining Optical Flow," *Artificial Intelligence*, vol. 17, pp. 185–204, 1981.

[39] A. Jain and R. Dubes, *Algorithms for Clustering Data.* Prentice Hall, 1988.

[40] A. Jain, *Fundamentals of Digital Image Processing.* Prentice Hall, 1989.

[41] R. Jain, "Extraction of Motion Information from Peripheral Processes," *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, vol. PAMI-3, N5, pp. 489–503, 1981.

[42] R. Jain, "Difference and Accumulative Difference Pictures in Dynamic Scene Analysis," *Image and Vision Computing*, vol. 2(2), pp. 99–108, 1984.

[43] R. Jain and H. Nagel, "On the Analysis of Accumulative Difference Pictures from Image Sequences of Real World Scenes," *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, vol. PAMI-1, N2, pp. 206–214, 1979.

[44] K. Jonhson, "Neuroimaging Primer (http:// www.med.harvard.edu/ AANLIB/ sigsors.html)." Harvard Medical School.

[45] M. Kaas, A. Witkin, and D. Terzopolous, "Snakes: Active Contour Models," *International Journal of Computer Vision*, vol. 1, pp. 321–331, 1987.

[46] R. Kashyap, "Random Field Models on Finite Lattices for Finite Images," in *Conference on Information Sciences and Systems, Johns Hopkins University*, pp. 215–220, 1981.

[47] P. Kjell and P. Wang, "Noise-tolerant Texture Classification and Image Segmentation," in *Intelligent Robots and Computer Vision*, pp. 553–560, 1991.

[48] A. Kokaram, *Motion Picture Restoration.* Springer Verlag, 1998.

[49] Lakshmanan and Derin, "Simultaneous Parameter Estimation and Segmentation of Gibbs Random Fields using Simulated Annealing .," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 11, pp. 799–813, 1989.

[50] S. Lankshmanan and H. Derin, "Valid Parameter Space for 2-D Gaussian Markov Random Fields," *IEEE Transactions on Information Theory*, vol. 39, no. 2, pp. 703–709, 1993.

[51] K. Laws, "Texture Energy Measures," in *Image Understanding Workshop*, pp. 47–51, 1979.

[52] K. Laws, *Textured Image Segmentation.* PhD thesis, University of Southern California, 1980.

[53] S. Z. Li, *Markov Random Field Modelling in Computer Vision.* Springer-Verlag, 1995.

[54] A. Maintz and M. Viergever, "A Survey of Medical Image Registration," *Medical Image Analysis*, vol. 2, pp. 1–36, 1998.

[55] S. Mallat, "A Theory of Multiresolution Signal Decomposition: The Wavelet Representation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 11, pp. 674–693, 1989.

[56] Mandelbrot, *The Fractal Geometry of Nature.* Freeman, 1982.

[57] B. Manjunath and R. Chellappa, "Unsupervised Texture Segmentation using Markov Random Field Models," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 13(5), pp. 478–482, 1991.

[58] D. Melas, *A Bayesian Approach to the Segmentation of Textural Images.* PhD thesis, Trinity College Dublin, 1998.

[59] D. E. Melas and S. P. Wilson, "Double Markov Random Fields and Bayesian Image Segmentation," *IEEE Transactions on Signal Processing*, vol. 50, no. 2, pp. 357–365, 2002.

[60] R. D. Morris, *Image Sequence Restoration using Gibbs Distributions.* PhD thesis, Cambridge University, England, 1995.

[61] N. Nikolaidis and I. Pitas, *3-D Image Processing Algorithms.* John Wiley and Sons Inc., 2001.

[62] S. Ourselin, A. Roche, G. Subsol, X. Pennec, and C. Sattonnet, "Automatic Alignment of Histological Sections for 3-D Reconstruction and Analysis ," research report, 1998.

[63] T. Pavlidis, *Structural Pattern Recognition.* Spinier Verlag Berlin Heidelberg New York, 1977.

[64] A. Pentland, "Fractal-based Description of Natural Scene," *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, vol. PAMI-6, pp. 661–674, 1984.

[65] M. Petrou and P. Bosdogioanni, *Image Processing: the Fundamentals.* John Wiley and sons, 1999.

[66] R. W. Picard, *Texture Modelling: Temperature Effects on Markov/Gibbs Random Fields.* PhD thesis, Massachusetts Institute of Technology, 1991.

[67] I. Pitas, *Digital Image Processing Algorithms.* Prentice Hall, 1993.

[68] J. Puzicha, T. Hofmann, and J. Buhmann, "Unsupervised Texture Segmentation on the Basis of Scale Space Features.".

[69] E. Ranguelova and A. Quinn, "Analysis and Synthesis of Textured Images on 3-D Lattices," Technical Report 00/EETCD/SP/02, Trinity College Dublin (TCD), Ireland, 2000.

[70] E. Ranguelova and A. Quinn, "Analysis and Synthesis of Three-Dimensional Gaussian Markov Random Field," in *Proceedings of the International Conference on Image Processing*, (Kobe, Japan), 1999.

[71] E. Ranguelova and A. Quinn, "Disparity-compensated Segmentation of 3-D Images," in *PhD workshop Cybernetics and Informatics Eurodays*, (Marianska, Czech Republic), 2000.

[72] E. Ranguelova and A. Quinn, "Registration Preprocessing for Enhanced 3-D Segmentation," in *Proceedings of the Irish Signals and Systems Conference*, (Maynooth, Ireland), pp. 346–351, June 2001.

[73] E. Ranguelova and A. Quinn, "Difference Field Estimation for Enhanced 3-D Texture Segmentation.," in *Proceedings of the British Machine Vision Conference*, (Cardiff, UK), pp. 373–383, BMVA, Sept. 2002.

[74] E. Ranguelova and A. Quinn, "Difference Estimation and Compensation via Entropy Minimization in 3-D Image Segmentation," in *Proceedings of the Second International Workshop on Spectral Methods and Multirate Signal Processing*, (Toulouse, France), pp. 131–136, Sept. 2002.

[75] T. Reed, H. Wechsler, and M. Werman, "Texture Segmentation using a Diffusion Region Growing Technique," *Pattern Recognition*, vol. 23, no. 9, pp. 953–960, 1990.

[76] J. Reichel, "Development of Fast Unsupervised Texture Segmentation Techniques for Computer-Aided Diagnosis," erasmus report, Trinity College Dublin (TCD), 1997.

[77] J. Reichel and A. Quinn, "A Fast and Fully Unsupervised Scheme for Model–Based Image Segmentation," in *Bayesian Inference for Inverse Problems*, SPIE Conference, 1998.

[78] D. Rey, G. Subsol, H. Delingette, and N. Ayache, "Automatic Detection and Segmentation of Evolving Processes in 3-D Medical Images: Application to Multiple Sclerosis," research report, 1998.

[79] F. Site, "MIT temporal textures database (ftp:// whitechapel.media.mit.edu/ pub/ szummer/ temporal-tecture/)."

[80] W. Site, "The Whole Brain Atlas (http:// www.med.harvard.edu/ AANLIB/ home.html)."

[81] M. Sonka, V. Hlavač, and R. Boyle, *Image Processing, Analysis and Machine Vision.* Chapman and Hall Computing, 1998.

[82] C. Stiller and J. Konrad, "Estimating Motion in Image Sequences," *IEEE Signal Processing Magazine*, pp. 71–91, July 1999.

[83] M. Szummer and R. W. Picard, "Temporal Texture Modelling," in *Proceedings of the International Conference on Image Processing*, vol. 3, pp. 823–826, 1996.

[84] M. Tanner, *Tools for Statistical Inference* . Springer-Verlag, 1996.

[85] M. Tekalp, *Digital Video Processing.* Prentice Hall Signal Processing Series, 1995.

[86] D. Terzopolous, M. Kaas, and A. Witkin, "Constraints on Deformable Models: Recovering 3-D Shape and Non-rigid Motion," *Artificial Intelligence Journal*, vol. 36, pp. 91–123, 1998.

[87] S. Ullman, *The Interpretation of Visual Motion.* The MIT Press, 1979.

[88] P. Vautrot, N. Bonnet, and M. Herbin, "Comparative study of Different Spatial/Spatial Frequency Methods for texture Segmentation/Classification," in *Proceedings of IEEE International Conference on Image Processing*, pp. 145–148, 1996.

[89] B. Vemuri, S. Huang, S. Sahni, C. Leonard, C. Mohr, R. Gilmore, and J. Fitzsimons, "An Efficient Motion Estimator with Application to Medical Image Registration," *Medical Image Analysis*, vol. 2, pp. 79–98, 1998.

[90] D. Vernon, *Machine Vision: Automated Visual Inspection and Robot Vision.* Prentice Hall, 1991.

[91] P. Willemin, T. Reed, and M. Kunt, "Image Sequence Coding by Split and Merge," *IEEE Transactions on Communications*, vol. 39(12), pp. 1945–1855, 1791.

[92] J. Woods, "Two-Dimensional Discrete Markovian Random Fields," *IEEE Transactions on Information Theory*, vol. 18, pp. 232–240, 1972.

[93] C. M. Wu, Y. Chen, and K. Hsieh, "Texture Features for Classification of Ultrasonic Liver Images," *IEEE Transactions of Medical Imaging*, vol. 11, pp. 141–152, 1992.

[94] G. Wu, *3-D Segmentation-based Image Sequence Processing.* PhD thesis, University of California, 1997.

[95] J. Yla-jaaski and O. Kubler, "Segmentation and Analysis of 3-D Volume Images," in *Proceedings of the 9th International Conference on Pattern Recognition*, pp. 951–953, 1988.

[96] W. Zhao and T. Young, "Registration and Three-Dimensional Reconstruction of Autoradiographic Images by the Disparity Analysis Method," *IEEE Transactions on Medical Imaging*, vol. 12, pp. 782–791, 1993.

Appendix A

DIAGONALIZATION OF CIRCULANTS OF LEVEL THREE

Here is the proof of Theorem 4.1 from Section 4.3.

In [22] it is proven that any block matrix $\mathbf{M}$ with n circulant blocks can be represented by

$$\mathbf{M} = \sum_{j=1}^{n} (\boldsymbol{\pi}_n^{j-1} \otimes \mathsf{M}_j), \tag{A.1}$$

where each M_j is a circulant of level 1 of order m, $\boldsymbol{\pi}_n$ is the permutation matrix from (4.2) of order n and $\otimes$ denotes the Kronecker product.

Let $\mathbf{C}$ be a level 3 circulant of class $\mathcal{C}(m, n, p)$. Hence, from (A.1) it follows:

$$\mathbf{C} = \sum_{k=1}^{p} \boldsymbol{\pi}_p^{k-1} \otimes \mathbf{C}_k, \tag{A.2}$$

where the blocks $\mathbf{C}_k$ are block-circulants of level 2 of class $\mathcal{C}(m, n)$.

Hence, (A.1) can be applied again to yield:

$$\mathbf{C}_k = \sum_{j=1}^{n} \boldsymbol{\pi}_n^{j-1} \otimes \mathsf{C}_{j,k}, \tag{A.3}$$

where each $\mathsf{C}_{j,k}$ is a circulant of level 1 and from (4.3) it follows:

$$\mathsf{C}_{j,k} = \sum_{i=1}^{m} \mathsf{c}_i \boldsymbol{\pi}_m^{i-1}. \tag{A.4}$$

The entry c_i is at location (i, j, k) in the matrix $\mathbf{C}$, i.e. is the entry $\mathbf{C}(i, j, k)$. Substituting (A.4) into (A.3) and then into (A.2) gives

$$\mathbf{C} = \sum_{k=1}^{p} \sum_{j=1}^{n} \sum_{i=1}^{m} \mathbf{C}(i, j, k) \boldsymbol{\pi}_m^{i-1} \boldsymbol{\pi}_n^{j-1} \boldsymbol{\pi}_p^{k-1}. \tag{A.5}$$

Since $\boldsymbol{\pi}_u^v = \mathbf{F}_u^* \boldsymbol{\Omega}_u^v \mathbf{F}_u$ [22], where $\mathbf{F}_u$ is the Fourier matrix (4.6) and Ω_u is the matrix from (4.7) of order u, it follows that:

$$\begin{aligned}
\mathbf{C} &= \sum_{k=1}^{p} \sum_{j=1}^{n} \sum_{i=1}^{m} \mathbf{C}(i, j, k) (\mathbf{F}_m^* \boldsymbol{\Omega}_m^{i-1} \mathbf{F}_m \otimes \mathbf{F}_n^* \boldsymbol{\Omega}_n^{j-1} \mathbf{F}_n \otimes \mathbf{F}_p^* \boldsymbol{\Omega}_p^{k-1} \mathbf{F}_p) \\
&= \sum_{i=1,j=1,k=1}^{m,n,p} (\mathbf{F}_m \otimes \mathbf{F}_n \otimes \mathbf{F}_p)^* \mathbf{C}(i, j, k) (\boldsymbol{\Omega}_m^{i-1} \otimes \boldsymbol{\Omega}_n^{j-1} \otimes \boldsymbol{\Omega}_p^{k-1}) (\mathbf{F}_m \otimes \mathbf{F}_n \otimes \mathbf{F}_p).
\end{aligned}$$

Hence

$$\mathbf{C} = \boldsymbol{\mathcal{F}}^H \boldsymbol{\Lambda} \boldsymbol{\mathcal{F}}, \tag{A.6}$$

where $\boldsymbol{\mathcal{F}}$ s defined by (4.12) and $\boldsymbol{\Lambda} = \left[\sum_{i,j,k=1}^{m,n,p} \mathbf{C}(i, j, k) (\boldsymbol{\Omega}_m^{i-1} \otimes \boldsymbol{\Omega}_n^{j-1} \otimes \boldsymbol{\Omega}_p^{k-1}) \right]$.

Appendix B

LEAST SQUARES ESTIMATION OF GMRF PARAMETERS

This is the derivation of the result in Section 5.2. Starting from (5.4):

$$\mathbf{e}_t = \mathbf{x}_t - \mathbf{Y}_t\boldsymbol{\theta}_t. \tag{B.1}$$

Hence, the sum of the squared errors becomes:

$$\begin{aligned} \mathbf{e}_t^T\mathbf{e}_t &= (\mathbf{x}_t - \mathbf{Y}_t\boldsymbol{\theta}_t)^T(\mathbf{x}_t - \mathbf{Y}_t\boldsymbol{\theta}_t) \\ &= \mathbf{x}_t^T\mathbf{x}_t - \mathbf{x}_t^T\mathbf{Y}_t\boldsymbol{\theta}_t - \boldsymbol{\theta}_t^T\mathbf{Y}_t^T\mathbf{x}_t + \boldsymbol{\theta}_t^T\mathbf{Y}_t^T\mathbf{Y}_t\boldsymbol{\theta}_t. \end{aligned} \tag{B.2}$$

(B.2) is quadratic with respect to $\boldsymbol{\theta}_t$, so there is a unique optimum of the function. Furthermore, since $\mathbf{Y}_t^T\mathbf{Y}_t$ is in Cholesky form, then

$$\frac{\partial^2\mathbf{e}_t^T\mathbf{e}_t}{\partial\boldsymbol{\theta}_t} = \mathbf{Y}_t^T\mathbf{Y}_t > 0. \tag{B.3}$$

Hence, the optimizer locals the unique minimum. Re-writing (B.2):

$$\mathbf{e}_t^T\mathbf{e}_t = (\boldsymbol{\theta}_t - \boldsymbol{\alpha}_t)^T\mathbf{Y}_t^T\mathbf{Y}_t(\boldsymbol{\theta}_t - \boldsymbol{\alpha}_t) + \beta_t, \tag{B.4}$$

where

$$\boldsymbol{\alpha}_t = (\mathbf{Y}_t^T\mathbf{Y}_t)^{-1}\mathbf{Y}_t^T\mathbf{x}_t, \tag{B.5}$$

$$\beta_t = \mathbf{x}_t^T\mathbf{x}_t - \mathbf{x}_t^T\mathbf{Y}_t(\mathbf{Y}_t^T\mathbf{Y}_t)^{-1}\mathbf{Y}_t^T\mathbf{x}_t. \tag{B.6}$$

From (B.4) it is clear that the minimum is reached when $\widehat{\boldsymbol{\theta}}_t = \boldsymbol{\alpha}_t$ i.e.

$$\widehat{\boldsymbol{\theta}}_t = (\mathbf{Y}_t^T \mathbf{Y}_t)^{-1} \mathbf{Y}_t^T \mathbf{x}_t. \tag{B.7}$$

Using the definition of the matrix $\mathbf{Y}_t = [\mathbf{y}_r^T], r \in \omega_t$:

$$\widehat{\boldsymbol{\theta}}_t = \left[\sum_{r \in \omega_t} \mathbf{y}_r \mathbf{y}_r^T \right]^{-1} \left[\sum_{r \in \omega_t} \mathbf{y}_r \bar{x}_r \right] \tag{B.8}$$

where $\bar{x}_r = x_r - \widehat{\mu}_t$ is the zero-mean realization.

$$\widehat{\mu}_t = \frac{1}{n_T} \sum_{r \in \omega_t} x_r \tag{B.9}$$

is the arithmetic mean in ω_t. Finally $\widehat{\sigma_t^2} = \frac{1}{n_T} \mathbf{e}_t^T \mathbf{e}_t \mid_{\widehat{\theta}_t}$:

$$\begin{aligned} \widehat{\sigma_t^2} &= \frac{1}{n_T} \left(\mathbf{x}_t - \mathbf{Y}_t \widehat{\boldsymbol{\theta}}_t \right)^T \left(\mathbf{x}_t - \mathbf{Y}_t \widehat{\boldsymbol{\theta}}_t \right) \\ &= \frac{1}{n_T} \sum_{r \in \omega_t} \left[x_r - \mathbf{y}_r^T \widehat{\boldsymbol{\theta}}_t \right]^2 \end{aligned} \tag{B.10}$$

Appendix C

TEST DATA

C.1 3-D SYNTHETIC STATIONARY GMRFs

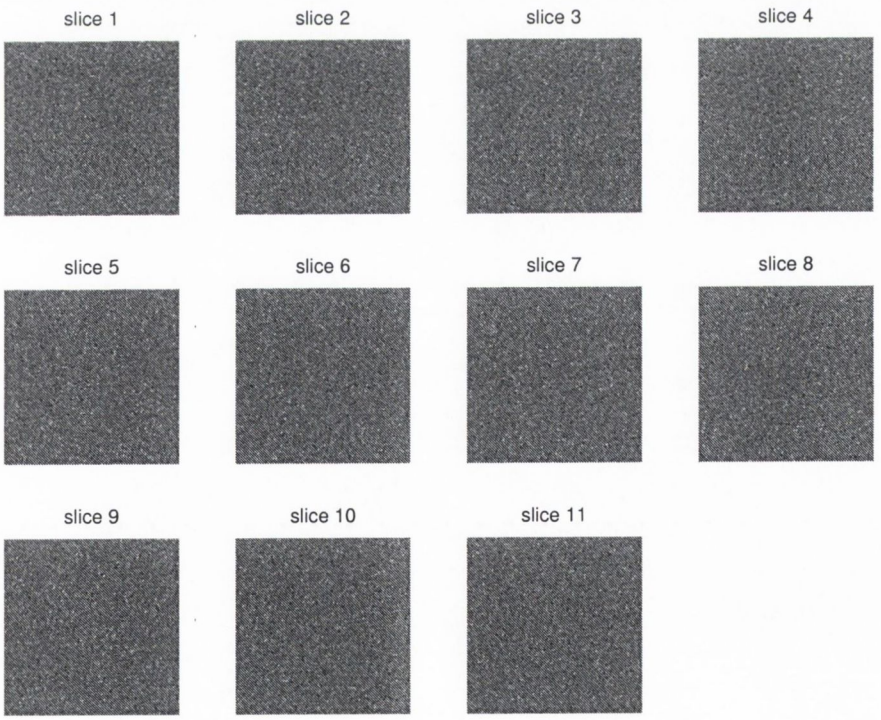

Figure C.1: 3-D grey level data: second order stationary GMRF, model 1 (Table 5.2).

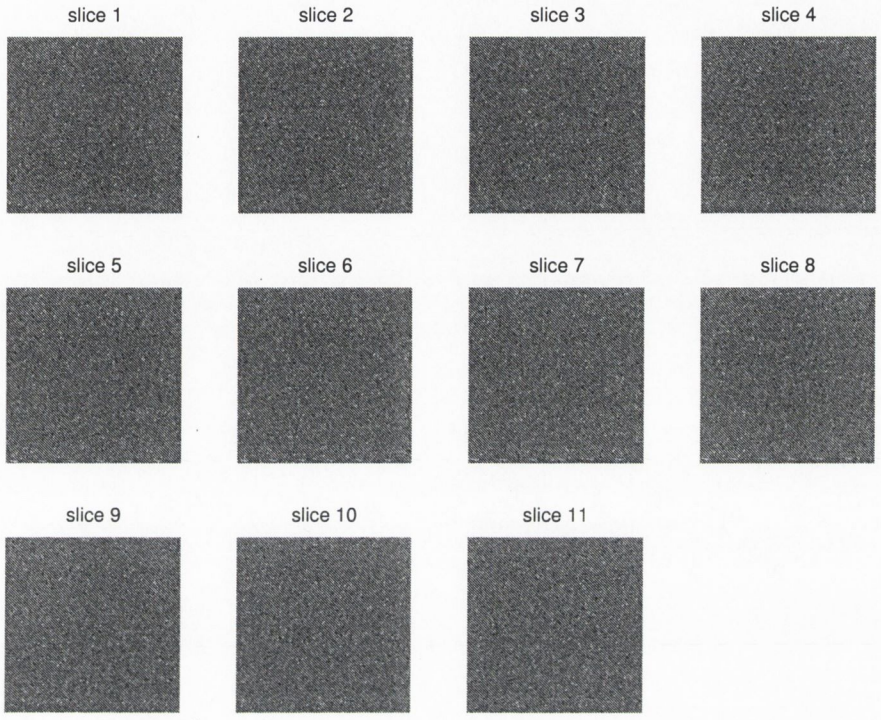

Figure C.2: 3-D grey level data: second order stationary GMRF, model 2 (Table 5.2).

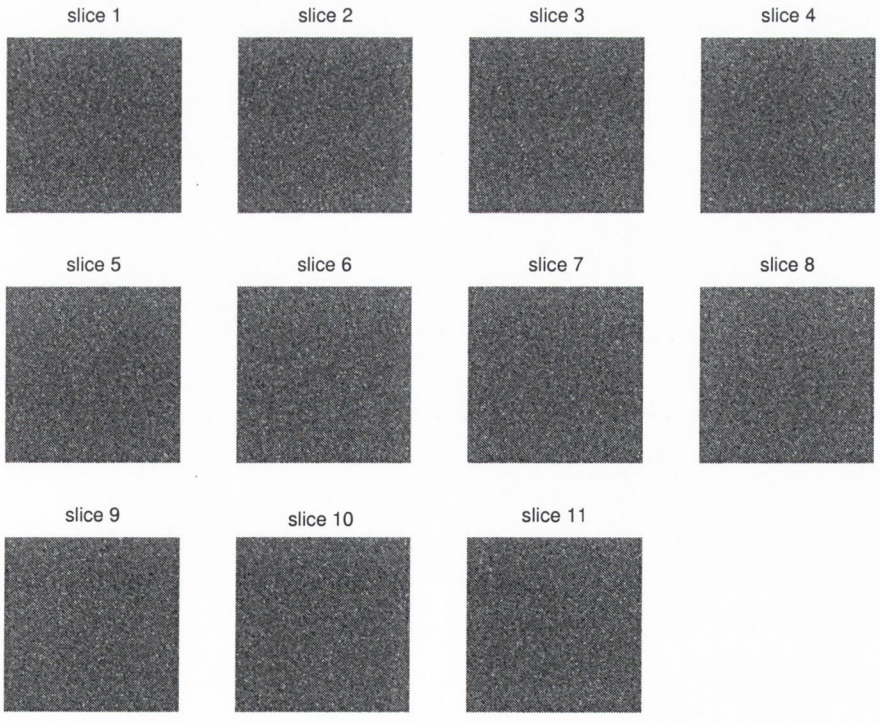

Figure C.3: 3-D grey level data: second order stationary GMRF, model 3 (Table 5.2).

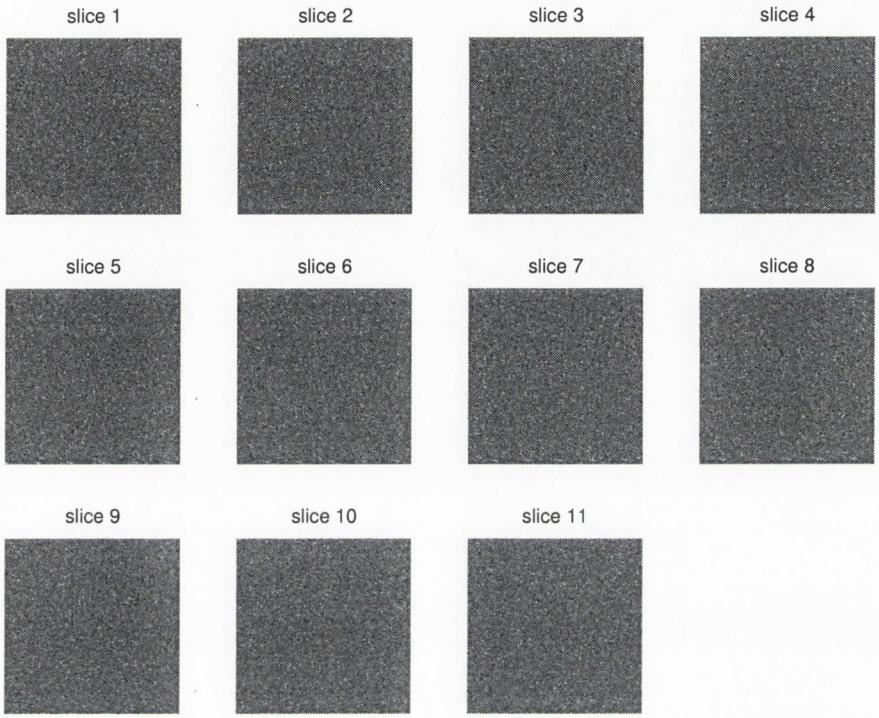

Figure C.4: 3-D grey level data: third order stationary GMRF, model 1 (Table 5.3).

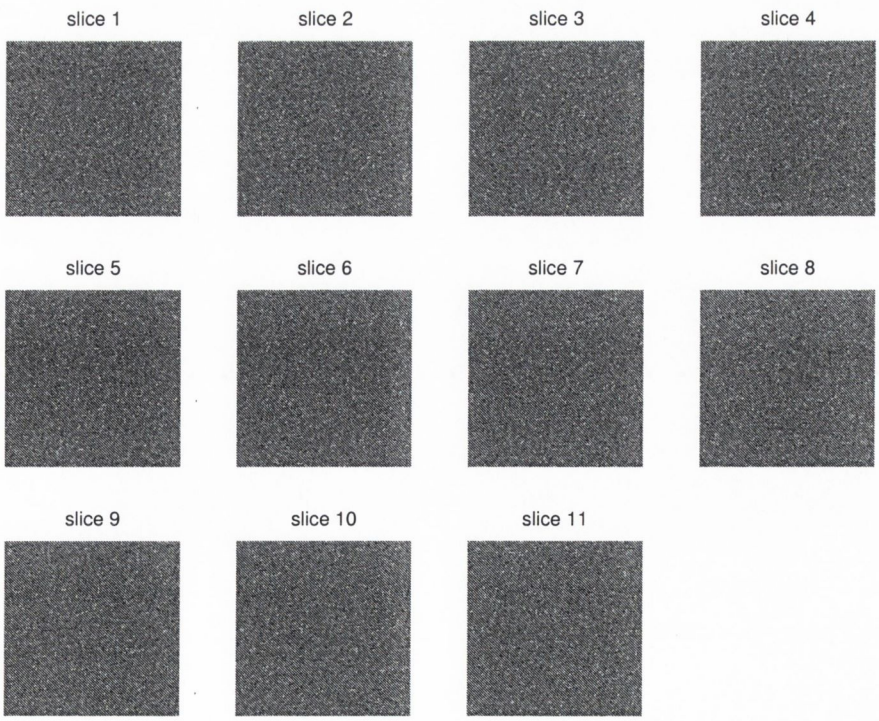

Figure C.5: 3-D grey level data: third order stationary GMRF, model 2 (Table 5.3).

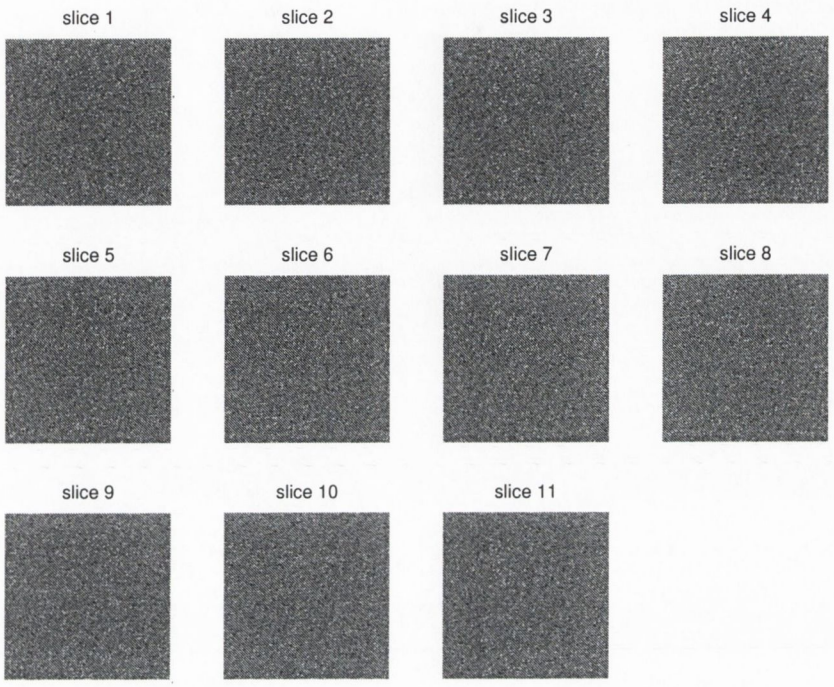

Figure C.6: 3-D grey level data: third order stationary GMRF, model 3 (Table 5.3).

C.2 3-D SYNTHETIC GMRF MIXTURES

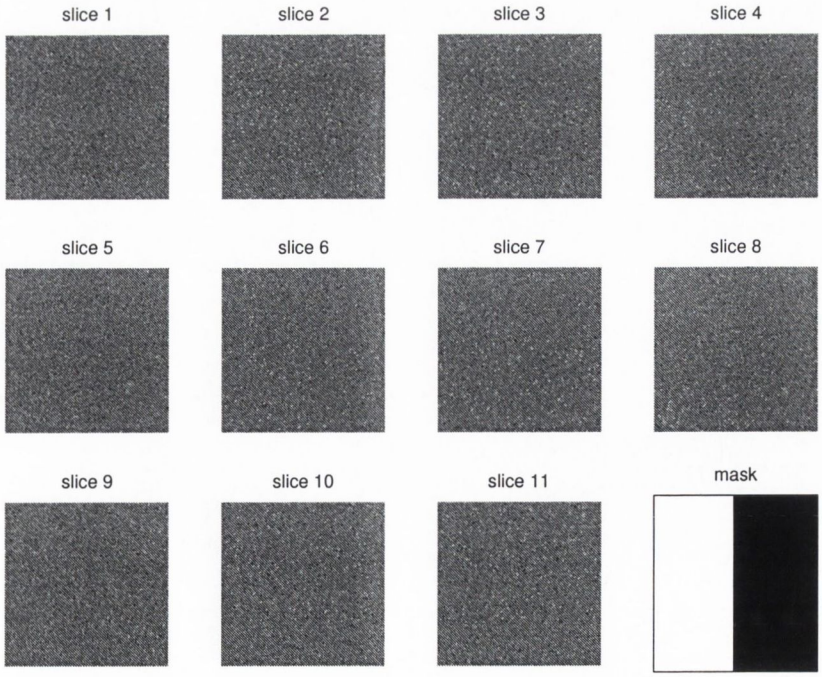

Figure C.7: 3-D grey level data: non-stationary volume, obtained by the stationary fields from Figs. C.1 and C.2 and the ground truth mask.

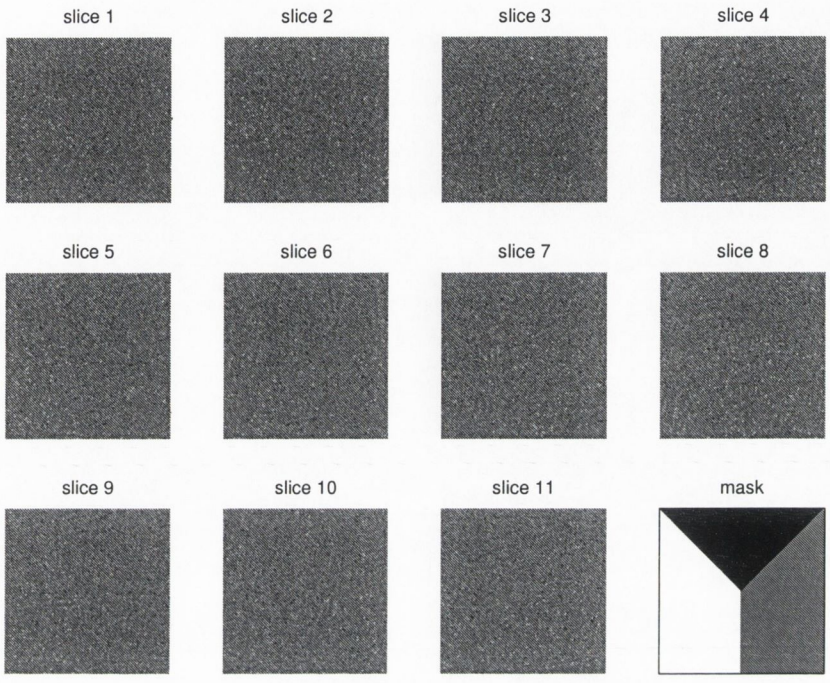

Figure C.8: 3-D grey level data: non-stationary GMRF volume, obtained by the stationary fields from Figs. C.4, C.5 and C.6 and the ground truth mask.

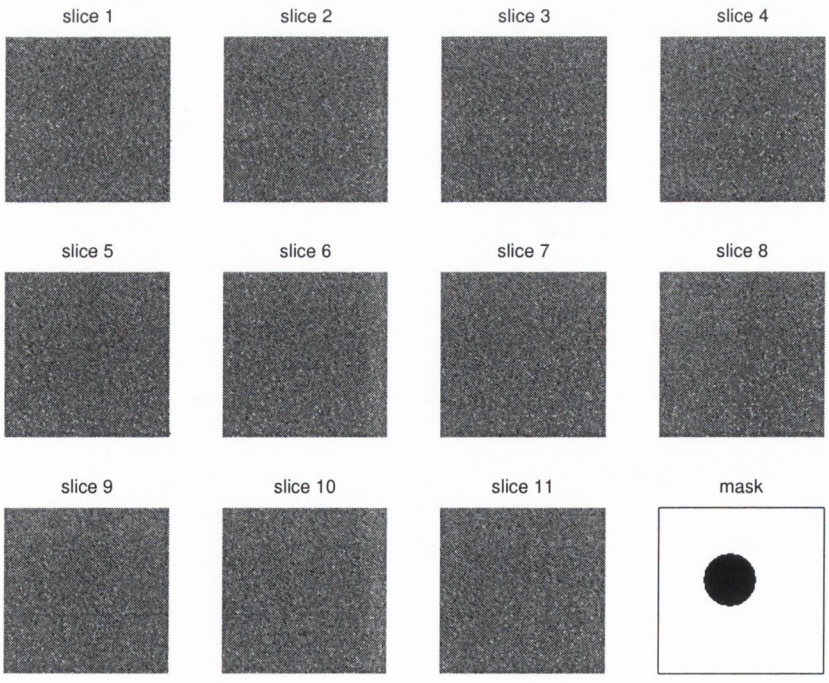

Figure C.9: 3-D grey level data: 'straight' cylinder volume, obtained by the stationary fields from Figs. C.1 and C.2 and the ground truth mask.

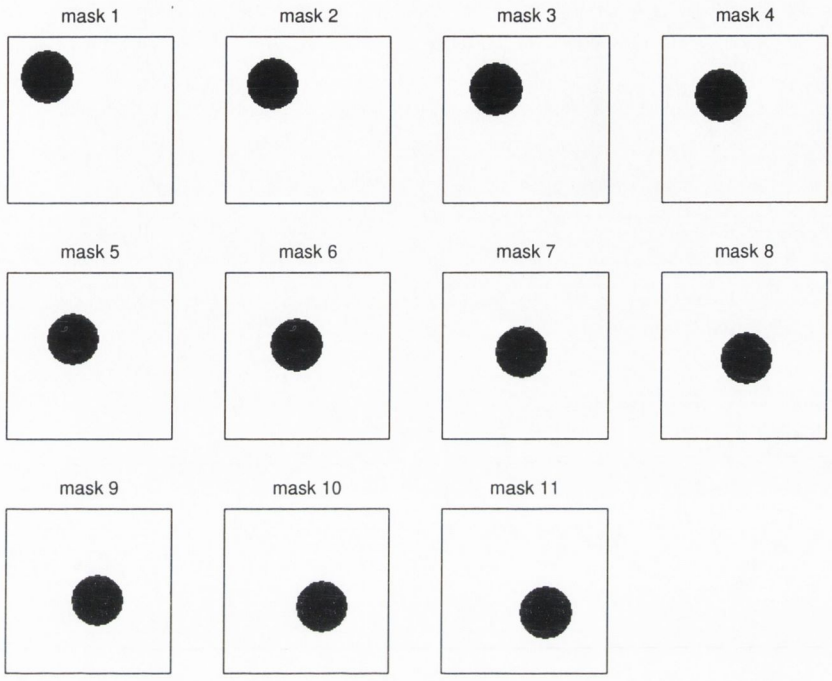

Figure C.10: Ground truth masks for the 'skewed' cylinder volume.

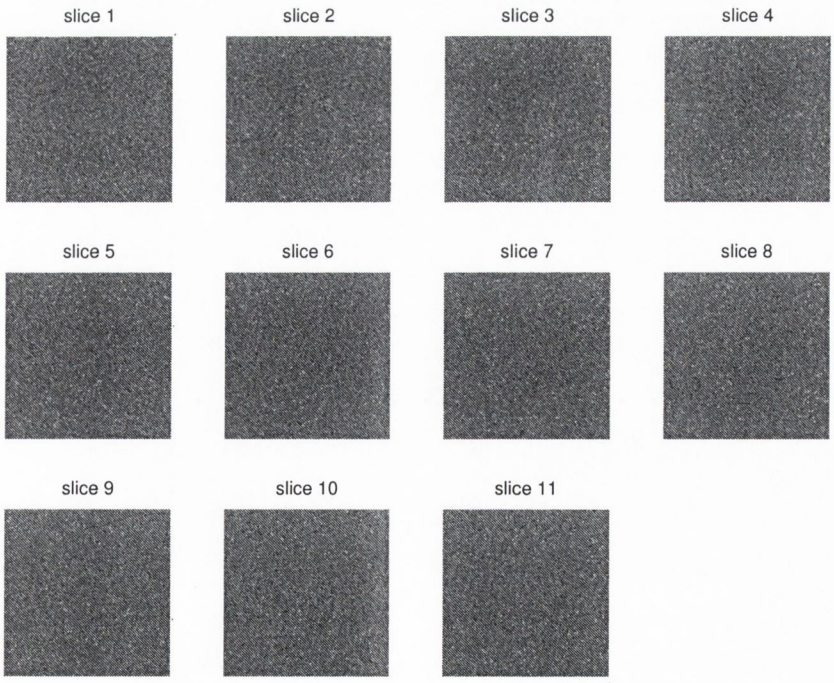

Figure C.11: 3-D grey level data: 'skewed' cylinder volume, obtained by the stationary fields from Figs. C.1 and C.2 and the ground truth masks from Fig. C.10.

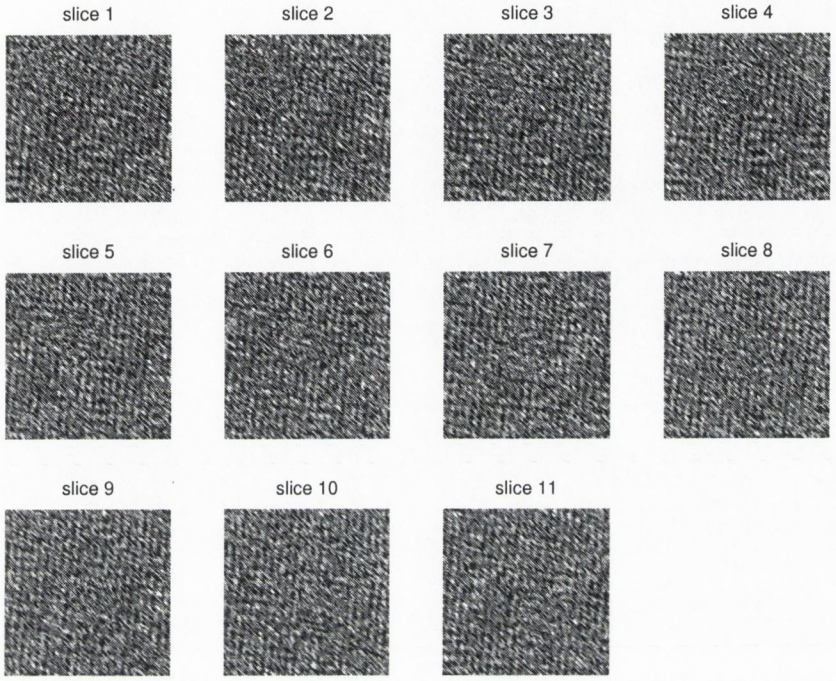

Figure C.12: 3-D grey level data: 'skewed' cylinder volume.

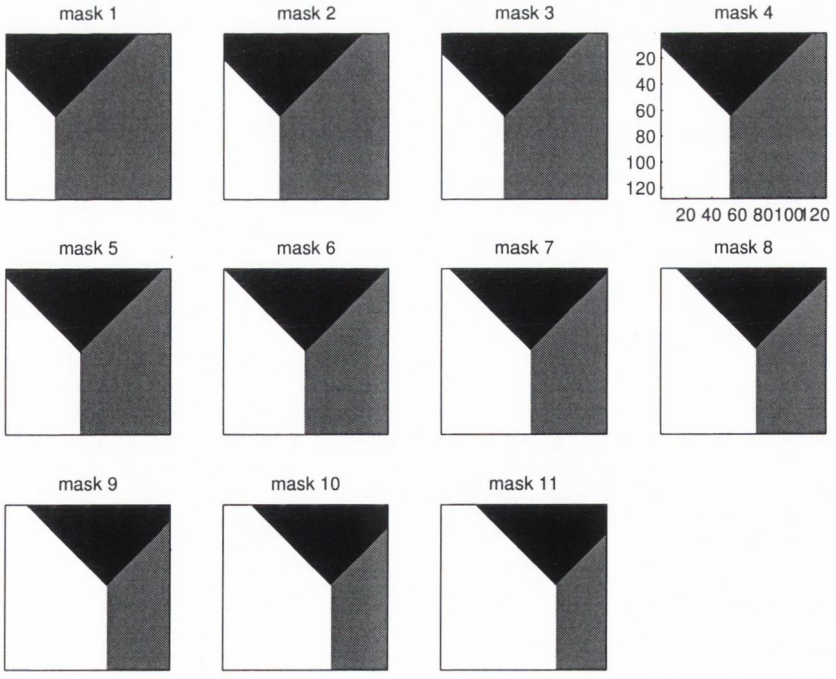

Figure C.13: Ground truth masks for the three textures volume.

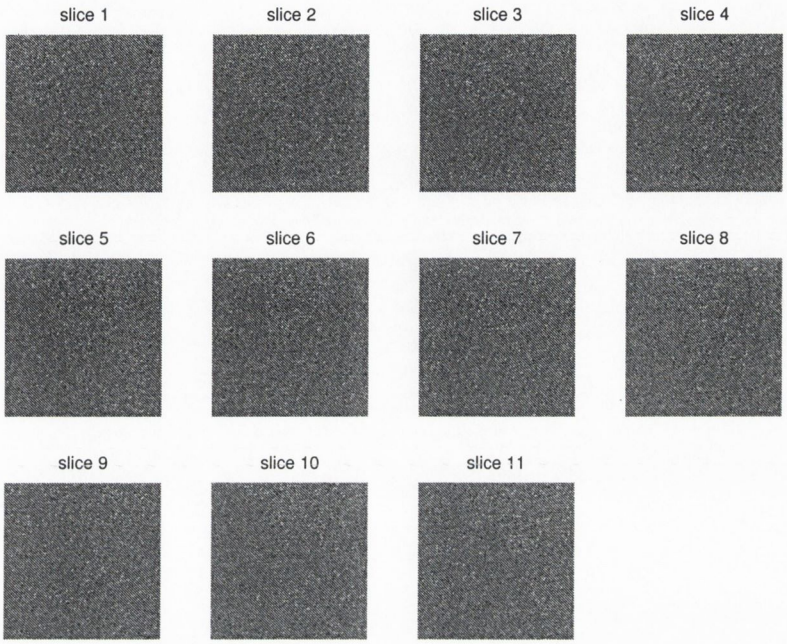

Figure C.14: 3-D grey level data: three textures volume, obtained by the stationary fields from Figs. C.4, C.5 and C.6 and the ground truth masks from Fig. C.13.

C.3 TEMPORAL TEXTURES

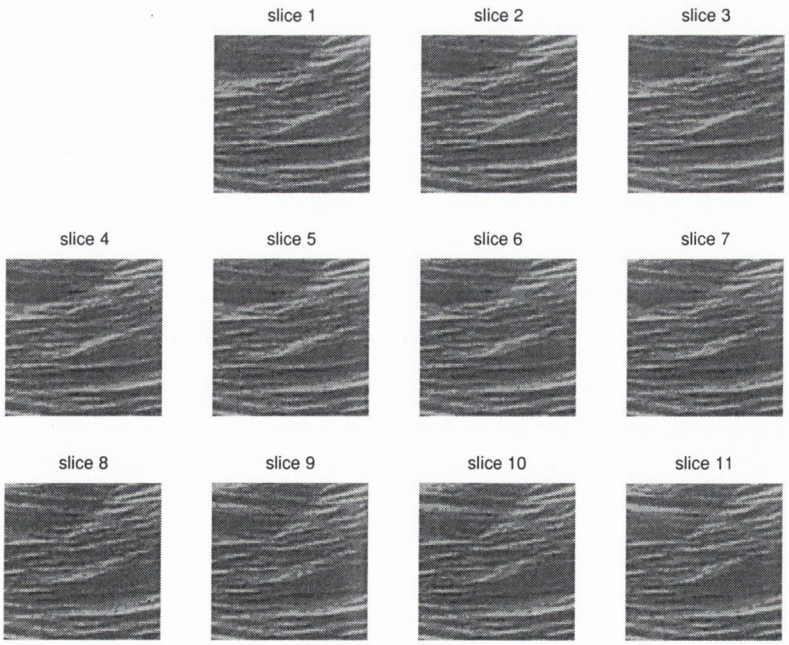

Figure C.15: 3-D grey level data: the 'plastic' temporal texture sequence.

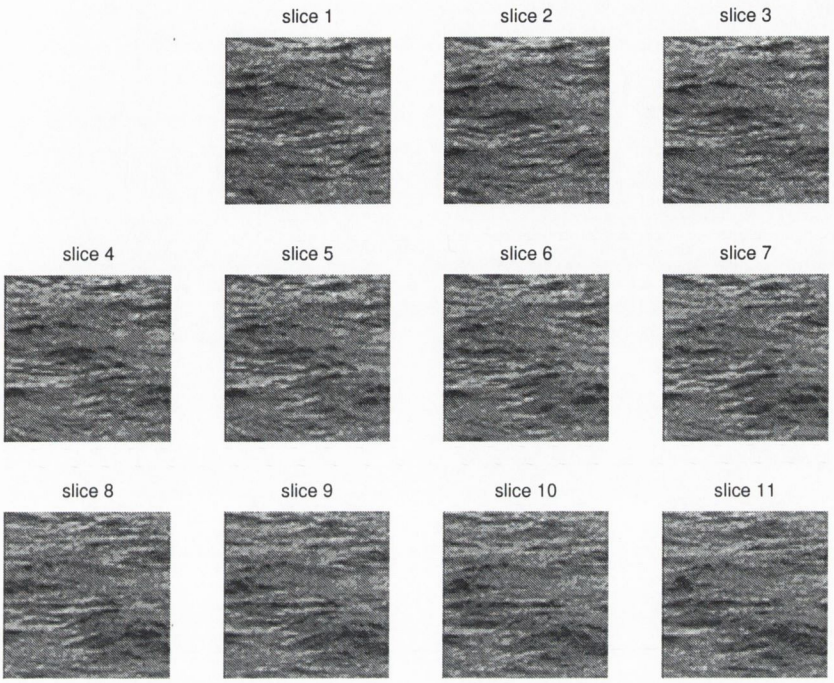

Figure C.16: 3-D grey level data: the 'river' temporal texture sequence.

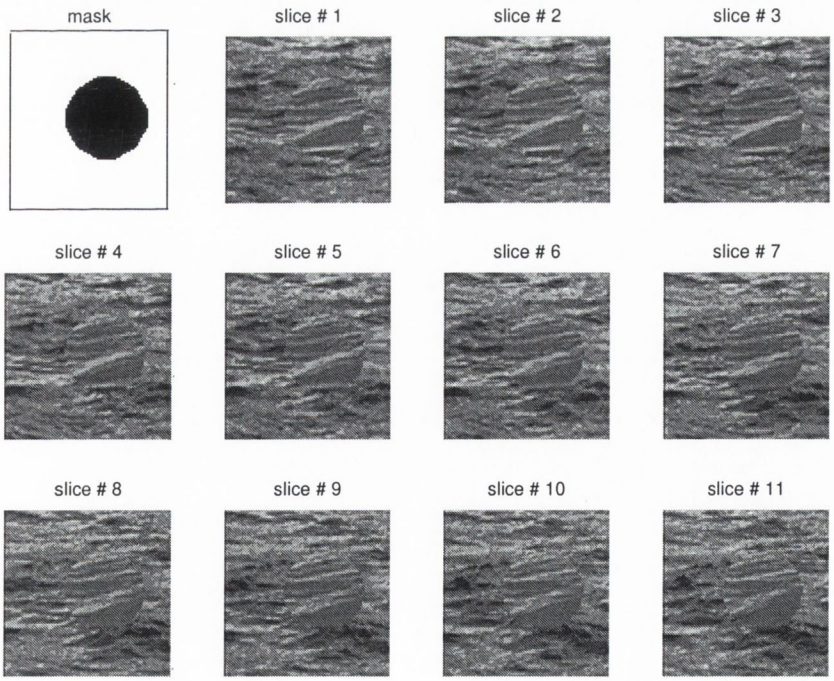

Figure C.17: Temporal textures mosaic using the ground truth mask and the data from Figs. C.15 and C.16.

C.4 MRI VOLUMES

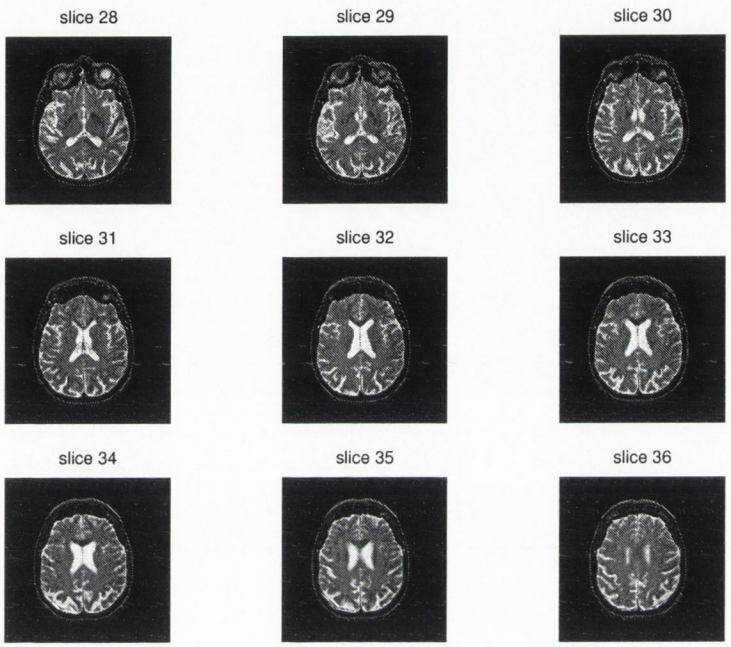

Figure C.18: Original grey level data: transverse T2-weighted slices of human brain.

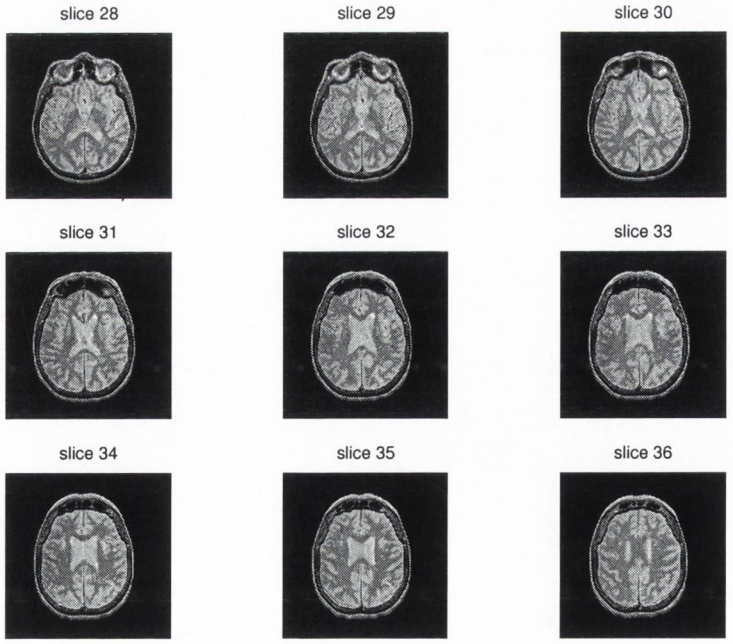

Figure C.19: Original grey level data: transverse PD-weighted slices of human brain.

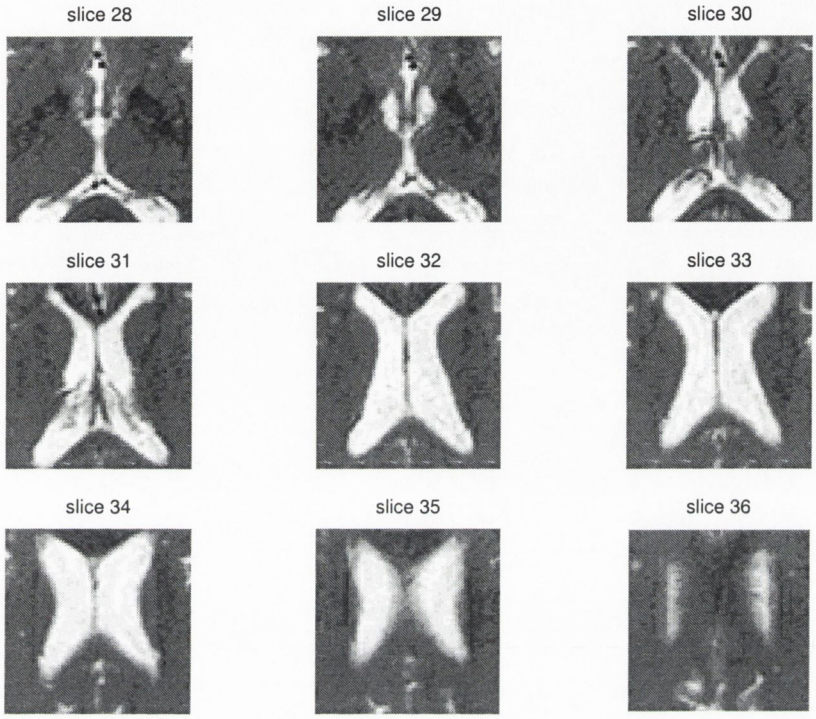

Figure C.20: Gray level data: region from transverse T2-weighted slices of human brain.

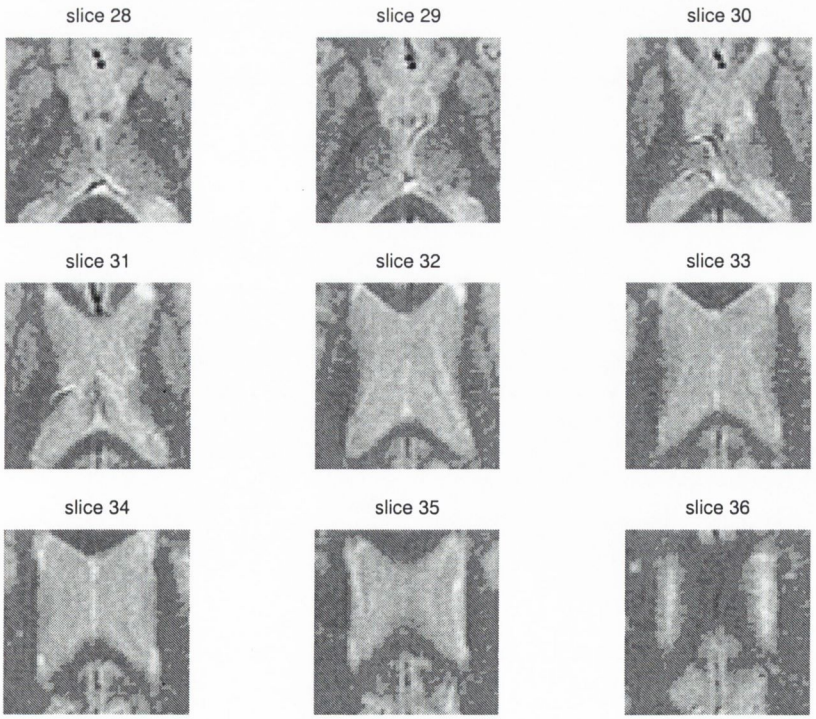

Figure C.21: Gray level data: region from transverse PD-weighted slices of human brain.

Appendix D

NON-COMPENSATED SEGMENTATION RESULTS

D.1 VOXEL-BY-VOXEL

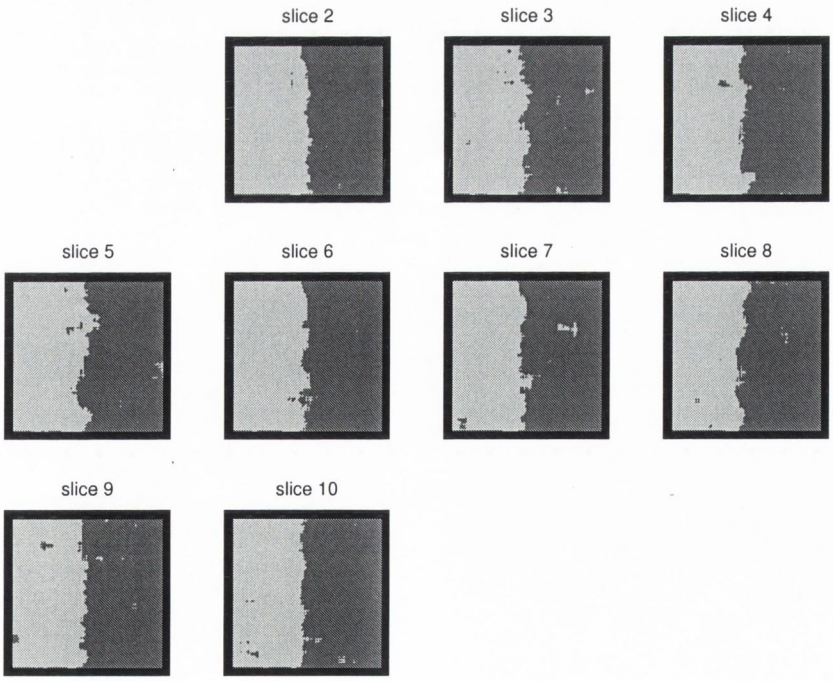

Figure D.1: 3-D VBV segmentation results for the volume from Fig. C.7, $d = 3$.

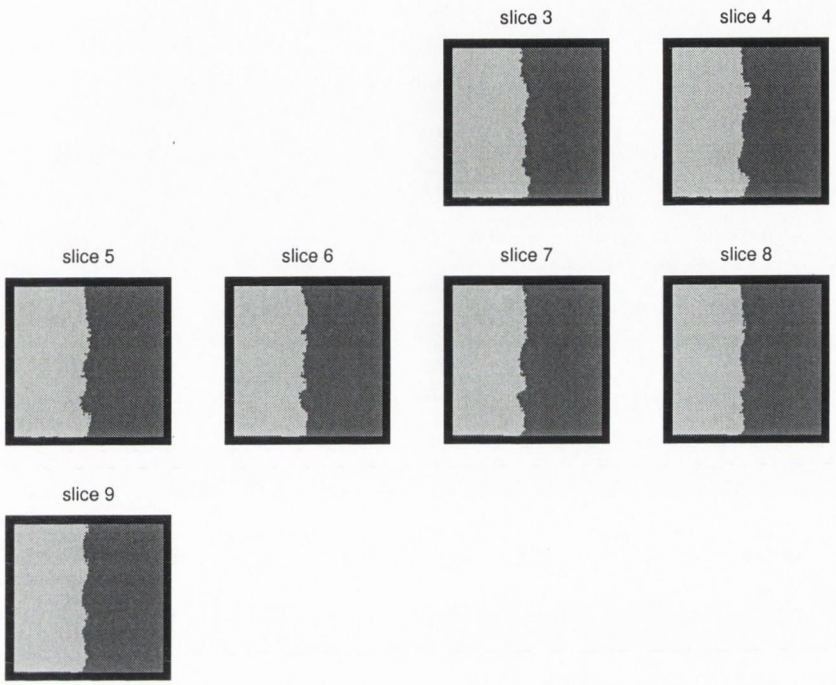

Figure D.2: 3-D VBV segmentation results for the volume from Fig. C.7, $d = 5$.

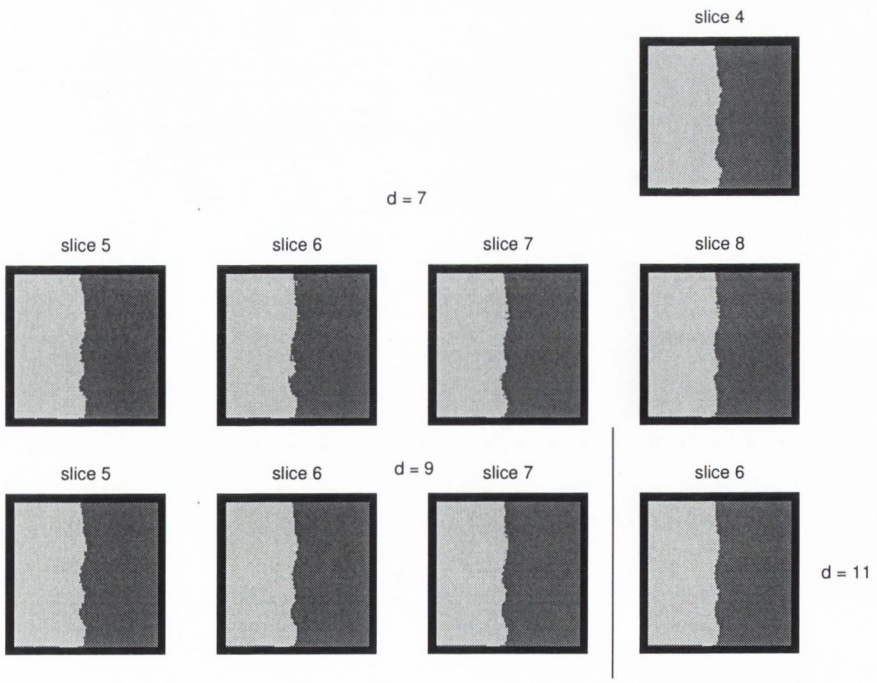

Figure D.3: 3-D VBV segmentation results for the volume from Fig. C.7, $d = 7, 9, 11$.

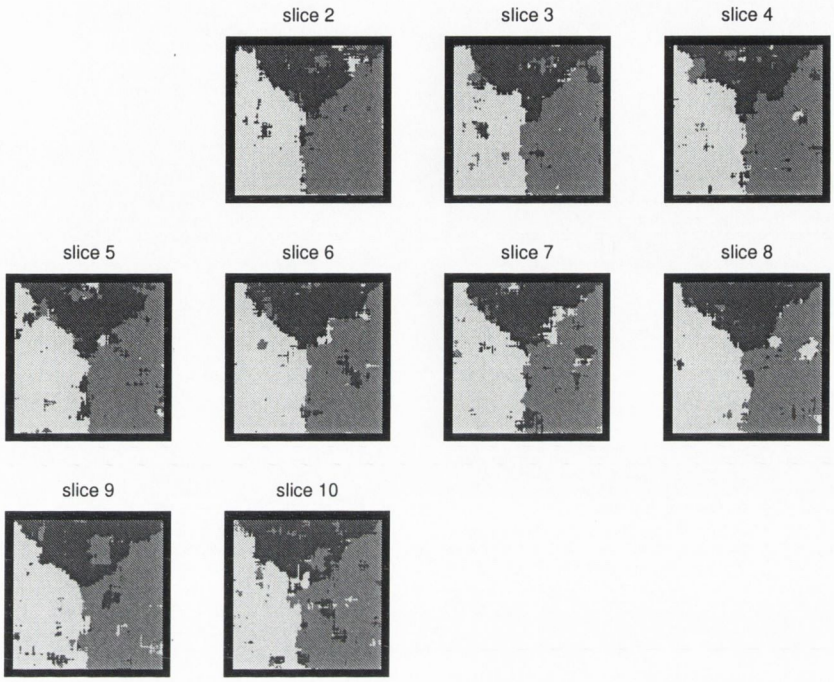

Figure D.4: 3-D VBV segmentation results for the volume from Fig. C.8, $d = 3$.

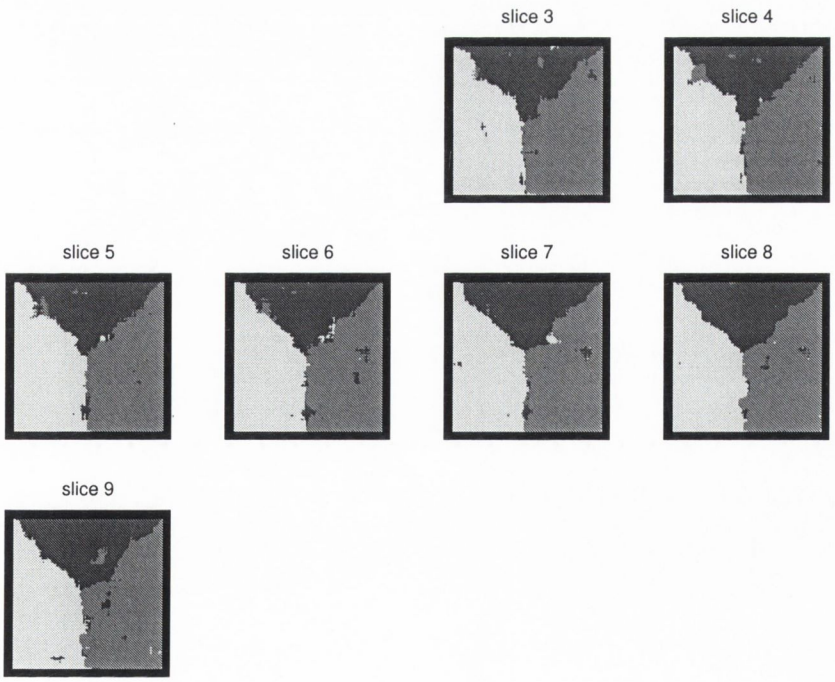

Figure D.5: 3-D VBV segmentation results for the volume from Fig. C.8, $d = 5$.

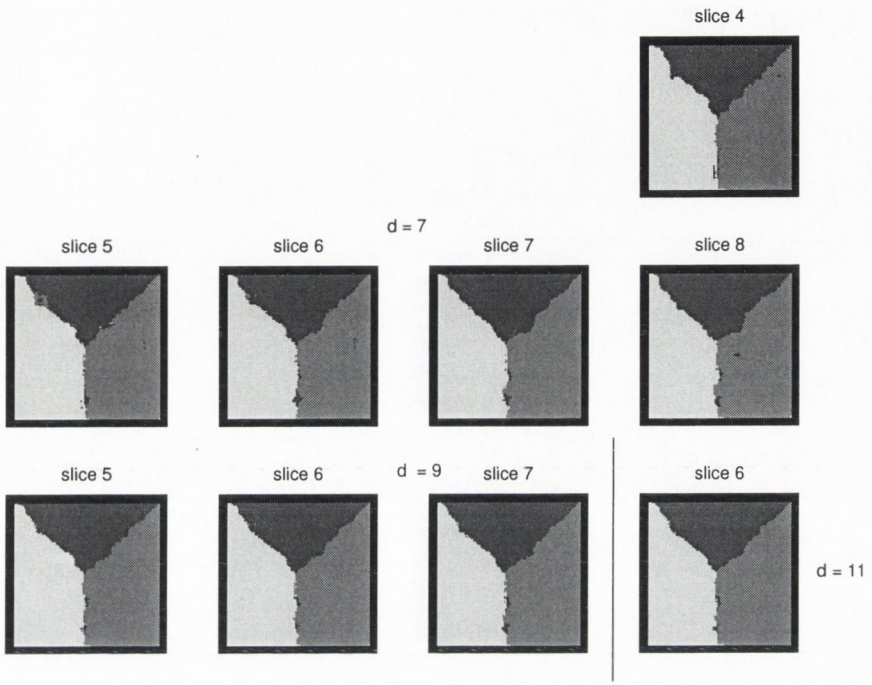

Figure D.6: 3-D VBV segmentation results for the volume from Fig. C.8, $d = 7, 9, 11$.

D.2 MAJORITY DECISION

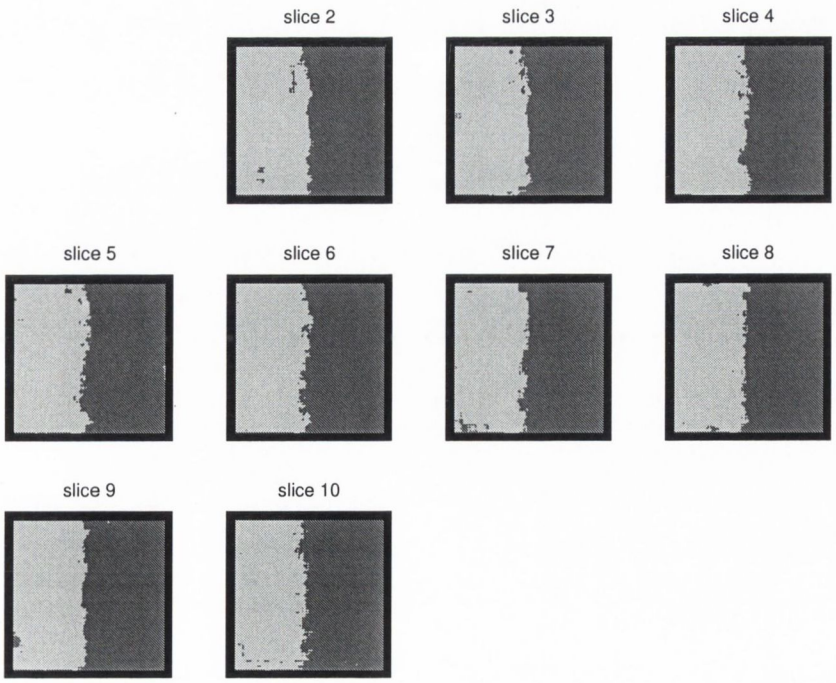

Figure D.7: 3-D MD segmentation results for the volume from Fig. C.8, $d = 3$.

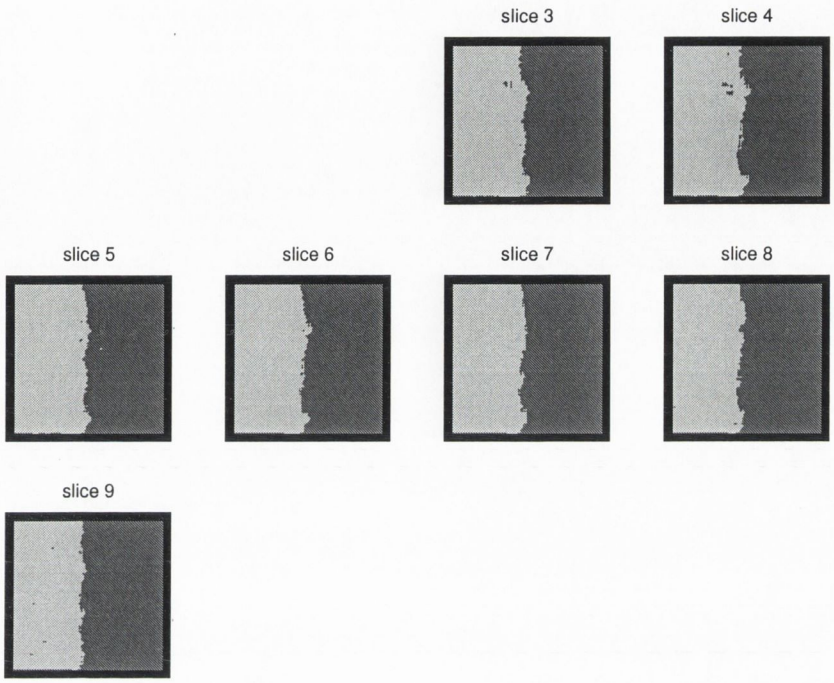

Figure D.8: 3-D MD segmentation results for the volume from Fig. C.8, $d = 5$.

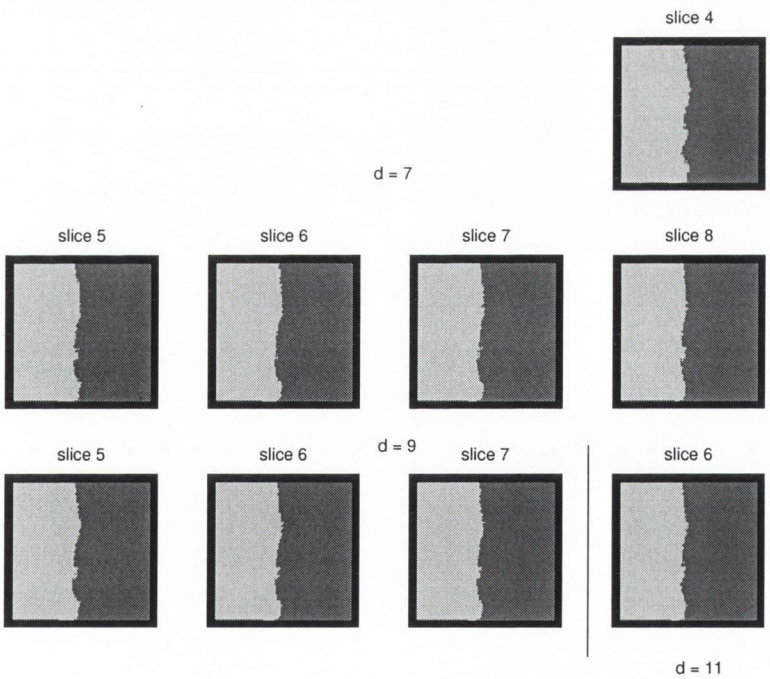

Figure D.9: 3-D MD segmentation results for the volume from Fig. C.8, $d = 7, 9, 11$.

D.3 UNANIMITY RULE

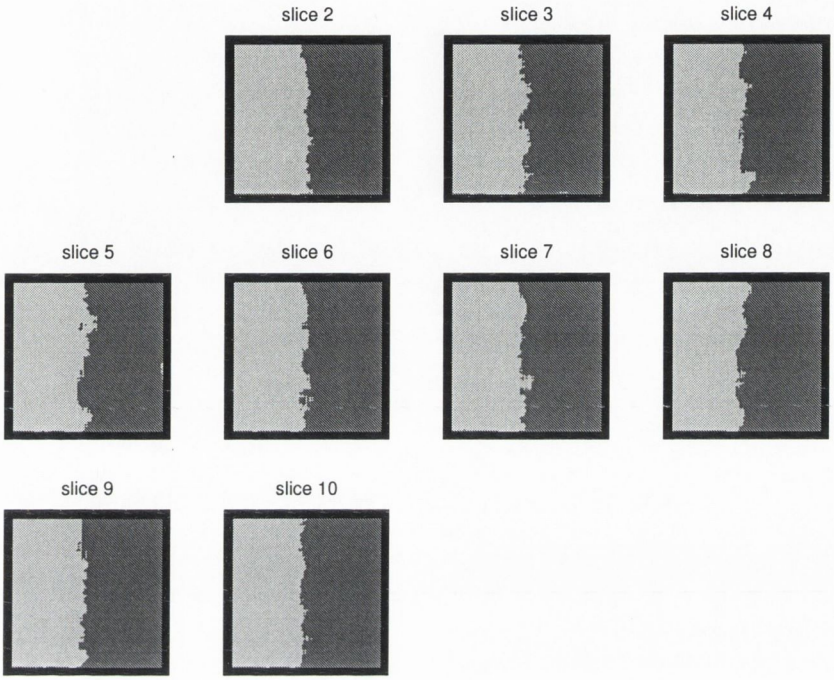

Figure D.10: 3-D UR segmentation results for the volume from Fig. C.7, $d = 3$.

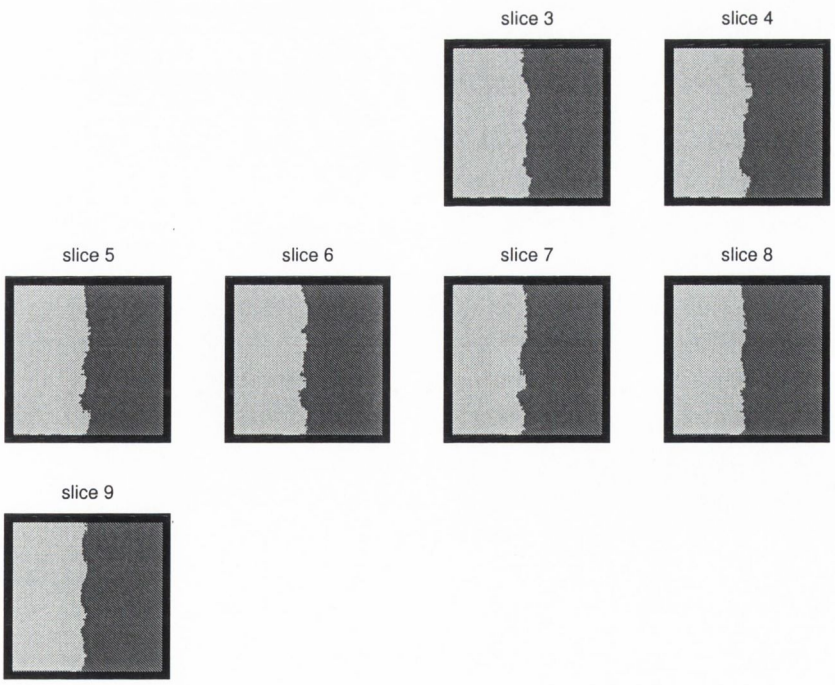

Figure D.11: 3-D UR segmentation results for the volume from Fig. C.7, $d = 5$.

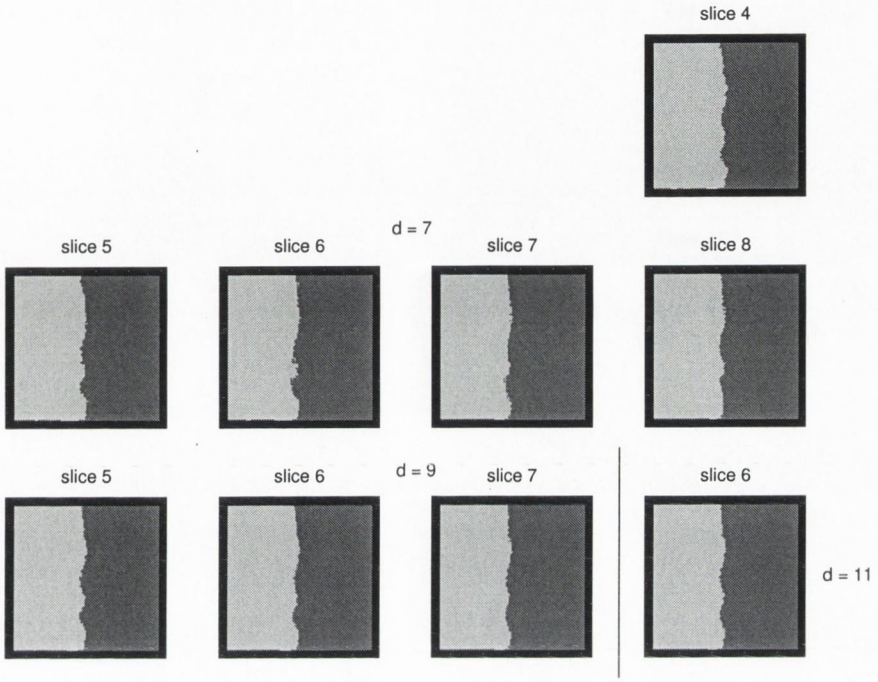

Figure D.12: 3-D UR segmentation results for the volume from Fig. C.7, $d = 7, 9, 11$.

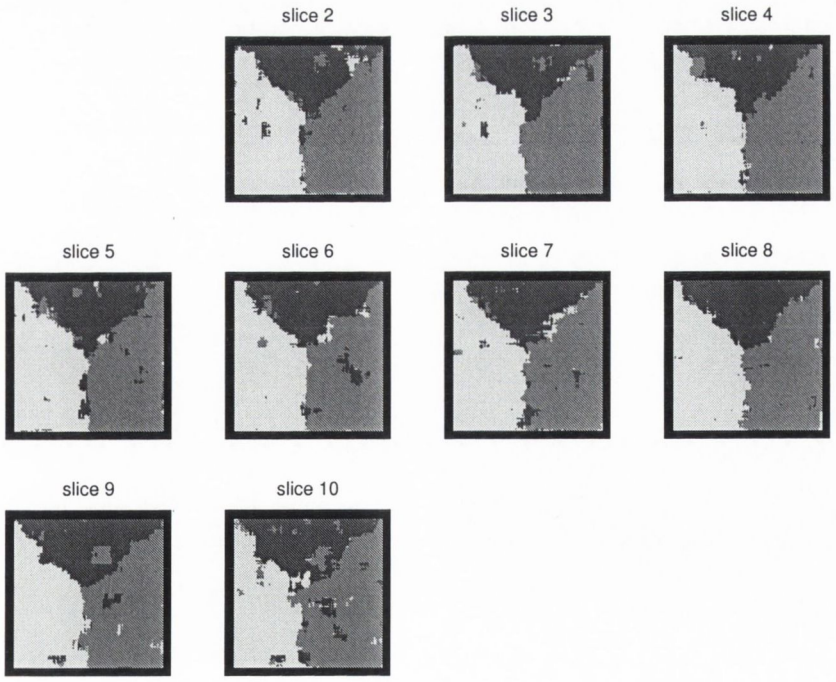

Figure D.13: 3-D UR segmentation results for the volume from Fig. C.8, $d = 3$.

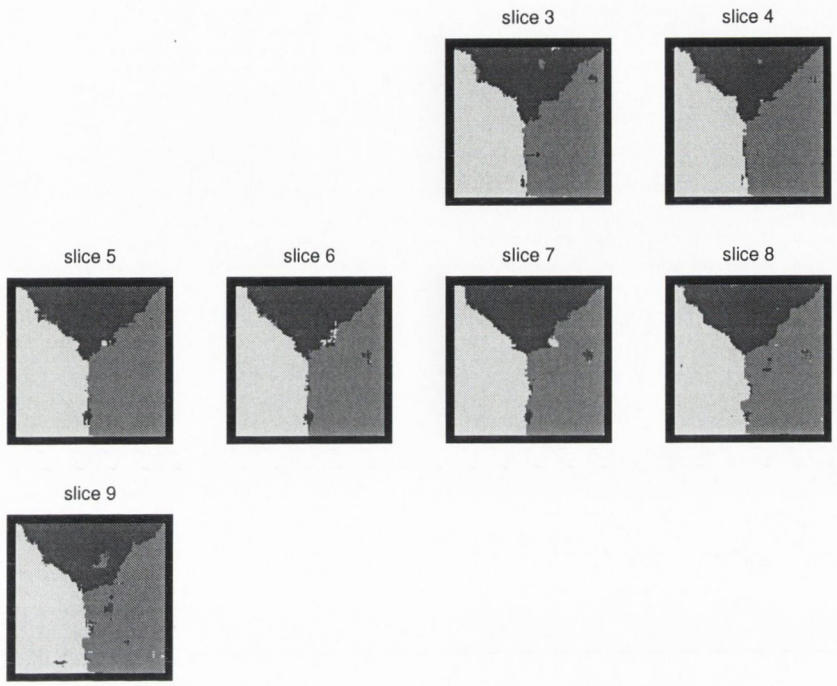

Figure D.14: 3-D UR segmentation results for the volume from Fig. C.8, $d = 5$.

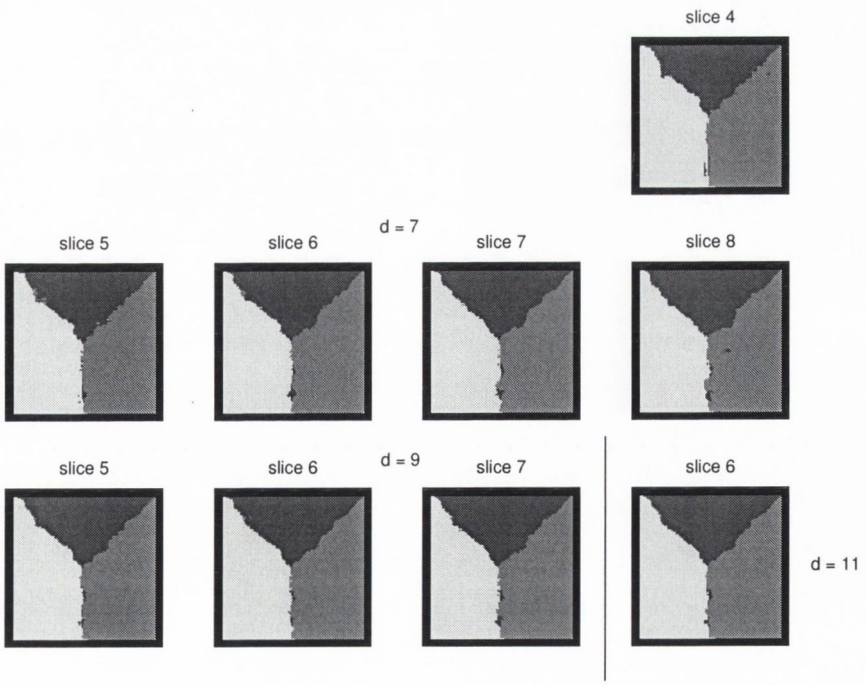

Figure D.15: 3-D UR segmentation results for the volume from Fig. C.8, $d = 7, 9, 11$.

D.4 RECURSIVE UNANIMITY RULE

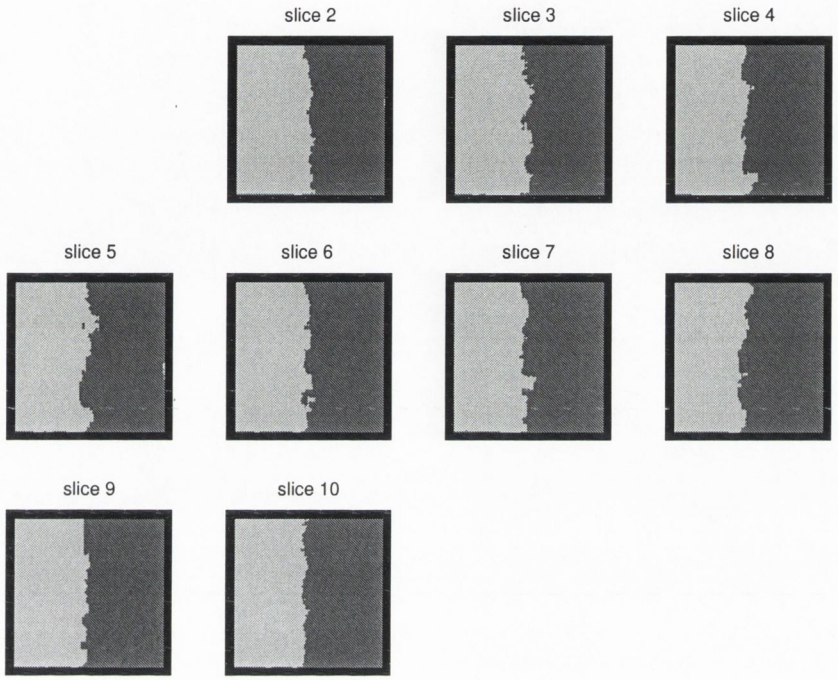

Figure D.16: 3-D RUR segmentation results for the volume from Fig. C.7, $d = 3$.

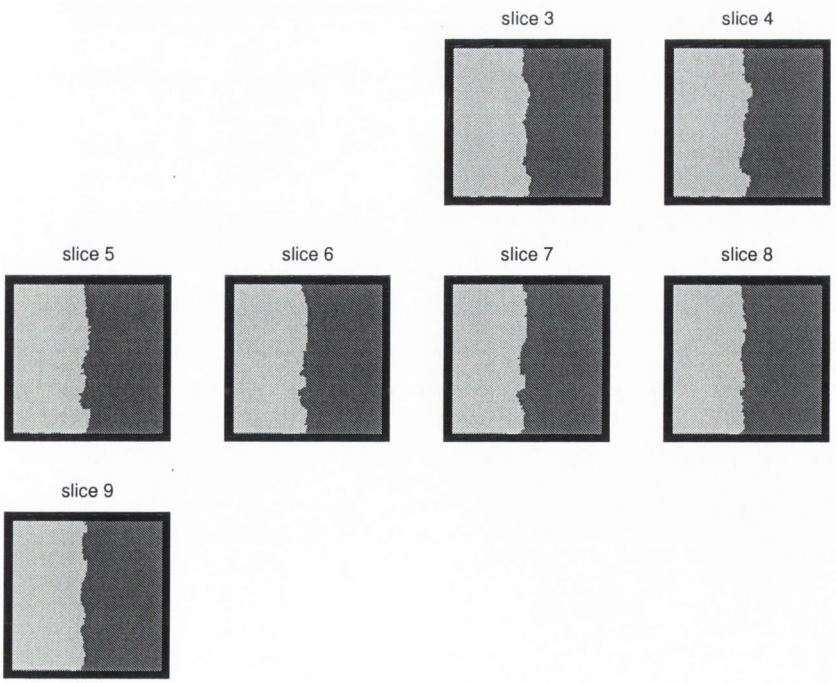

Figure D.17: 3-D RUR segmentation results for the volume from Fig. C.7, $d = 5$.

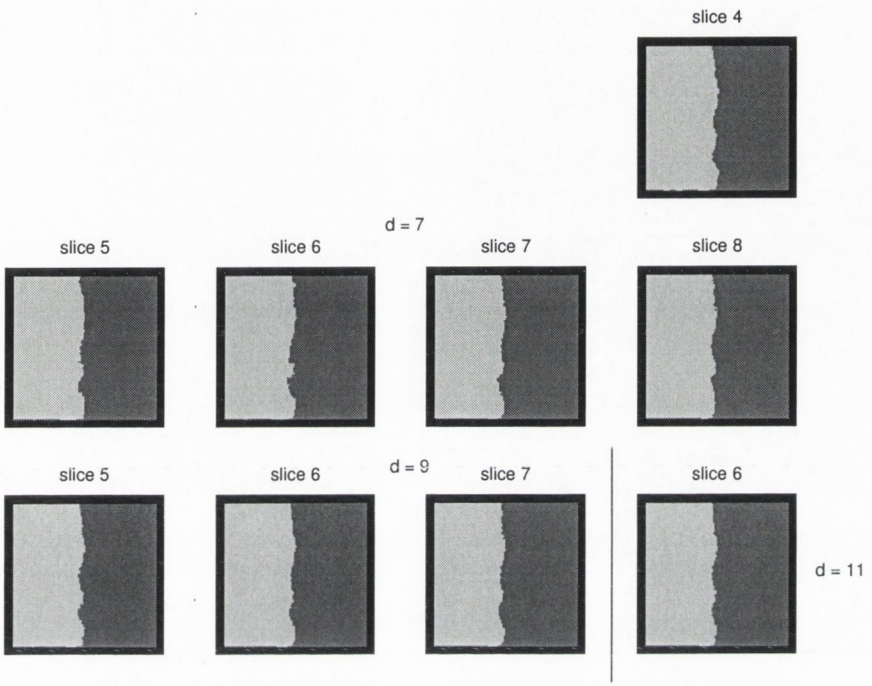

Figure D.18: 3-D RUR segmentation results for the volume from Fig. C.7, $d = 7, 9, 11$.

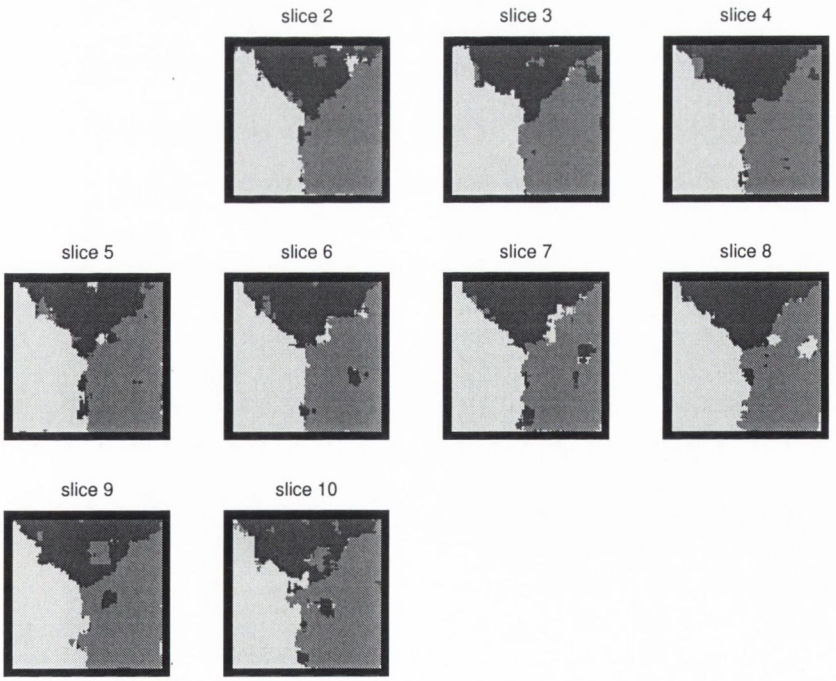

Figure D.19: 3-D RUR segmentation results for the volume from Fig. C.8, $d = 3$.

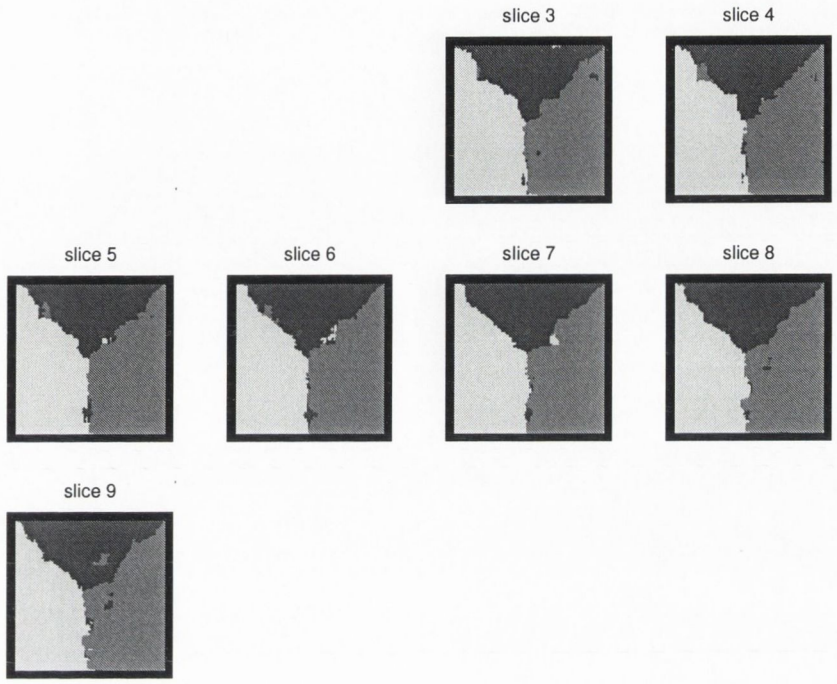

Figure D.20: 3-D RUR segmentation results for the volume from Fig. C.8, $d = 5$.

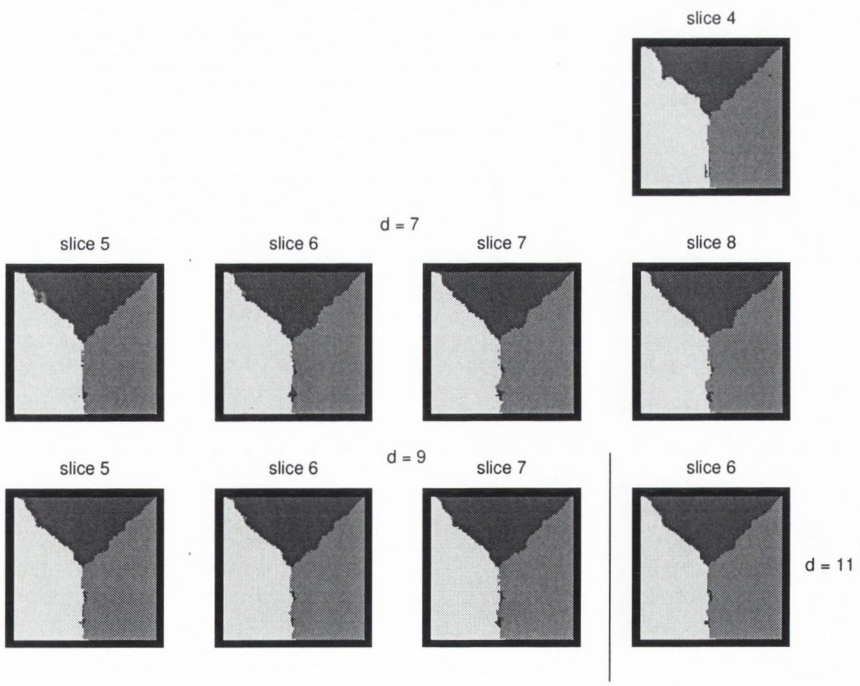

Figure D.21: 3-D RUR segmentation results for the volume from Fig. C.8, $d = 7, 9, 11$.

D.5 TEMPORAL TEXTURES

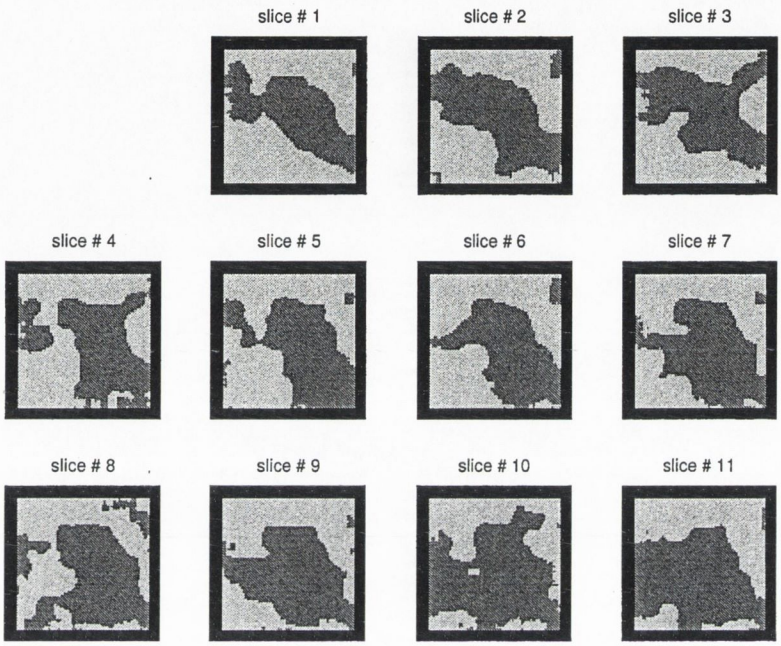

Figure D.22: 2-D RUR segmentation of the temporal textures sequence from Fig. C.17

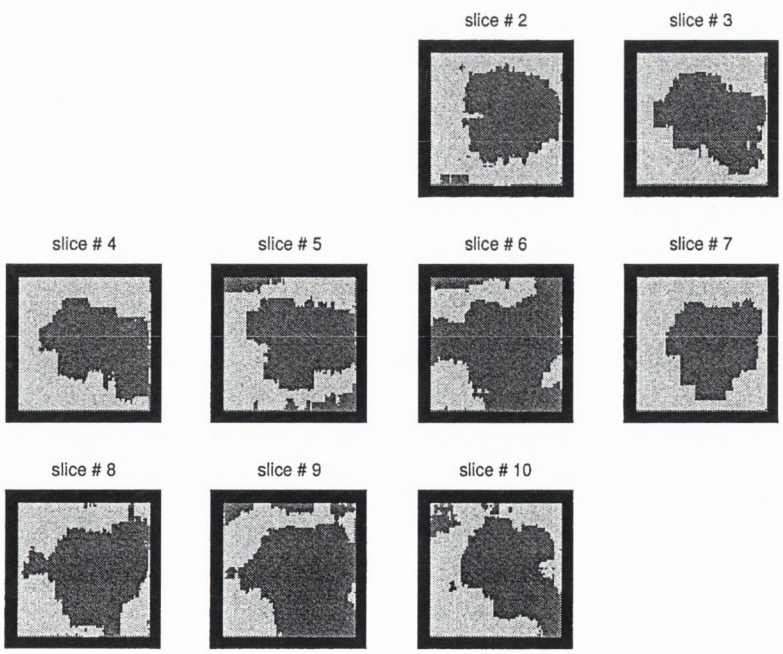

Figure D.23: 3-D RUR segmentation of the temporal textures sequence from Fig. C.17, $d = 3$

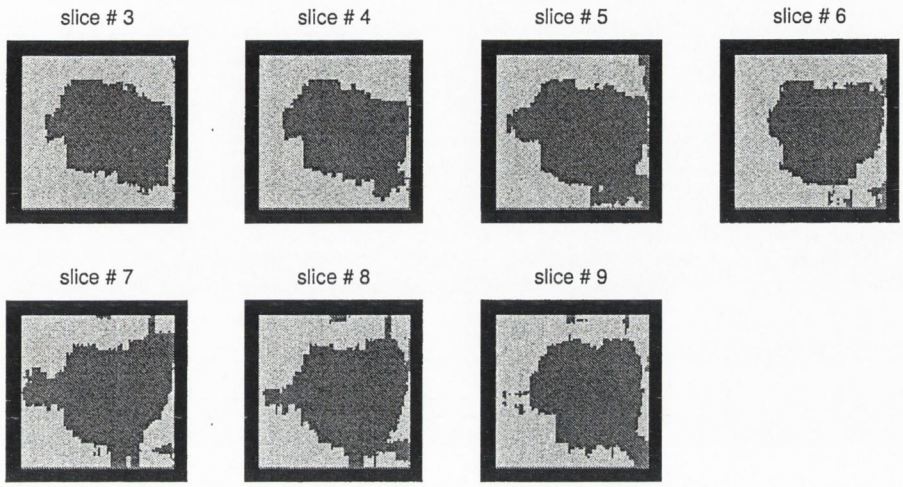

Figure D.24: 3-D RUR segmentation of the temporal textures sequence from Fig. C.17, $d = 5$

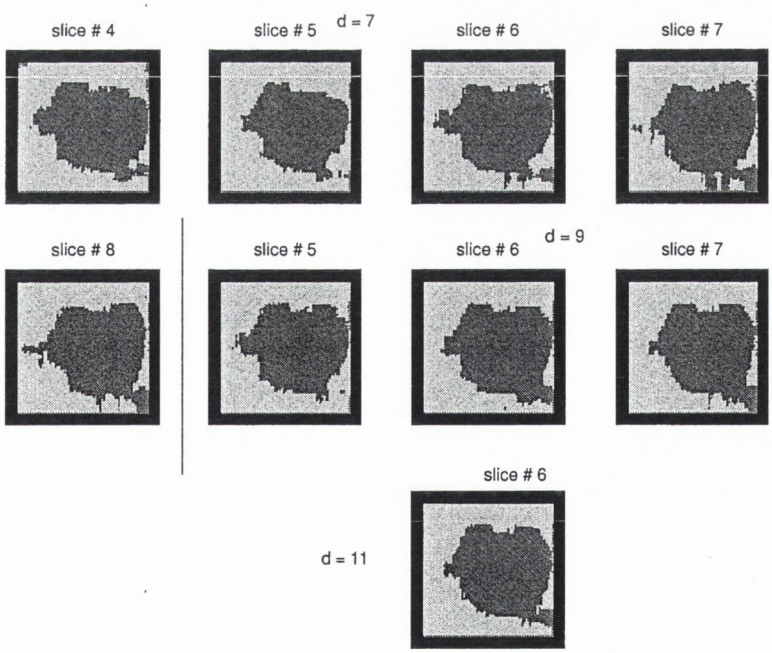

Figure D.25: 3-D RUR segmentation of the temporal textures sequence form Fig. C.17, $d = 7, 9, 11$

Appendix E

DIFFERENCE COMPENSATED SEGMENTATION

E.1 INTEREST POINTS DETECTION AND MATCHING

E.1.1 Modified MO

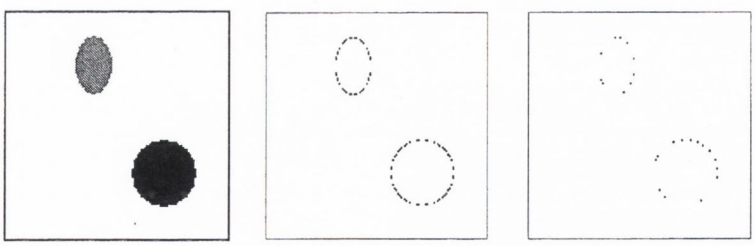

Figure E.1: Modified MO. Original image: three class mask, all corners detected and the selected IPs.

Figure E.2: Modified MO. 3-D non-compensated 'thin-plate' RUR segmentation for slice #5 from the volume in Fig. C.11, all corners detected and the selected IPs.

Figure E.3: Modified MO. 3-D non-compensated 'thin-plate' RUR segmentation for slice #6 from the volume in Fig. C.11, all corners detected and the selected IPs.

Table E.1: Number of IP detected by the modified MO and elapsed time [s].

Fig. #	**Total # corners**	**# IPs**	**El.time**
E.1	92	24	0.53
E.2	69	13	0.46
E.3	73	16	0.41

E.1.2 Matching Algorithm

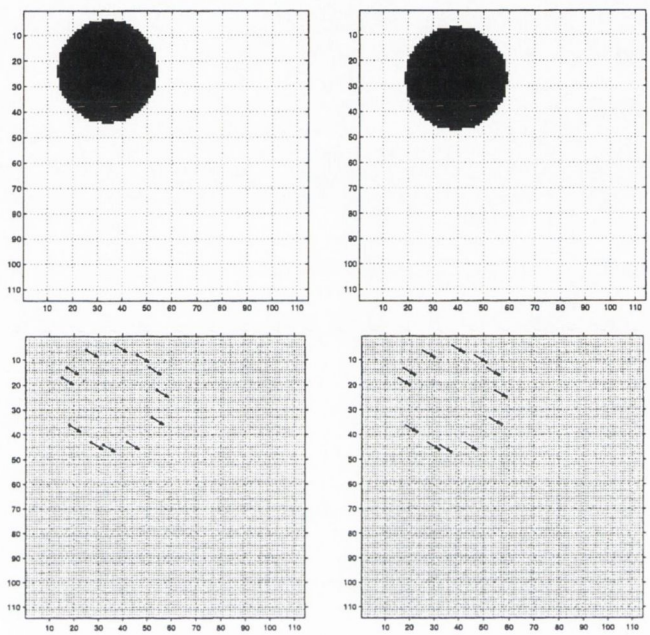

Figure E.4: Circular object masks displaced by $(3, 2)$, the difference vectors estimated in respect to IP1 and IP2 respectively.

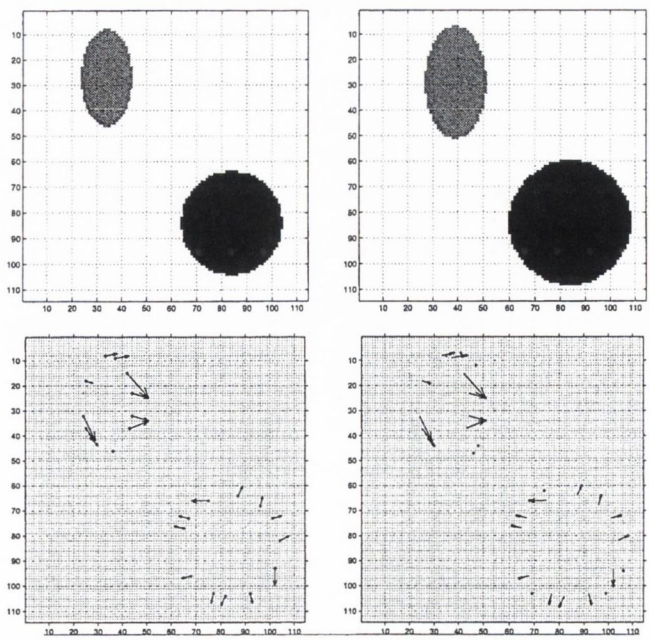

Figure E.5: Original images: slices #5 and #6 from a test volume, the difference vectors estimated in the set of IPs.

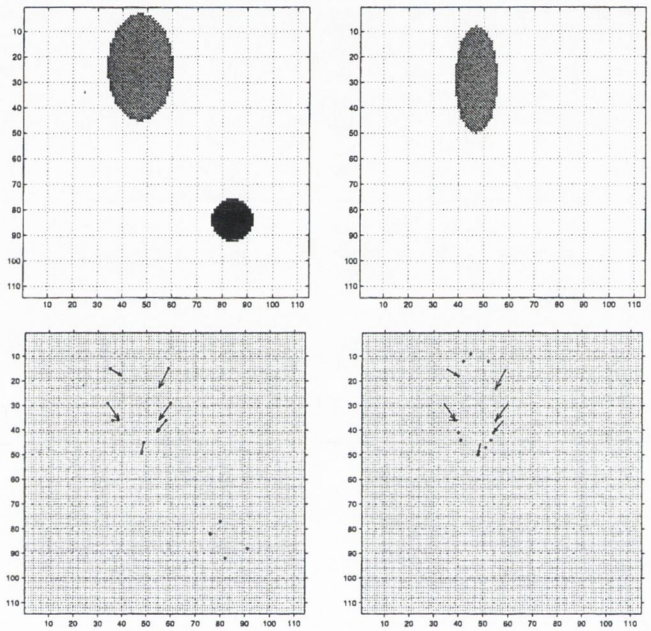

Figure E.6: Original images: slices #8 and #9 from a test volume, the difference vectors estimated in the set of IPs.

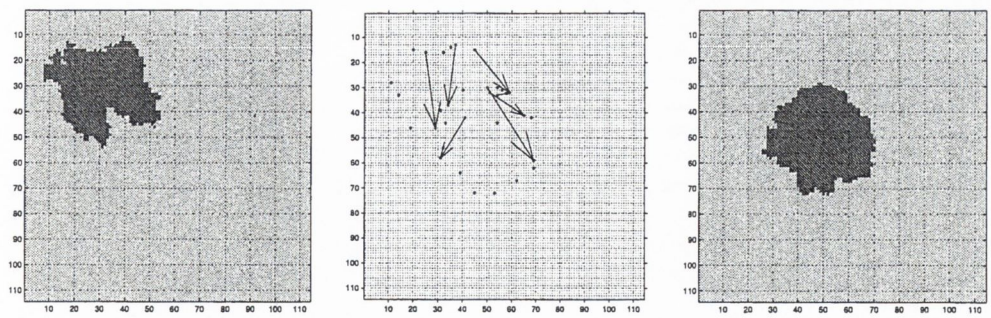

Figure E.7: Original images and superimposed IPs: 3-D non-compensated 'thin-plate' segmentation of slices # 2 and # 6 from the volume in Fig. C.11 and the estimated sparse difference field between them.

E.1.3 Compensated Segmentation Results

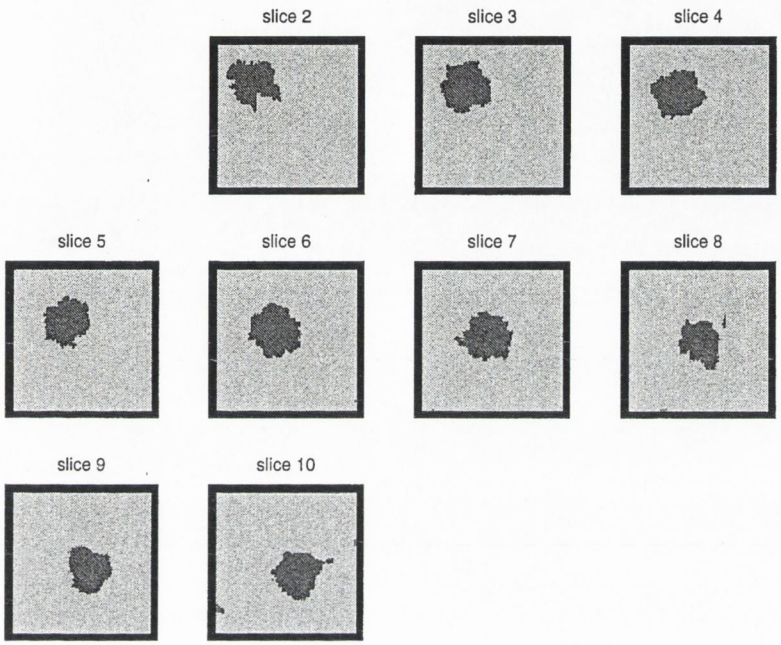

Figure E.8: 3-D RUR compensated segmentation results for the 'skewed' cylinder volume (Fig. C.11), $d = 3$. Difference field estimated via IP detection and matching.

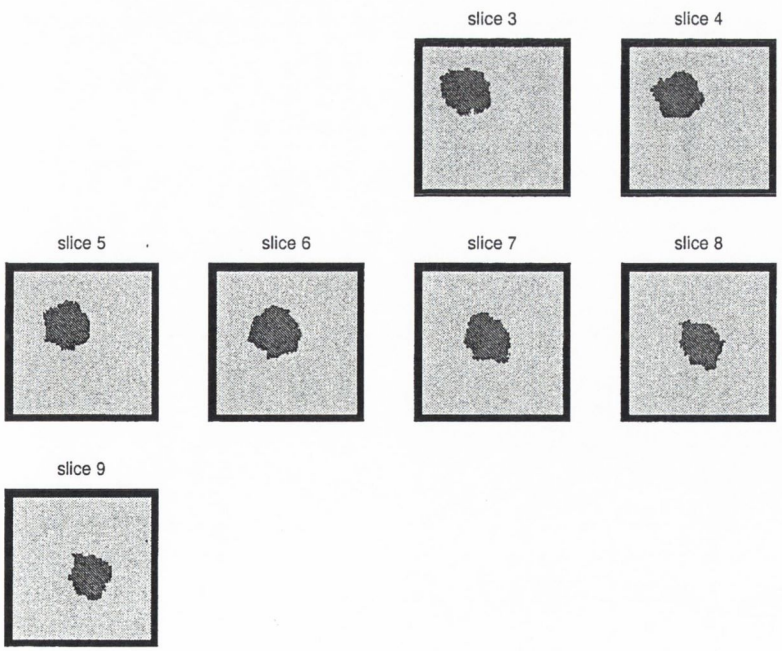

Figure E.9: 3-D RUR compensated segmentation results for the 'skewed' cylinder volume (Fig. C.11), $d = 5$. Difference field estimated via IP detection and matching.

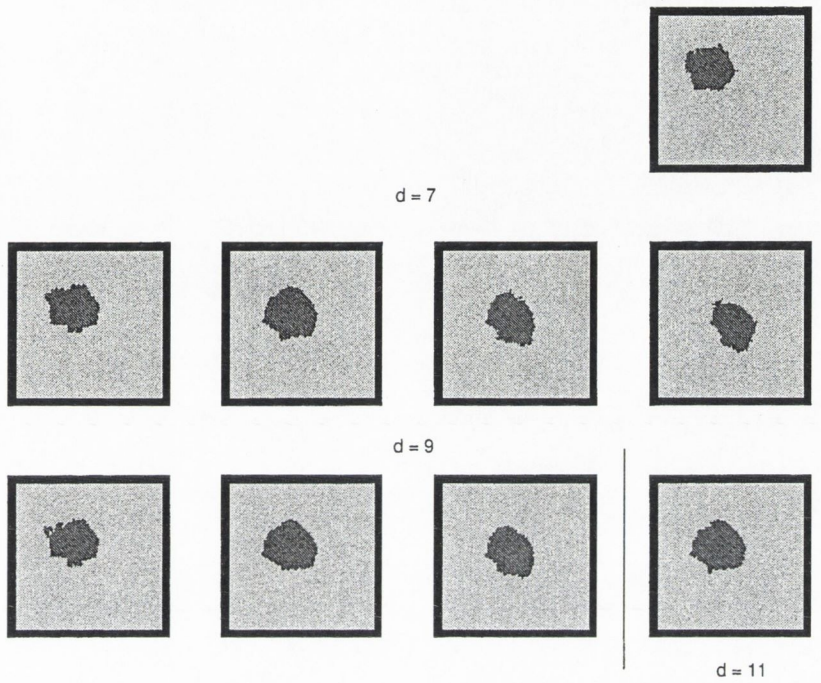

Figure E.10: 3-D RUR compensated segmentation results for the 'skewed' cylinder volume (Fig. C.11), $d = 7, 9, 11$. Difference field estimated via IP detection and matching.

E.2 CROSS-CORRELATION

E.2.1 2-D Correlation Coefficient

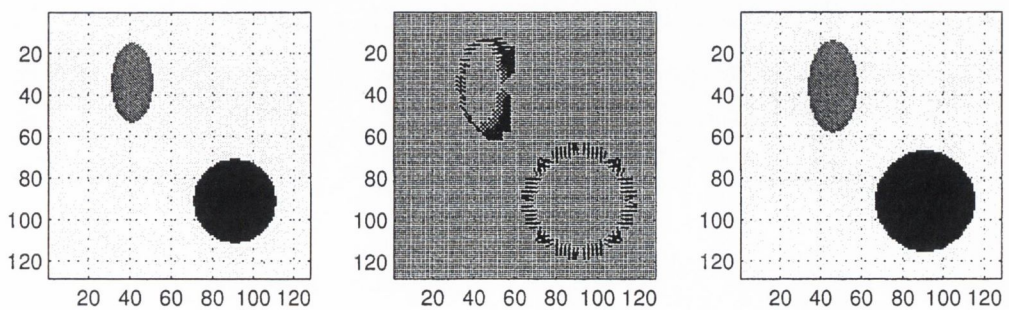

Figure E.11: Original grey level images and the difference vectors estimated via ZNCC between them.

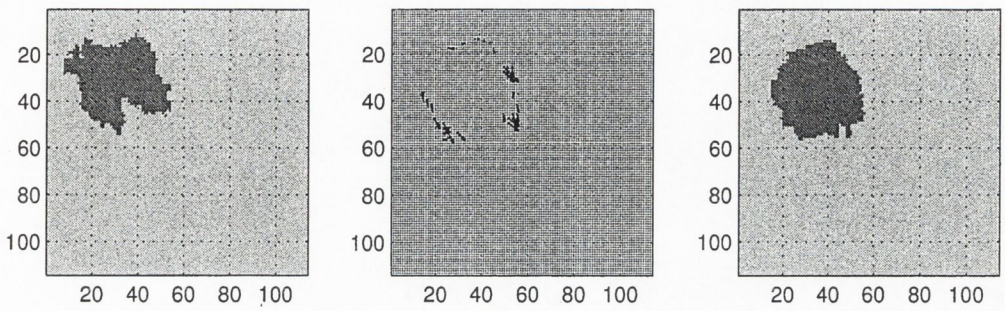

Figure E.12: 3-D non-compensated RUR segmentation results, $d = 3$, for slices #2 and #3 of the 'skewed' cylinder volume (Fig.C.11) and the difference field estimated via ZNCC between them.

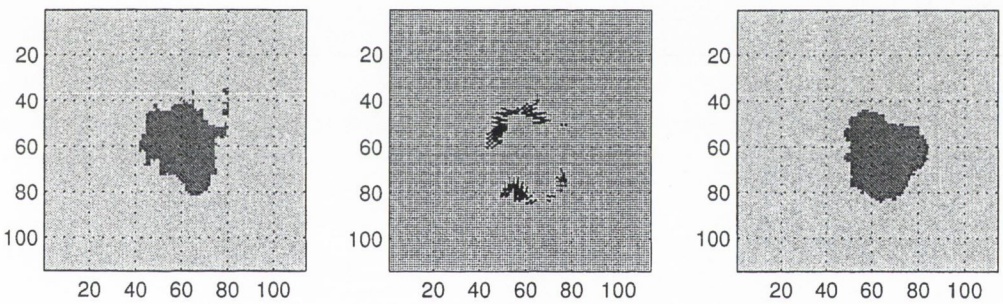

Figure E.13: 3-D non-compensated RUR segmentation results, $d = 3$, for slices #8 and #9 of the 'skewed' cylinder volume (Fig.C.11) and the difference field estimated via ZNCC between them.

E.2.2 Compensated Segmentation Results

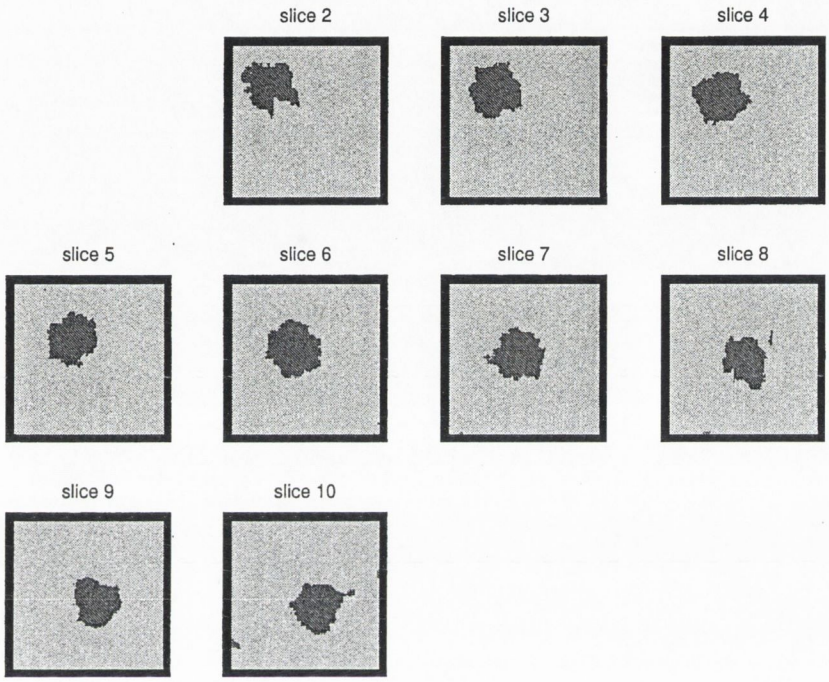

Figure E.14: 3-D RUR compensated segmentation results for the 'skewed' cylinder volume (Fig. C.11), $d = 3$. Difference field estimated via ZNCC.

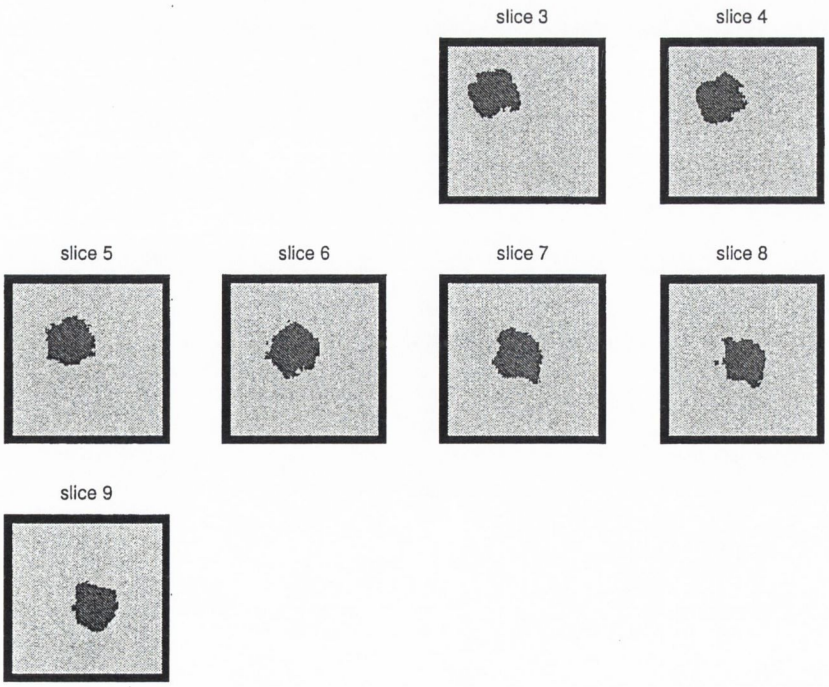

Figure E.15: 3-D RUR compensated segmentation results for the 'skewed' cylinder volume (Fig. C.11), $d = 5$. Difference field estimated via ZNCC.

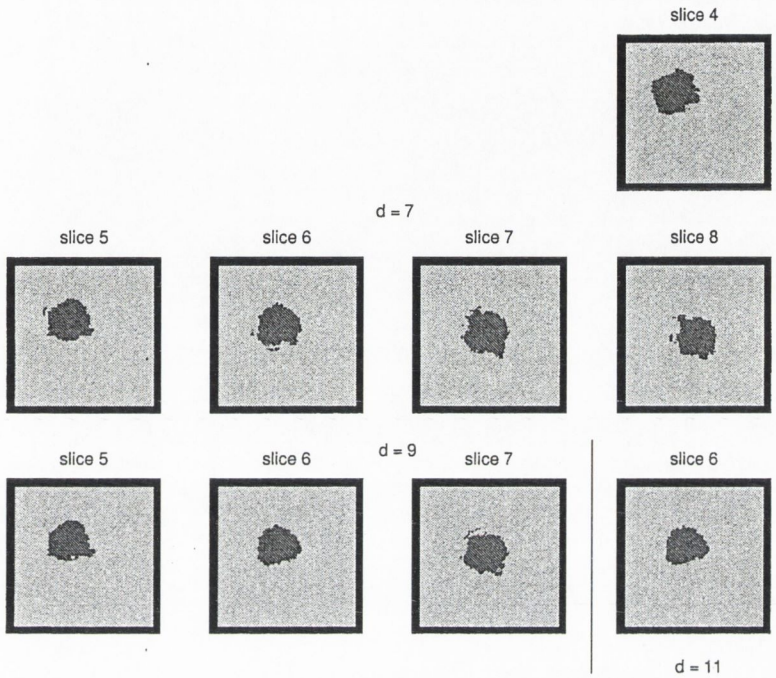

Figure E.16: 3-D RUR compensated segmentation results for the 'skewed' cylinder volume (Fig. C.11), $d = 7, 9, 11$. Difference field estimated via ZNCC.

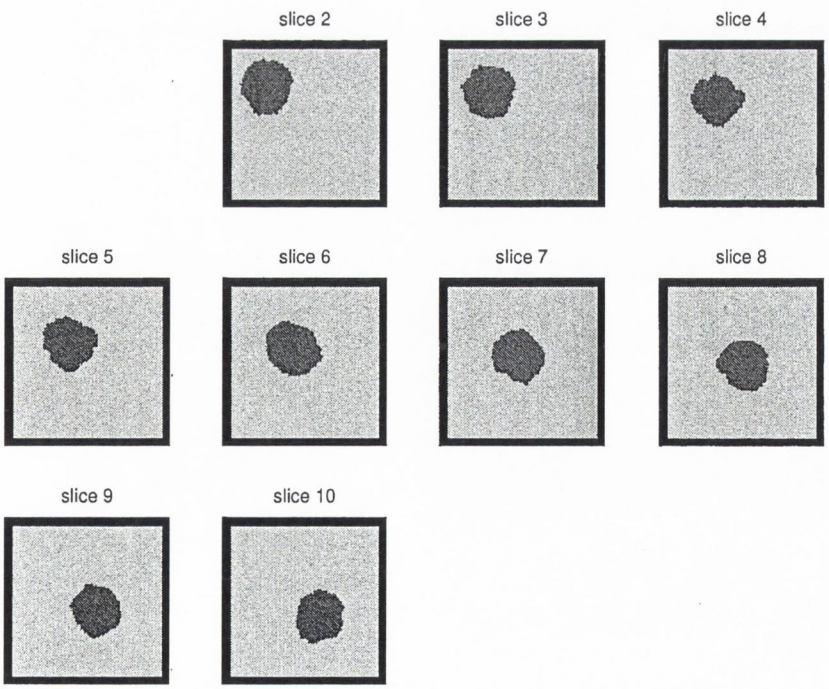

Figure E.17: 3-D RUR compensated segmentation results for the 'skewed' cylinder volume (Fig. C.12), $d = 3$. Difference field estimated via ZNCC.

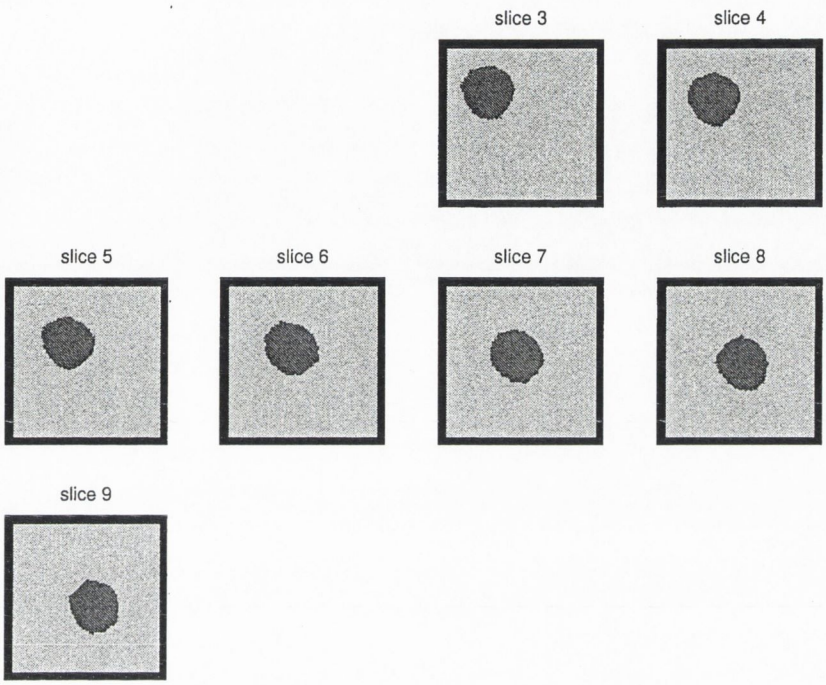

Figure E.18: 3-D RUR compensated segmentation results for the 'skewed' cylinder volume (Fig. C.12), $d = 5$. Difference field estimated via ZNCC.

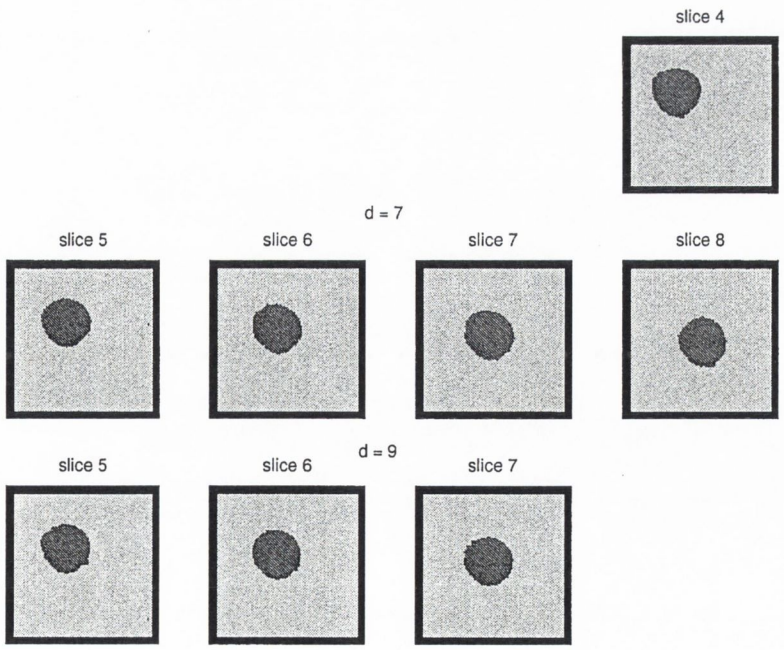

Figure E.19: 3-D RUR compensated segmentation results for the 'skewed' cylinder volume (Fig. C.12), $d = 7, 9$. Difference field estimated via ZNCC.

E.3 KULLBACK-LEIBLER MINIMIZATION

E.3.1 Difference Estimation

Parameters: $L_b = 5$ $d_b = 3$, $step = 2$; $d_{max} = 5$, $\Delta = 1$. Elapsed time: $25.32s$..

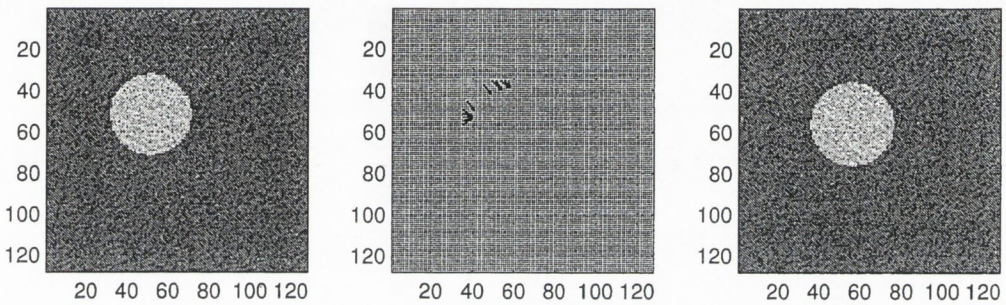

Figure E.20: Slices #5 and #6 from a 'skewed' cylinder volume of textures with different mean and the difference field estimated via KLD minimization between them.

E.3.2 Compensated Segmentation Results

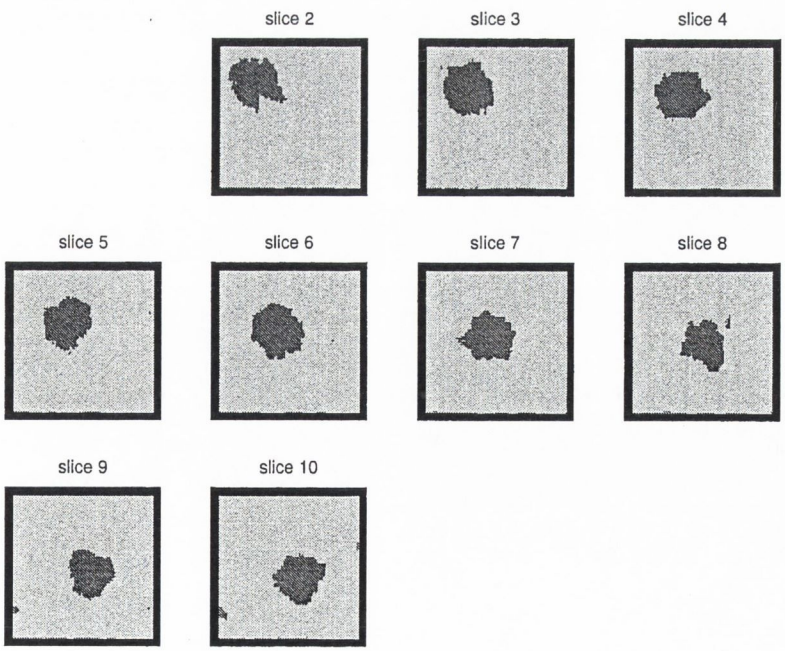

Figure E.21: 3-D RUR compensated segmentation results for the 'skewed' cylinder volume (Fig. C.11), $d = 3$. Difference field estimated via KLD minimization.

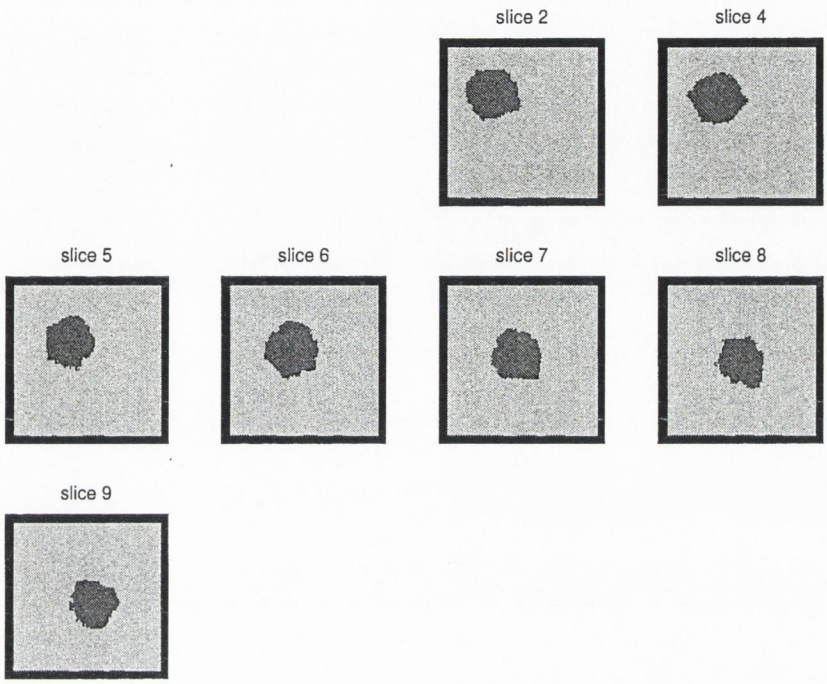

Figure E.22: 3-D RUR compensated segmentation results for the 'skewed' cylinder volume (Fig. C.12), $d = 5$. Difference field estimated via KLD minimization.

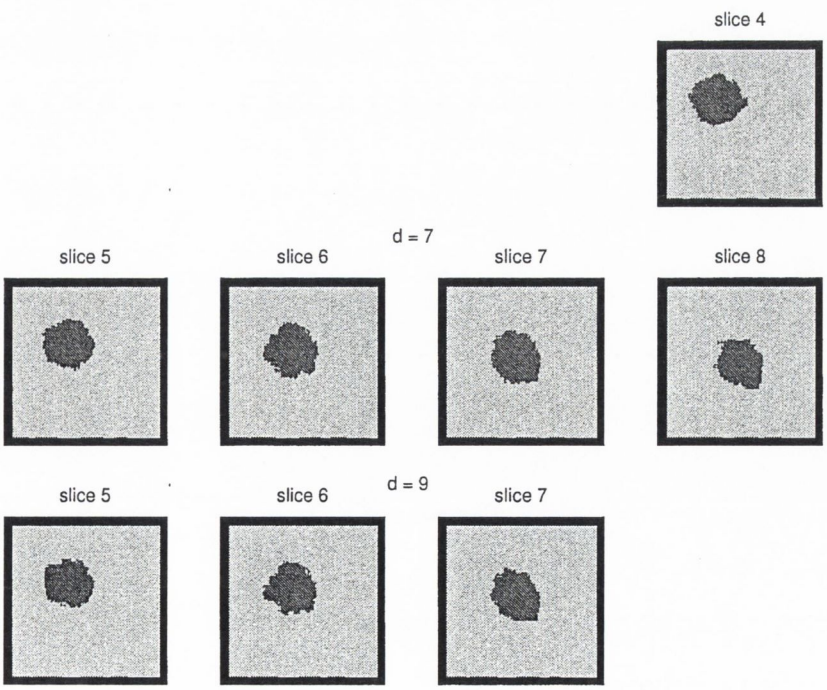

Figure E.23: 3-D RUR compensated segmentation results for the 'skewed' cylinder volume (Fig. C.12), $d = 7, 9$. Difference field estimated via KLD minimization.

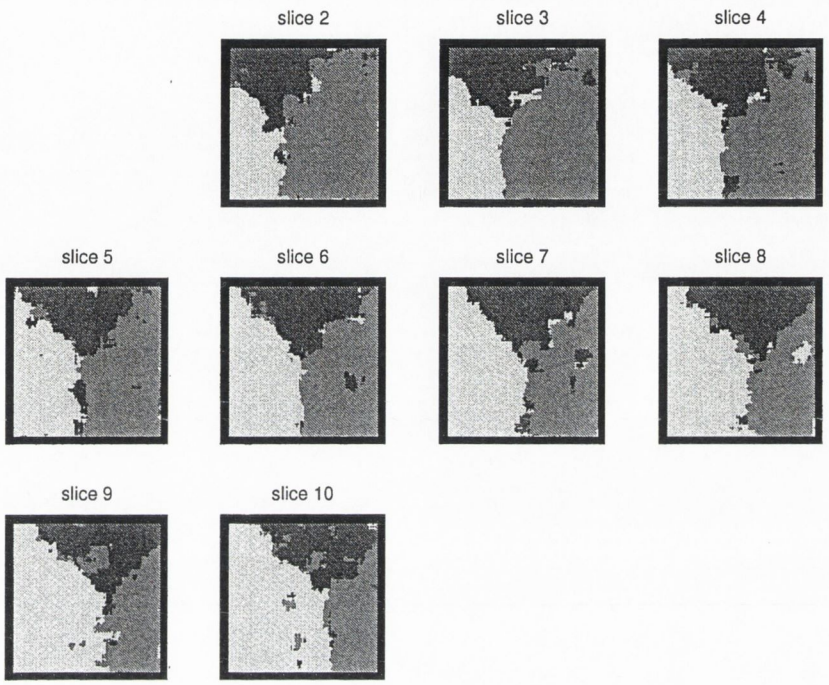

Figure E.24: 3-D RUR non-compensated segmentation results for the three textures volume (Fig.C.14), $d = 3$.

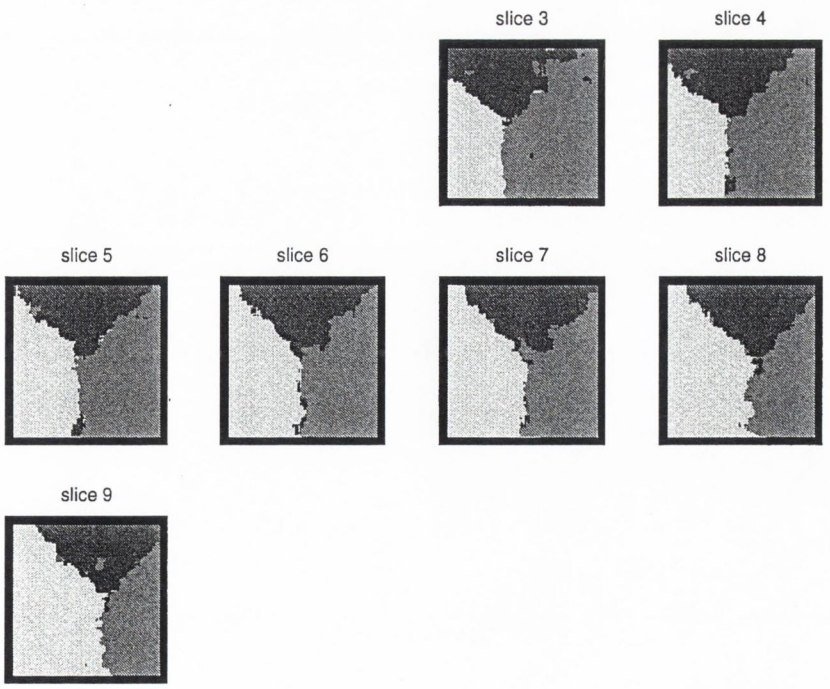

Figure E.25: 3-D RUR non-compensated segmentation results for the three textures volume (Fig.C.14), $d = 5$.

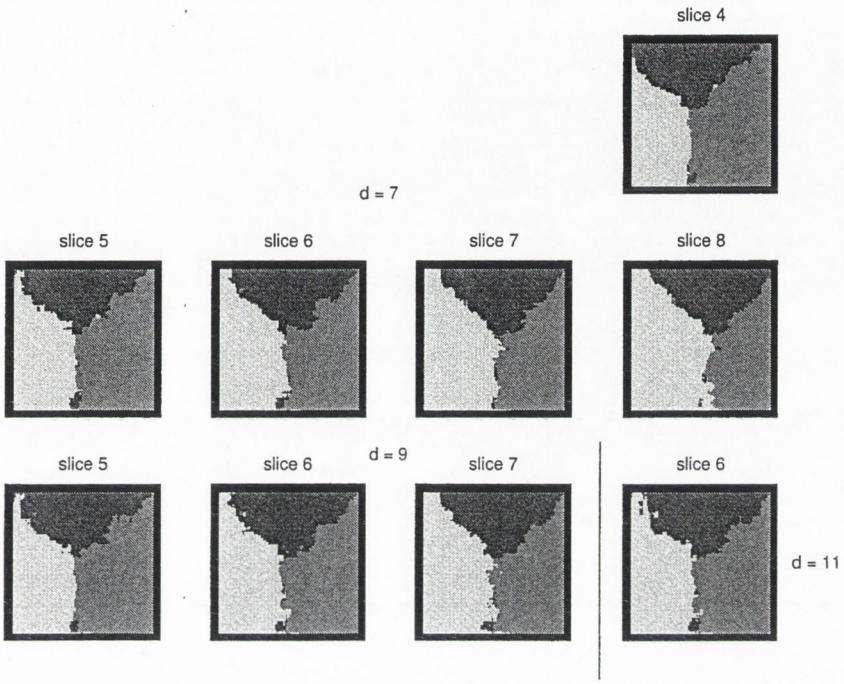

Figure E.26: 3-D RUR non-compensated segmentation results for the three textures volume (Fig.C.14), $d = 7, 9$.

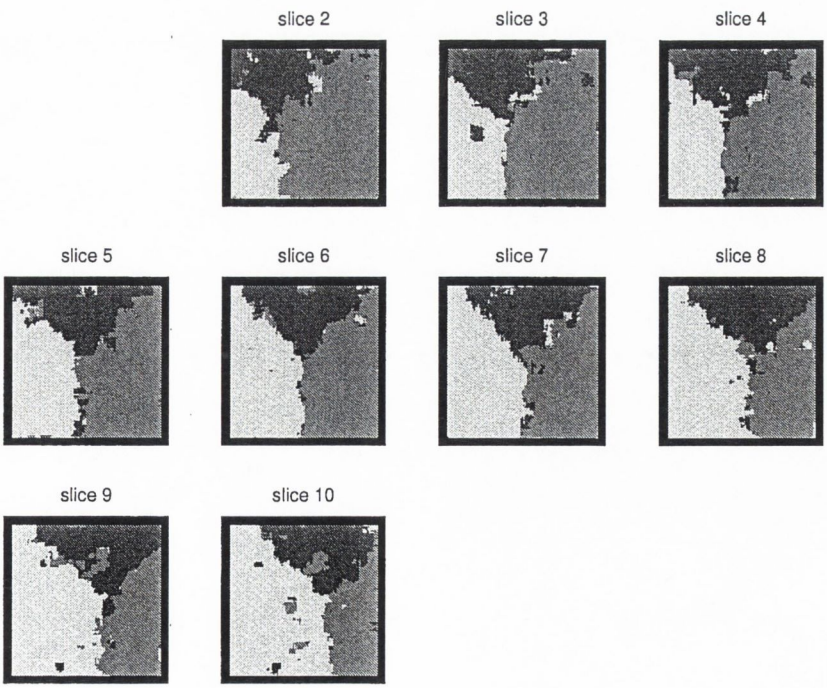

Figure E.27: 3-D RUR compensated segmentation results for the three textures volume (Fig.C.14), $d = 3$. Exact difference field.

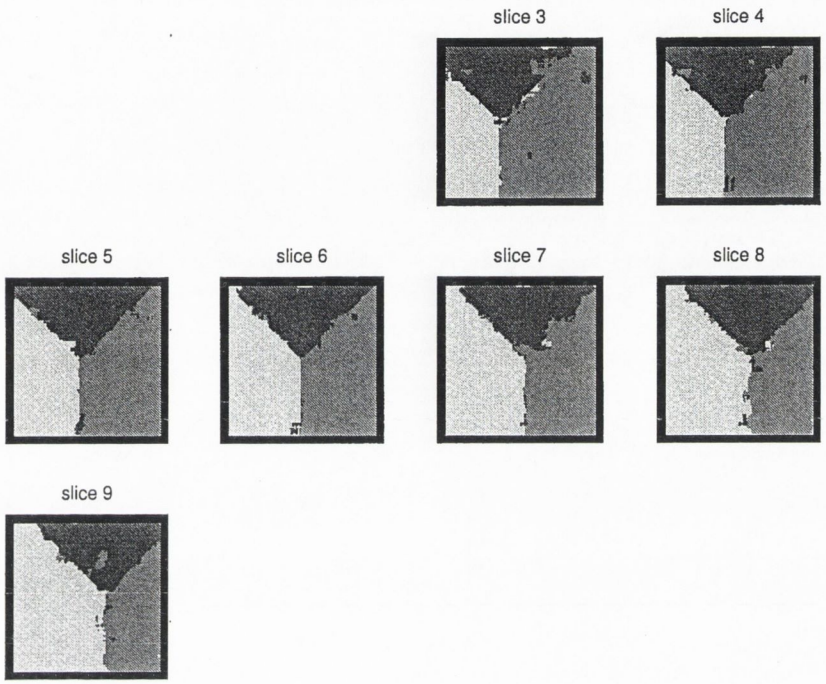

Figure E.28: 3-D RUR compensated segmentation results for the three textures volume (Fig.C.14), $d = 5$. Exact difference field.

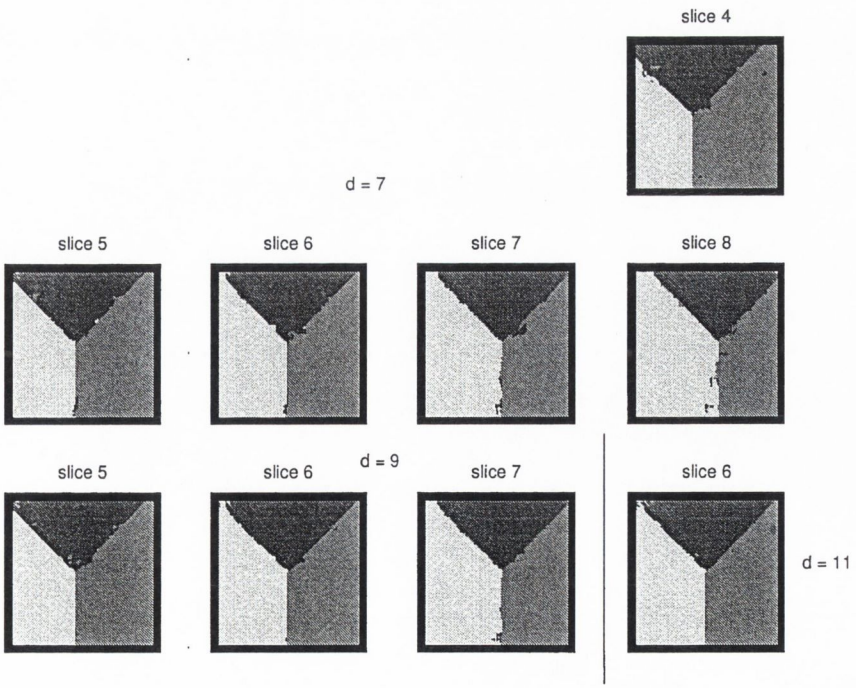

Figure E.29: 3-D RUR compensated segmentation results for the three textures volume (Fig.C.14), $d = 7, 9, 11$. Exact difference field.

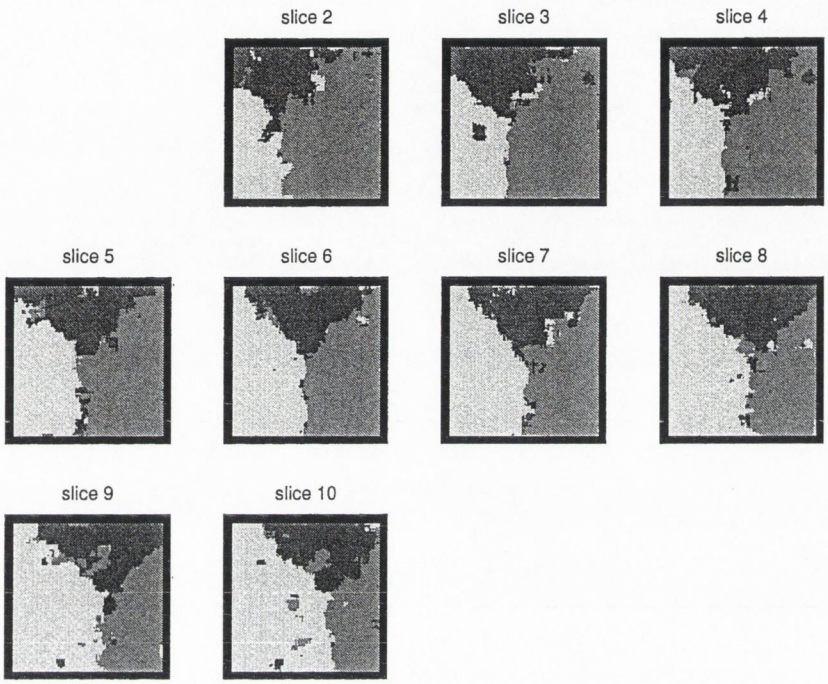

Figure E.30: 3-D RUR compensated segmentation results for the three textures volume (Fig.C.14), $d = 3$. Difference field estimated via KLD minimization.

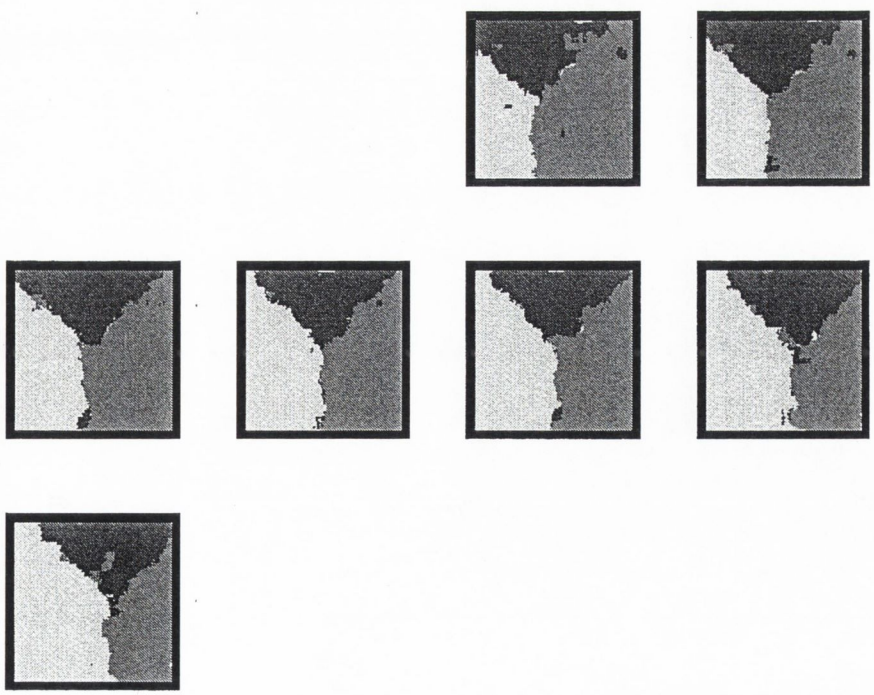

Figure E.31: 3-D RUR compensated segmentation results for the three textures volume (Fig.C.14), $d = 5$. Difference field estimated via KLD minimization.

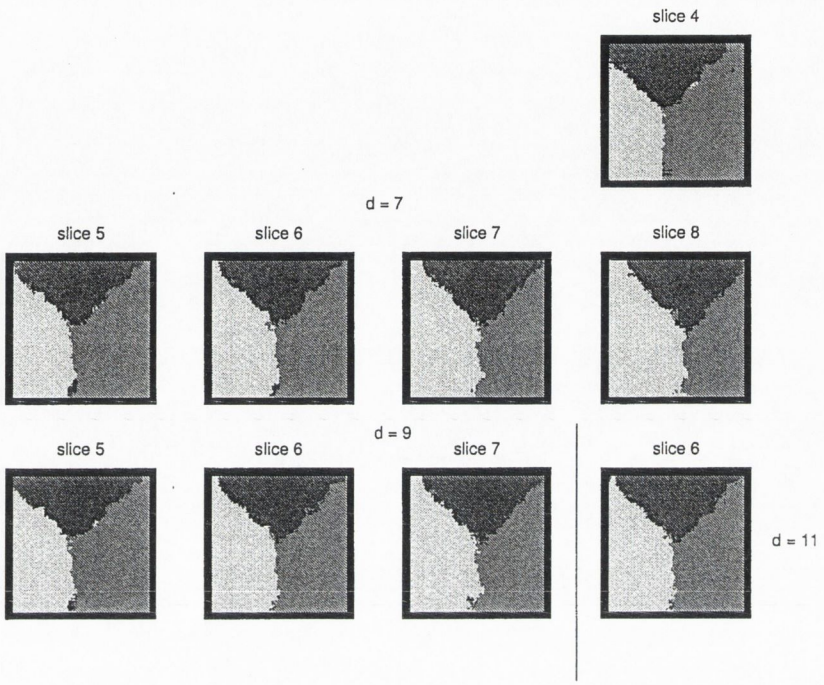

Figure E.32: 3-D RUR compensated segmentation results for the three textures volume (Fig.C.14), $d = 7, 9, 11$. Difference field estimated via KLD minimization.

E.4 MRI SEGMENTATION

E.4.1 Difference Estimation

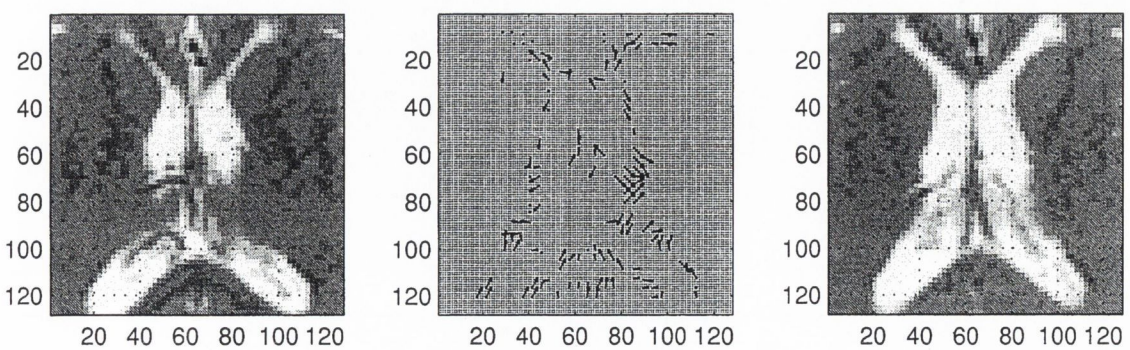

Figure E.33: Slices 30 and 31 from the volume in Fig. C.20 and the difference vectors estimated via ZNCC between them.

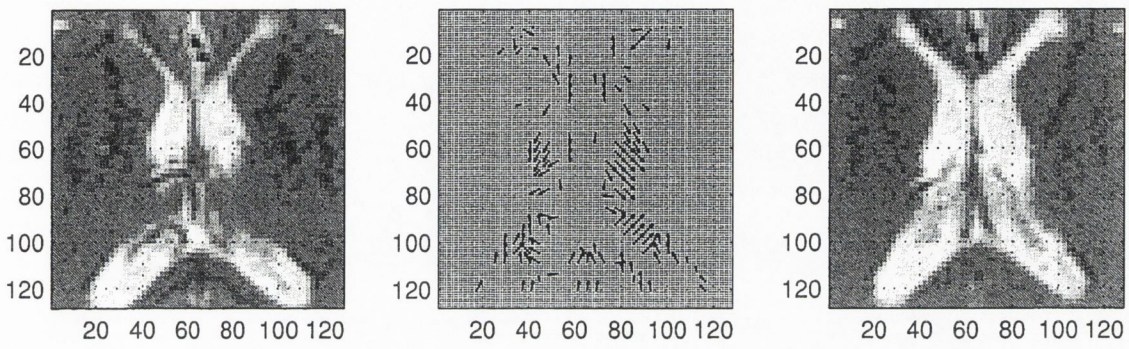

Figure E.34: Slices 30 and 31 from the volume in Fig. C.20 and the difference vectors estimated via 3-D ZNCC between them.

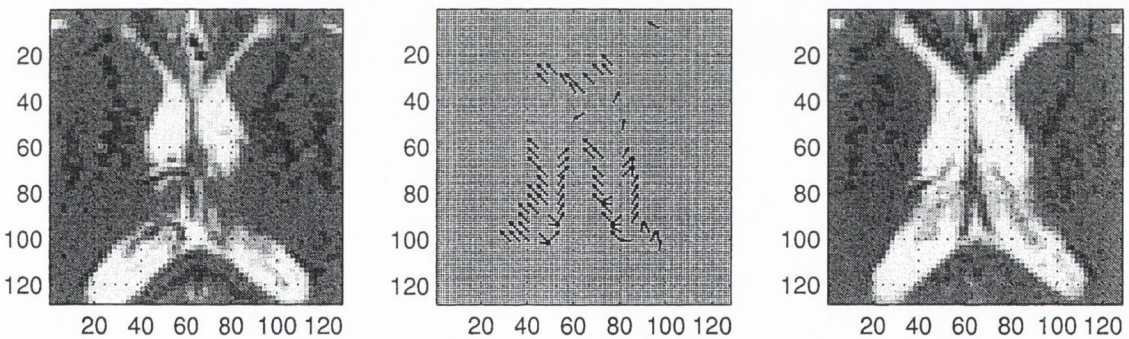

Figure E.35: Slices 30 and 31 from the volume in Fig. C.20 and the difference vectors estimated via KLD between them.

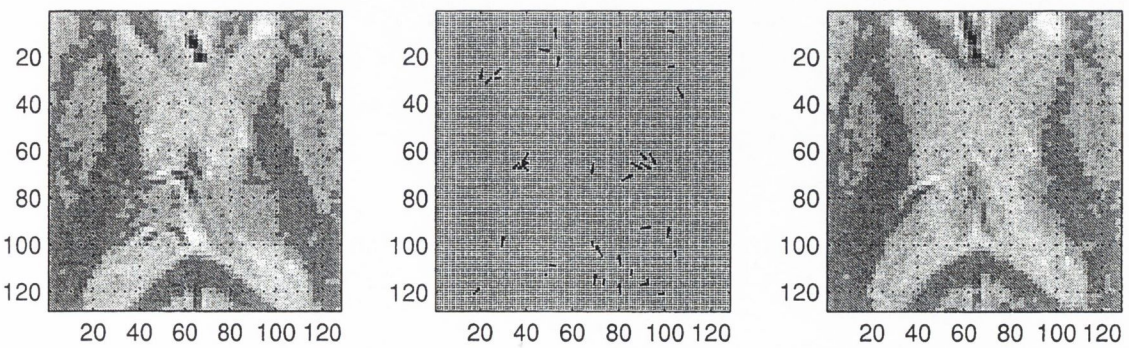

Figure E.36: Slices 30 and 31 from the volume in Fig. C.21 and the difference vectors estimated via ZNCC between them.

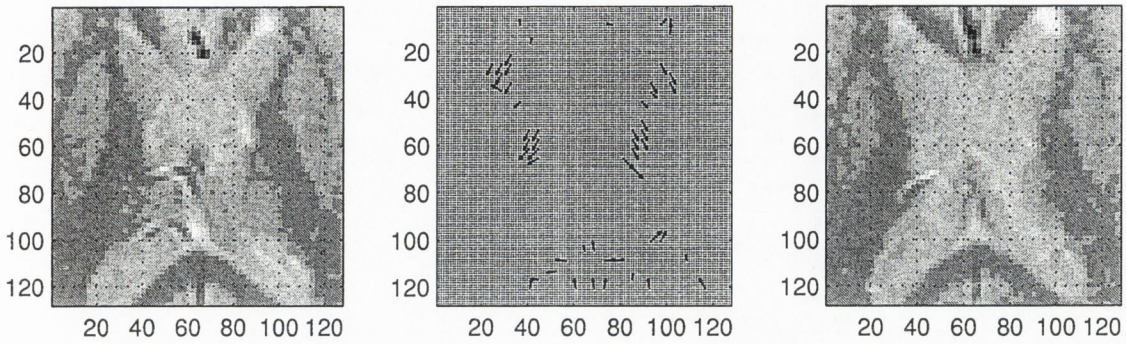

Figure E.37: Slices 30 and 31 from the volume in Fig. C.21 and the difference vectors estimated via 3-D ZNCC between them.

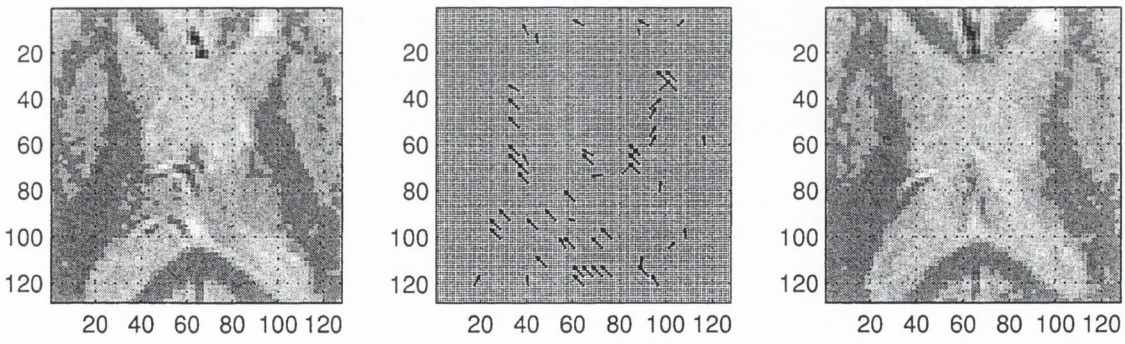

Figure E.38: Slices 30 and 31 from the volume in Fig. C.21 and the difference vectors estimated via KLD between them.

E.4.2 Segmentation

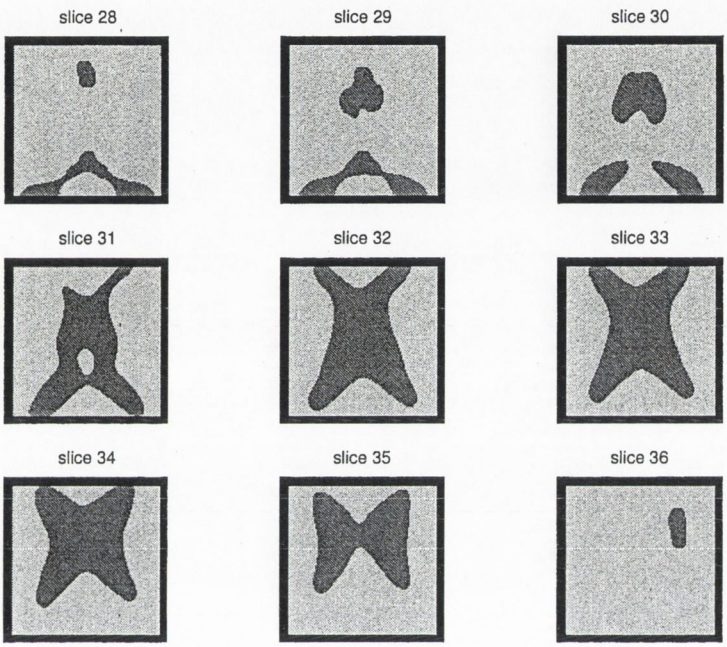

Figure E.39: 2-D RUR segmentation results for the T2 MRI volume (Fig.C.20).

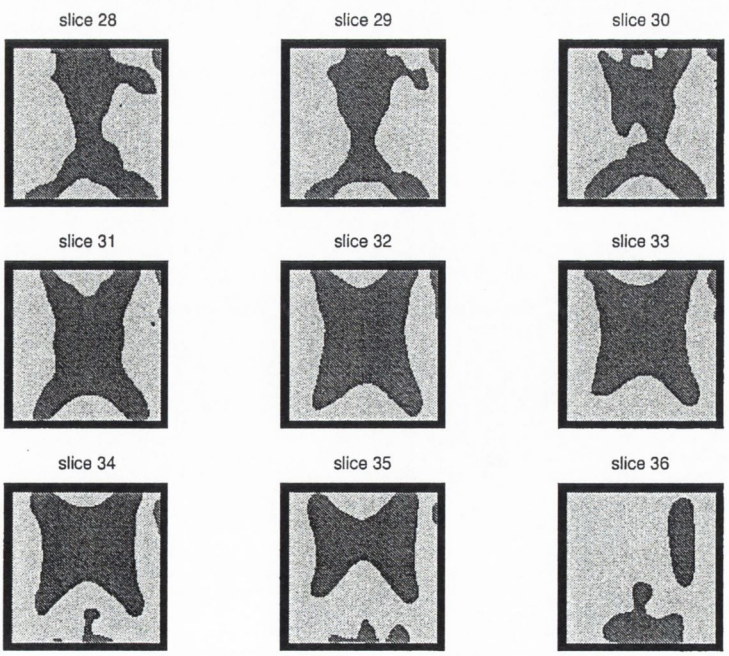

Figure E.40: 2-D RUR segmentation results for the PD MRI volume (Fig. C.21).

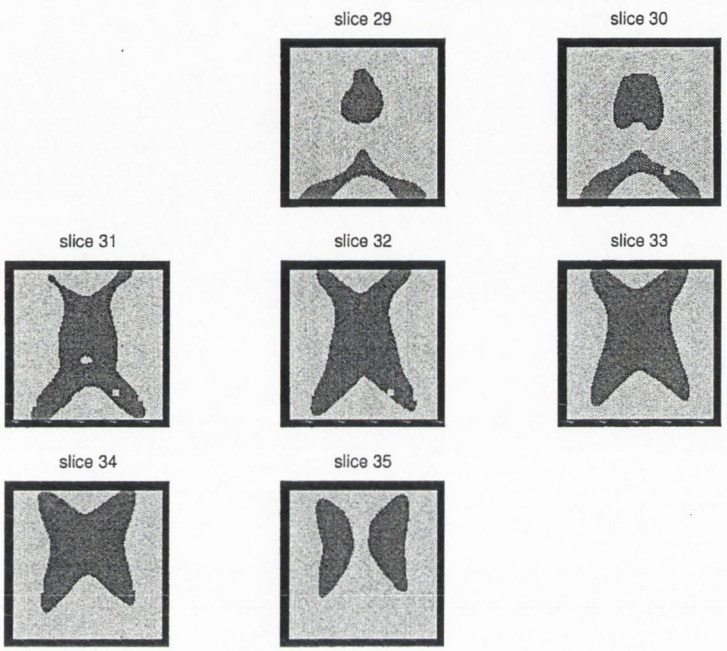

Figure E.41: 3-D RUR non-compensated segmentation results for the T2 MRI volume (Fig. C.20), $d = 3$.

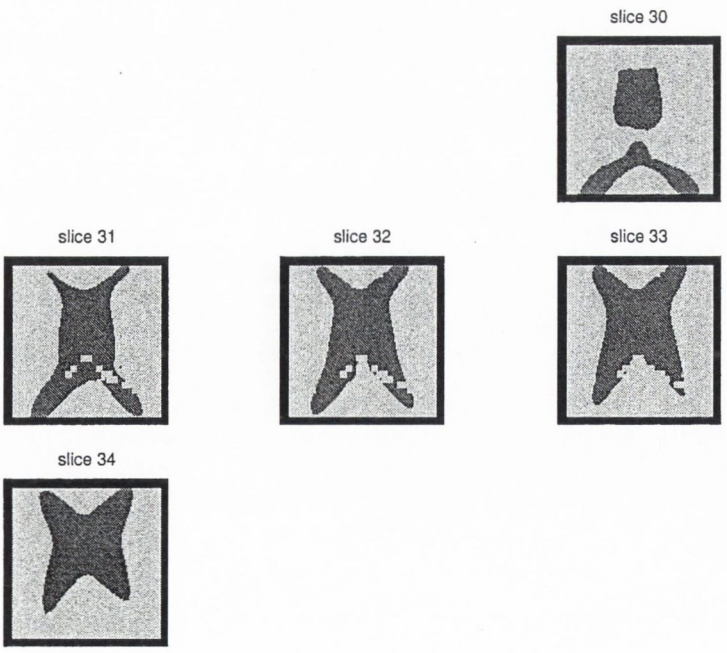

Figure E.42: 3-D RUR non-compensated segmentation results for the T2 MRI volume (Fig. C.20), $d = 5$.

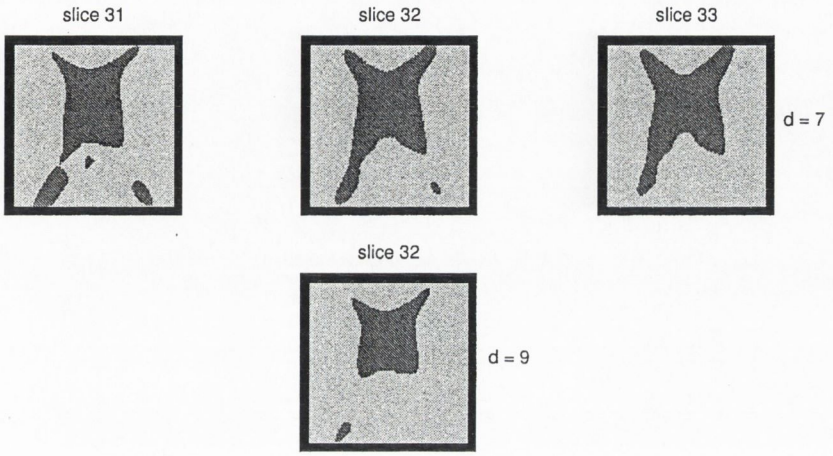

Figure E.43: 3-D RUR non-compensated segmentation results for the T2 MRI volume (Fig. C.20), $d = 7, 9$.

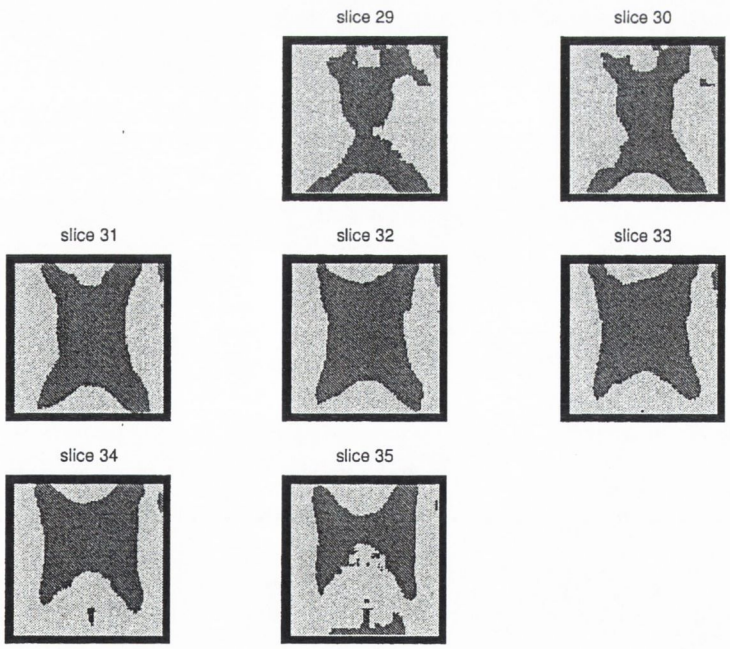

Figure E.44: 3-D RUR non-compensated segmentation results for the PD MRI volume (Fig. C.21), $d = 3$.

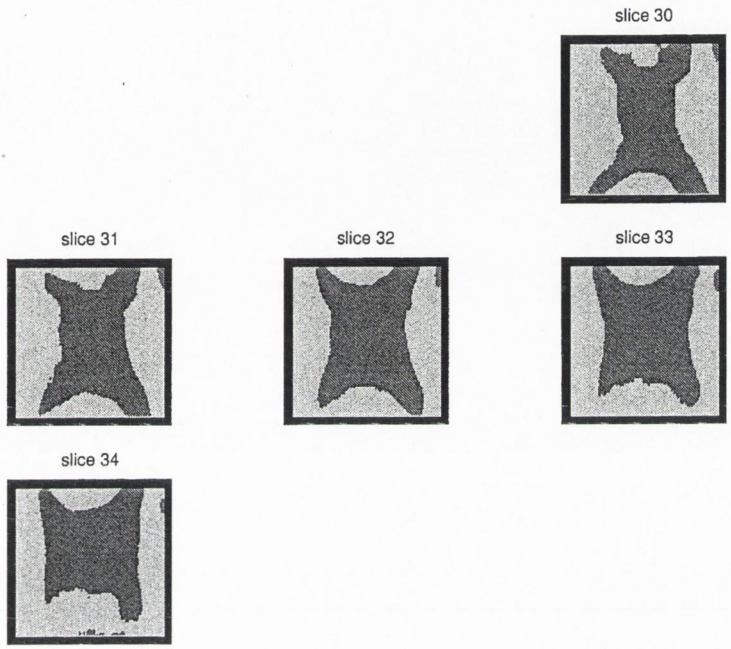

Figure E.45: 3-D RUR non-compensated segmentation results for the PD MRI volume (Fig. C.21), $d = 5$.

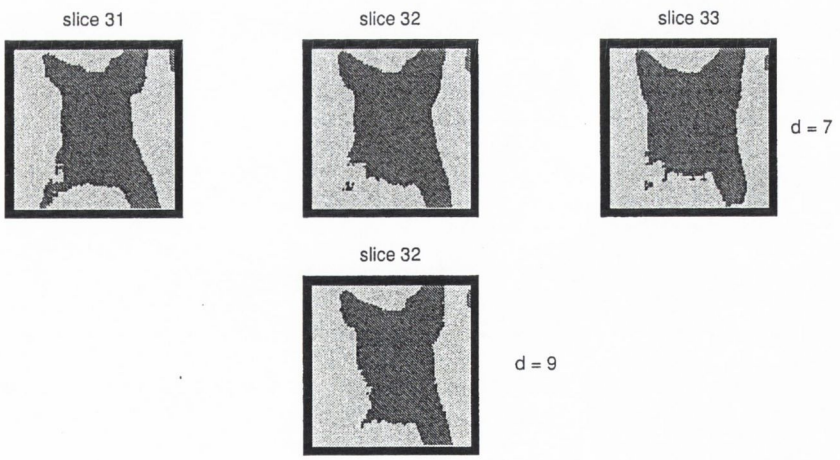

Figure E.46: 3-D RUR non-compensated segmentation results for the PD MRI volume (Fig. C.21), $d = 7, 9$.

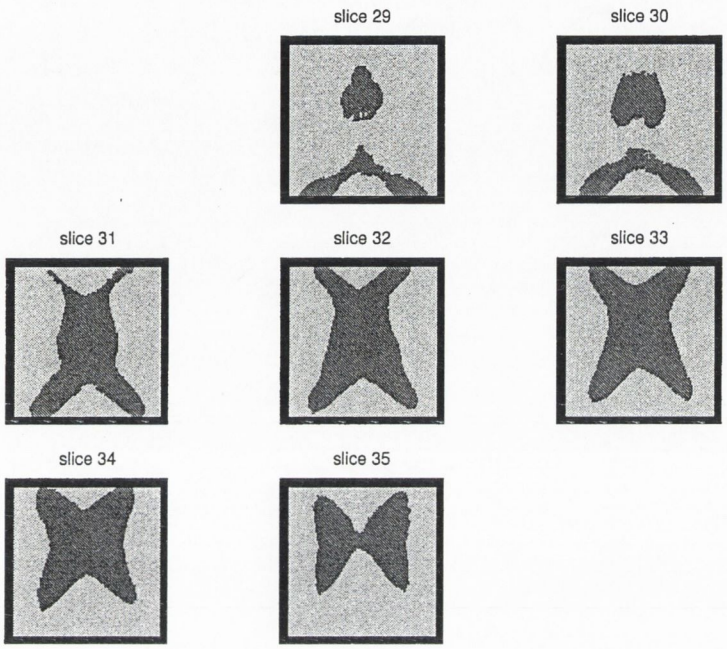

Figure E.47: 3-D RUR compensated segmentation results for the T2 MRI volume (Fig. C.20), $d = 3$. Difference estimation via 2-D ZNCC.

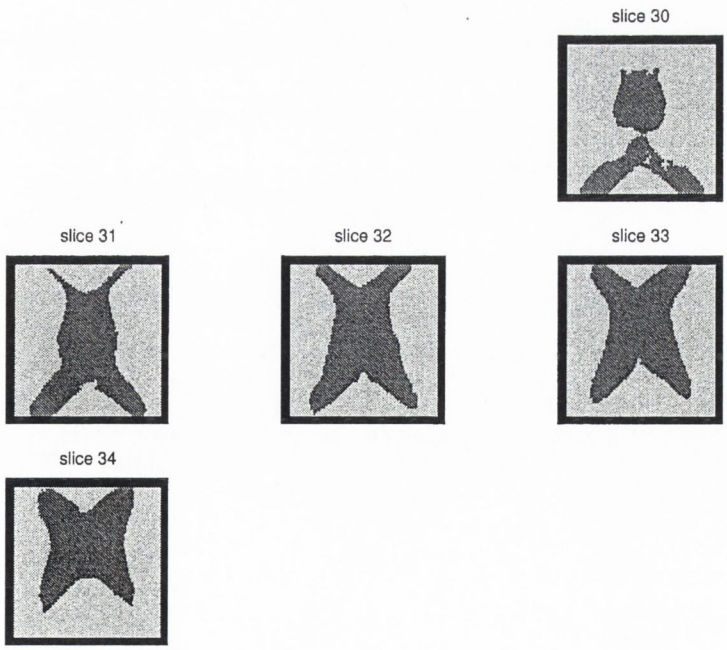

Figure E.48: 3-D RUR compensated segmentation results for the T2 MRI volume (Fig. C.20), $d = 5$. Difference estimation via 2-D ZNCC.

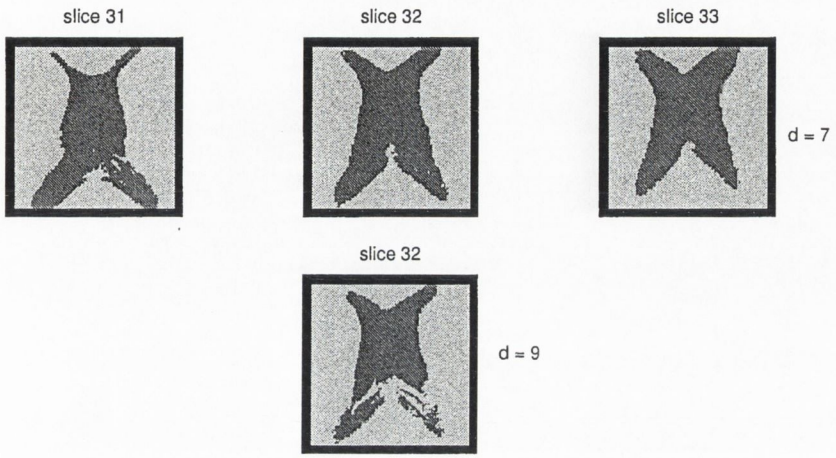

Figure E.49: 3-D RUR compensated segmentation results for the T2 MRI volume (Fig. C.20), $d = 7, 9$. Difference estimation via 2-D ZNCC.

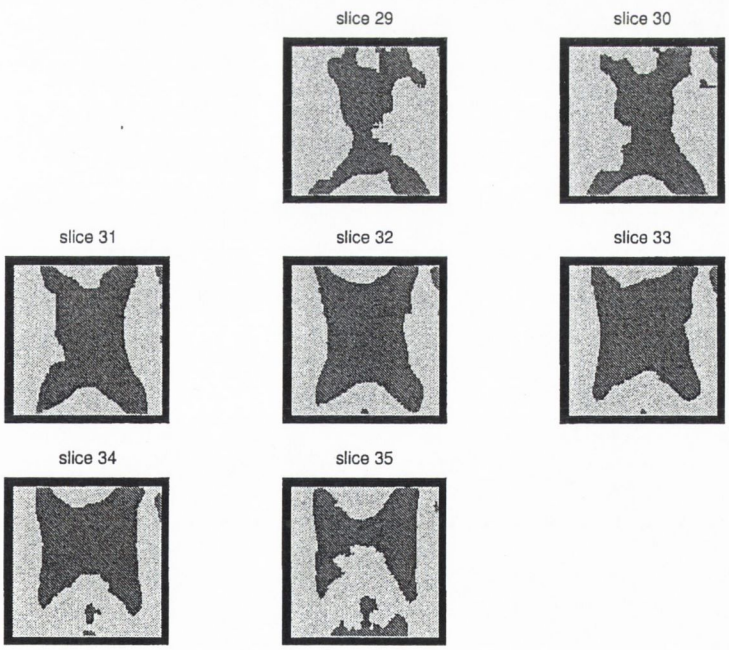

Figure E.50: 3-D RUR compensated segmentation results for the PD MRI volume (Fig. C.21), $d = 3$. Difference estimation via KLD.

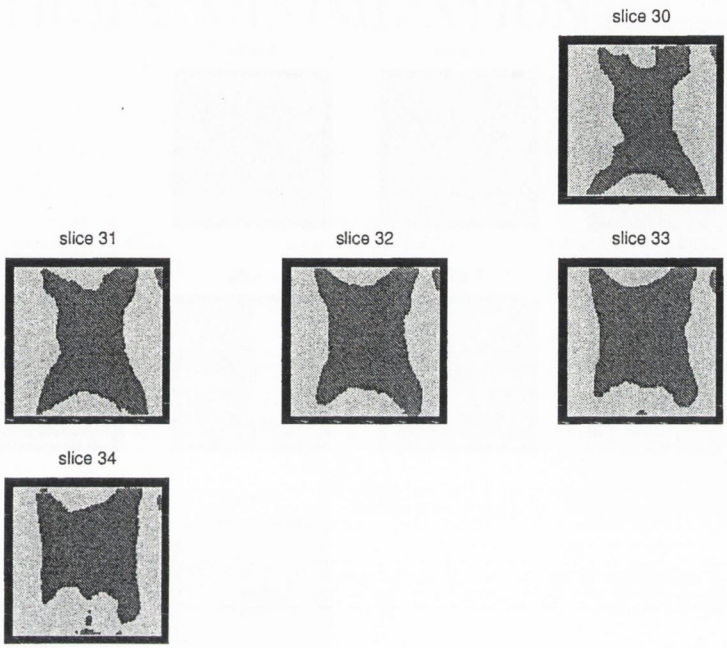

Figure E.51: 3-D RUR compensated segmentation results for the PD MRI volume (Fig. C.21), $d = 5$. Difference estimation via KLD.

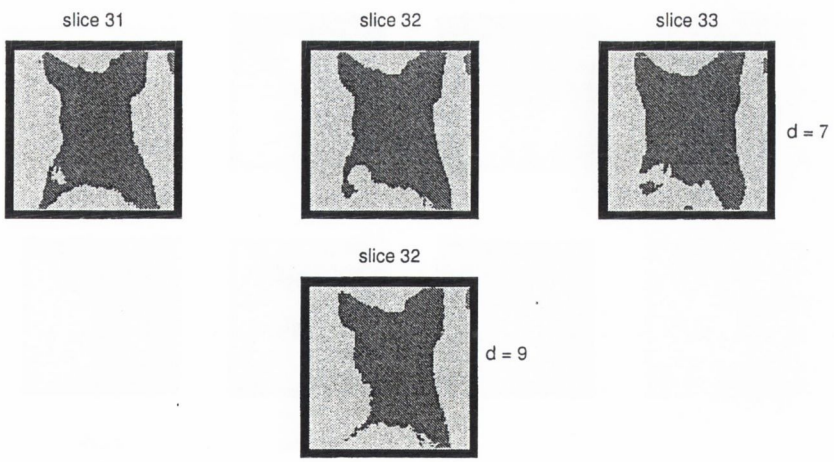

Figure E.52: 3-D RUR compensated segmentation results for the PD MRI volume (Fig. C.21), $d = 7, 9$. Difference estimation via KLD.

E.5 ENTROPY MINIMIZATION

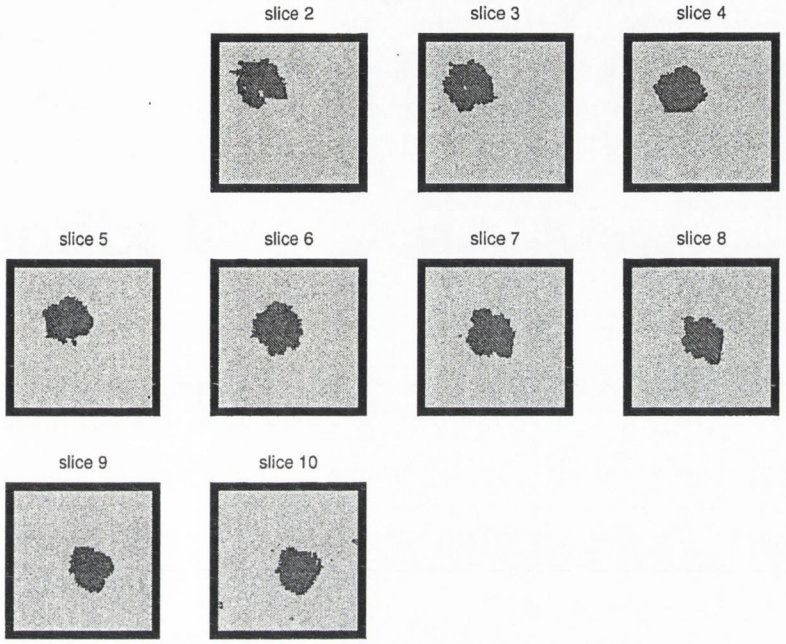

Figure E.53: 3-D RUR compensated segmentation results via entropy minimization for the 'skewed' cylinder volume (Fig. C.11).

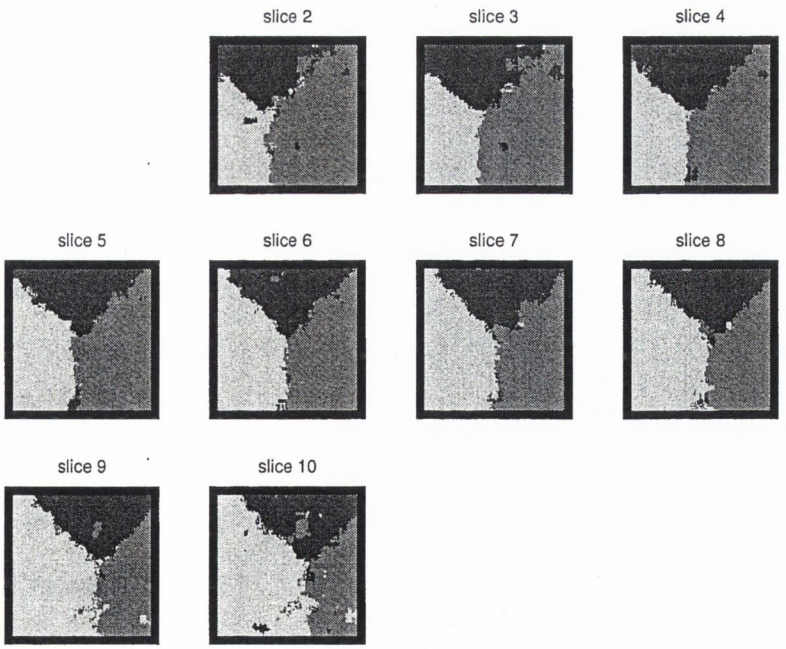

Figure E.54: 3-D RUR compensated segmentation results via entropy minimization for the three class volume (Fig. C.14).

Appendix F

PUBLICATIONS

1. [70] E. Ranguelova and A. Quinn," Analysis and Synthesis of Three-Dimensional Gaussian Markov Random Field", *Proceedings of the International Conference on Image Processing (ICIP)* , Kobe, Japan, 1999

2. [71] E. Ranguelova and A. Quinn, "Disparity-compensated Segmentation of 3-D Images", *PhD workshop Cybernetics and Informatics Eurodays*, Marianska, Czech Republic, 2000

3. [72] E. Ranguelova and A. Quinn, "Registration Preprocessing for Enhanced 3-D Segmentation", *Proceedings of the Irish Signals and Systems Conference* Maynooth, Ireland, June 2001, pp. 346–351

4. [73] E. Ranguelova and A. Quinn, "Difference Field Estimation for Enhanced 3-D Texture Segmentation.", *Proceedings of the British Machine Vision Conference*, Cardiff, UK, September 2002 pp. 373–383

5. [74] E. Ranguelova and A. Quinn, "Difference Estimation and Compensation via Entropy Minimization in 3-D Image Segmentation", *Proceedings of the Second International Workshop on Spectral Methods and Multirate Signal Processing*, Toulouse, France, September 2002, pp. 131–136