Graph-spectral Methods for Computer Vision

Antonio A. Robles-Kelly

Submitted for the degree of Philosophy Doctor

Department of Computer Science

THE UNIVERSITY of York

May 8, 2003

Abstract

This thesis describes a family of graph-spectral methods for computer vision that exploit the properties of the first eigenvector of the adjacency matrix of a weighted graph. The algorithms are applied to segmentation and grouping, shape-from-shading and graph-matching.

In Chapter 3, we cast the problem of grouping into an evidence combining setting where the number of clusters is determined by the modes of the adjacency matrix. With the number of clusters to hand, we model the grouping process using two sets of variables. These are the cluster memberships and the pairwise affinities or link-weights for the nodes of a graph. From a simple probability distribution for these parameters, we show how they may be estimated using the apparatus of the expectation-maximisation (EM) algorithm. The new method is demonstrated on the problems of line-segment grouping and gray-scale image segmentation. The method is shown to outperform a non-iterative eigenclustering method.

In Chapter 4, we present a more direct graph-spectral method for segmentation and grouping by developing an iterative maximum likelihood framework for perceptual clustering. Here, we focuss in more detail on the likelihood function that results from the Bernoulli model. Rather than using the EM algorithm to estimate an updated link-weight matrix, we show how a modal decomposition technique can be used to remove noisy link-weights. We establish the relationship between the modes of the link-weight matrix and the maxima of the log-likelihood function. The matrix analysis of the log-likelihood function results in a method that is faster to converge and which gives better defined clusters. The method is evaluated on gray-scale image segmentation, line-segment grouping and motion analysis.

In Chapter 5, we turn our attention to the problem of recovering the 3D representation of an object from its shading information. To do this, we develop a graph-spectral method for shape-from-shading. We commence by characterising the field of surface normals using a transition matrix whose elements are computed from the sectional curvature

between different image locations. With the transition matrix to hand, we use a graph seriation method to define a curvature minimising surface integration path for the purposes of height reconstruction. We extend the surface height recovery algorithm to perform surface normal adjustment. We do this by fitting quadric patches to the height data. A performance study of the resulting shape-from-shading algorithm is presented. We perform experiments on real-world imagery and compare our results to those obtained using two alternative shape-from-shading algorithms.

Finally, in Chapter 6, we show how the eigenstructure of the adjacency matrix can be used for purposes of graph edit distance computation. We make use of the graph seriation method developed in Chapter 5 to convert the adjacency matrix into a string or sequence order. We pose the problem of graph-matching as a maximum *a posteriori* probability alignment of the seriation sequences for pairs of graphs. We model this probability by making use of a compatibility co-efficient for the edge alignment of the sequences and the probability of individual node correspondences. With these ingredients, we compute the edit distance by finding the sequence of string edit operations which minimise the cost of the path traversing the edit lattice. The edit costs are defined in terms of the *a posteriori* probability of visiting a site on the lattice. We demonstrate the utility of the method for purposes of graph clustering.

Contents

List of Figures

List of Tables

List of Symbols

A	Adjacency matrix
L	Laplacian matrix
I	Identity matrix
$\mathcal{P}$	Permutation matrix
$G = (V, E)$	Graph
V	Set of nodes
E	Set of edges
d_i	Degree of the node indexed i
$\mathcal{L}(A, S)$	Log-likelihood function
Ω	Set of clusters
$s_{i\omega}$	Cluster membership variable corresponding to the node indexed i in the cluster ω
$\zeta_{i,j,\omega}^{(n)}$	*a posteriori* cluster membership probability
λ	Eigenvalue
$\mathbf{x}, \mathbf{z}, \phi$	Eigenvector
e_i	Eigenvector of the Hessian matrix
$\vec{\mathbf{N}}$	Surface normal
$\vec{\mathbf{L}}$	Unit vector in the light source direction
$\mathbf{e}$	All-ones vector
$\mathcal{S}$	Surface
Π	Plane
Γ	Path on the graph
$\Gamma_{\mathcal{S}}$	Path on the surface $\mathcal{S}$
Γ_{Π}	Path across the plane Π
$\mathcal{E}(\Gamma)$	Cost associated to the path Γ
$g(\varrho), \hat{g}(\vec{x}), \hat{g}_E(\vec{x})$	Penalty function
ν_i	Normalised brightness at the pixel-site indexed i
$f_p(x_i, y_i)$	Bi-quadric function
X, Y	Sequence of edge connected nodes
$\eta(\gamma_k \longrightarrow \gamma_{k+1})$	Elementary edit cost
$R_{k,k+1}$	Edge compatibility coefficient

Acknowledgements

First of all, I would like to dedicate this thesis to my wife Neri. Her help and support were invaluable for the successful completion of my PhD. I also owe special gratitude to my family for continuous and unconditional support of all my undertakings, scholastic and otherwise.

I would like to express my sincere appreciation and gratitude to my supervisor, Prof. Edwin Hancock, for his assistance in the preparation of this manuscript and his support and encouragement during the more than three years of postgraduate work at York. I also thank my assessor, Dr. Richard Wilson, and my external examiner, Prof. David Hogg, for their impartial assessment and constructive criticism on the material in this thesis.

I am indebted with Prof. Sudeep Sarkar, for his valuable suggestions and help during my visit to the University of South Florida (USF), and with Dr. Eraldo Ribeiro, Dr. Adrian Bors, Prof. Bin Luo, Dott. Andrea Torsello and MSc Abdullah Al-Shaher, whose advise contributed to this work. I also extend my appreciation to all my colleagues from the computer vision group at York for their assistance and support.

Finally, I wish to thank the Conacyt for sponsoring my PhD, the EPSRC for allowing me to be a Research Assistant under the MathFit framework and the Gibbs/Plessey Trust for their financial support to visit the USF.

Declaration

I declare that all the work in this thesis is solely my own except where attributed and cited to another author. Some of the material in the following chapters has been previously published. A full list of publications, ordered by chapter, is provided in Appendix D.

Chapter 1

Introduction

Spectral-graph theory is a branch of mathematics that is concerned with characterising the structural properties of graphs using the eigenvectors and eigenvalues of the adjacency matrix (or the closely related Laplacian matrix) (Chung, 1997; Varga, 2000; Cvetković et al., 1980). Recently, techniques from spectral-graph theory have been used to develop a powerful array of algorithms in computer vision. For instance, Shi and Malik (Shi and Malik, 1997) have shown how the Fiedler vector (Fiedler, 1975a; Fiedler, 1975b) (i.e. the eigenvector associated to the second smallest eigenvalue of the Lapalcian matrix) can be used to separate the foreground from the background structure in images so as to maximise the normalised graph cut. Sarkar and Boyer (Sarkar and Boyer, 1998) have shown how the leading eigenvector of the weighted proximity matrix can be used to group line segments. There have been a number of attempts to use eigenvector methods for the purposes of correspondance matching (Umeyama, 1988; Scott and Longuet-Higgins, 1991; Shapiro and Brady, 1991). Graph-spectral methods have also been used to cluster and index graph representations of shape (Luo and Hancock, 2001; Shokoufandeh et al., 1998; Horaud and Sossa, 1995).

The attractive feature of eigenvector methods is that they rely on a simple matrix representation of the problem at hand and result in algorithms that do not require complex search procedures or control structures. For instance, in the case of the normalised cut algorithm for image segmentation, the test for assigning the pixel to the background or the foreground is made using the signs of the components of the Fiedler vector. In the case of graph-matching, the permutation (or correspondance) matrix is often found by taking the outer product of the singular vectors of the adjacency matrix of the graphs being matched (Scott and Longuet-Higgins, 1991; Shapiro and Brady, 1991). Moreover, since efficient numerical algorithms (Nyström, 1928; Stewart and guang Sun, 1990; Wilkinson, 1965) exist for approximating the leading eigenvectors of matrices, graph-spectral methods offer the advantage of being simple, elegant and efficient.

Unfortunately, one of the drawbacks of graph-spectral methods is their lack of statistical robustness. Furthermore, eigenvectors are unstable under small matrix perturbations. However, of even greater importance is the fact that the pattern of eigenvalues and eigenvectors are highly unstable under structural modifications of the graph. Hence, if the number of nodes and edges changes, the spectrum of the graph may be very different. This has limited the application of graph-spectral methods in high level vision, where structural differences must be accommodated and corrected. For instance, Umeyama's method (Umeyama, 1988) fails when graphs of different sizes are being matched. A second shortcoming of existing graph-spectral methods in computer vision is that they make relatively little use of the information residing in the eigensystem.

There have also been a number of attempts to use eigenvector methods for recovering paths on graphs. For instance, Atkins *et al.* (Atkins et al., 1998) have shown how to use an eigenvector of the Laplacian matrix to sequence relational data. The method has been successfully applied to the consecutive ones problem and a number of DNA sequencing tasks. There is an obvious parallel between this method and the use of eigenvector methods to locate steady state random walks on graphs (Bremaud, 2001). Eigenvector methods have also been successfully employed for solving path-based problems such as routing (Faloutsos et al., 1999; Tangmunarunkit et al., 2002; Gkantsidis et al., 2003).

Hence, the aims in this thesis are twofold. First, we aim to show how graph-spectral methods for segmentation and grouping can be placed in a statistical setting. Second, we aim to show how graph-spectral methods can be used for solving path-based problems in computer vision.

Turning our attention to the first of these issues, we show how the grouping and segmentation problem can be cast into a maximum likelihood setting. Here we show how a simple Bernoulli mixture model can be used to represent the cluster formation process. In Chapter 3, we show how the EM algorithm can be used to locate clusters based on the eigenvalues of a weighted adjacency matrix. In Chapter 4, we take our analysis further and investigate this model in more detail. We show how the log-likelihood function for the Bernoulli model can be cast into a matrix setting. This allows us to use matrix factorisation methods to refine the structure of the weighted adjacency matrix. Our first contribution is therefore to develop graph-spectal algorithms for pairwise clustering.

As noted earlier, eigenvector methods can also be used to identify paths on graphs. Unfortunately, existing methods do not impose edge connectivity constraints on the solution. To overcome this shortcoming, we cast the problem of locating an edge connected path on a graph into an energy minimisation setting. In Chapter 5, we show how an eigenvector method can be used to find an optimal solution.

The new graph-spectral methods for pairwise clustering and path recovery are applied to a number of practical problems in computer vision. For instance, in Chapters 3 and 4 we show how the pairwise clustering methods can be used for image segmentation and line grouping. One of the contributions here is to show how the line-linking process can be modeled using an elongated polar grouping field. Turning our attention to the path-based approach, we make two practical contributions. In Chapter 5, we present a graph-spectral method for shape-from-shading. This method combines ideas from differential geometry and spectral-graph theory to perform surface reconstruction from a field of surface normals. To do this, we use the leading eigenvector of the transition matrix to find a curvature minimising path through a field of surface normals. The path is used for performing surface height recovery. In Chapter 6, we address the problem of graph-matching. We use the path-finding approach developed in Chapter 5 for mapping graphs onto strings. Once the graphs have been converted into strings, the graph edit distance may be computed using string-matching techniques. This problem also leads to a definition of the edit costs based upon the maximum *a posteriori* probability of visiting a site on the edit lattice.

Chapter 2

Literature review

In this chapter, our aim is to review the literature relevant to this thesis. The review covers graph-spectral methods and their application in computer vision. In addition, we review general literature on segmentation and grouping, shape-from-shading and graph matching. The chapter is organised in the following way.

In Section 2.2, we commence by presenting a review of the algorithms that employ graph-spectral theory to perform segmentation and grouping. Since the algorithms presented in Chapters 3 and 4 employ perceptually meaningful image properties for performing line-segment clustering, we also review the literature on perceptual grouping in computer vision and pattern recognition. Next, we proceed to examine the literature on shape-from-shading. Since graph-spectral methods have not been applied to shape-from-shading, the section reviews previous work from a general perspective. Finally, in Section 2.4, we review the relevant literature on graph edit distance to motivate the method presented in Chapter 6. Since early work on graph edit distance is based upon ideas from structural pattern recognition, we commence by surveying previous work on graph matching and conclude the section with a review of the literature on graph edit distance.

2.1 The Spectrum of a Graph

Although well established in several areas of mathematics, the application of graph-spectral techniques in computer vision and pattern recognition is a recent development. Spectral-graph theory allows graphs to be described in a compact form due to the fact that the set of leading eigenvalues and eigenvectors provide an embedding of the nodes in the graph into a *k*-dimensional subspace.

One of the main arguments leveled against graph-spectral methods is that the spectrum of a graph is not unique (i.e. it is possible to find two graphs with the same spectrum). However, the probability of finding two graphs with the same spectrum decreases rapidly as the number of edges and nodes increases. Further, graph-spectra are rotation and permutation invariant (Chung, 1997; Cvetković et al., 1980).

Biggs (Biggs, 1993) and Cvetković *et al* (Cvetković et al., 1980) define the spectrum of a graph as the eigenvalues and eigenvectors of the adjacency matrix. For a weighted graph $G = (V, E)$, the elements of the adjacency matrix A are defined to be

$$A_{i,j} = \begin{cases} w_{i,j} & \text{if } (i,j) \in E \\ 0 & \text{otherwise} \end{cases} \tag{2.1}$$

where $w_{i,j}$ is the weight of the edge between the nodes indexed i and j.

Chung (Chung, 1997), on the other hand, defines the spectrum of a graph as the set of eigenvalues and eigenvectors of the Laplacian L. The elements of the Laplacian are assigned according to the rule

$$L_{i,j} = \begin{cases} 1 & \text{if } i = j \\ -\frac{1}{\sqrt{d_i d_j}} & \text{if } (i,j) \in E \\ 0 & \text{otherwise} \end{cases} \tag{2.2}$$

where d_i denotes the degree of the node indexed i. Although the Laplacian is symmetric, it is, in general, non-positive (i.e. at least one of its elements is negative). In contrast, the adjacency matrix of an un-directed graph is non-negative and symmetric. Further, the eigenvalues of the adjacency matrix are guaranteed to be real. As a consequence, the adjacency matrix provides a more computationally efficient means for representing the graph and its spectrum is, in general, less computationally expensive to compute. Hence, several graph-spectral algorithms use the adjacency matrix in preference to the Laplacian.

Since it is non-negative, the adjacency matrix may be analysed using the Perron-Frobenius theorem (Varga, 1962; Varga, 2000; Gantmacher, 1971; Bremaud, 2001). The Perron-Frobenius theorem states that if a matrix A is symmetric, non-negative and irreducible (i.e. there exists an integer k such that $A^k > 0$), then there exists a positive eigenvalue λ_1 with algebraic and geometric multiplicity one. Furthermore, the eigenvalue λ_1 is the largest in magnitude and the coefficients of its associated eigenvector are all of the same sign. This theorem has a number of important consequences. Firstly, the convergence rates of iterative graph-spectral algorithms can be estimated using the bounds it provides for the leading eigenvalue of the adjacency matrix. Secondly, and most importantly, it proves the uniqueness and existence of the leading eigenvalue and eigenvector. The latter, is an important prerequisite for the development of the algorithms presented in this thesis.

Finally, it is important to note that, for non-regular graphs, the adjacency matrix is substochastic in nature. A matrix is called substochastic if $\sum_{j=1}^{|V \times V|} A_{i,j} \geq 1$, with strict inequality for at least one of the rows indexed $i \in \mid V \times V \mid$. It can be shown that the leading eigenvector of a symmetric, non-negative, substochastic matrix is linearly independent of the all-ones vector $\mathbf{e} = (1, 1, \ldots, 1)^T$ (Bremaud, 2001). This is an important property of the leading eigenvector of the adjacency matrix that is used in Chapters 5 and 6 for developing a spectral approach to graph seriation.

2.2 Segmentation and Grouping

Recently, graph-spectral clustering algorithms have attracted the attention of the computer vision and pattern recognition community. The main reasons for this are threefold. Firstly, graph-spectral methods rely on simple eigenvector algorithms whose stability is well understood. Secondly, they often give good results. Finally and most importantly, since graph-spectral clustering algorithms use pairwise relations between members of the set on which they operate, the rules that govern perceptual organisation can be incorporated into the clustering process. Hence, they offer the possibility of using pairwise clustering methods to develop psychophysically plausible perceptual grouping algorithms.

2.2.1 Graph-spectral Clustering

One of the first methods to employ graph-spectral analysis for the purposes of clustering was that of Scott and Longuet-Higgins (Scott and Longuet-Higgins, 1990). They used the eigenvectors of the proximity matrix to identify point-clusters. The “eigenanalysis” developed by Scott and Longuet-Higgins is closely related to the concept of “bond order” in molecular physics. The algorithm commences by constructing the matrix C whose columns are the eigenvectors of the proximity matrix H. The elements indexed i, j of the matrix H are given by $H_{i,j} = \exp(-kr_{i,j}^2)$, where k is a scaling factor and $r_{i,j}$ is the Euclidean distance between the points i and j. The matrix C is normalised to satisfy the condition $CC^T = I$. With the normalised matrix C to hand, the “bond order” matrix B is given by $B = C^T JC$, where J is a diagonal matrix whose first M diagonal elements are set to unity and the remaining elements are set to zero.

In a related development, Perona and Freeman (Perona and Freeman, 1998) proposed an algorithm based on thresholding the proximity matrix by using its first eigenvector (i.e. the eigenvector corresponding to the largest eigenvalue). They show that the proximity matrix can be factorised into two disjoint sub-matrices that represent the foreground and the background. This approach is closely related to the algorithm of Sarkar and Boyer (Sarkar and Boyer, 1998), where the leading eigenvector of the adjacency matrix is used for finding perceptual groupings of line-tokens extracted from aerial images of man-made structures. This work is a generalisation of the eigenvector-based approach to data-base organisation reported by Sengupta and Boyer (Sengupta and Boyer, 1998), where the spectrum is used as a gross structural description for the attributed graphs of a set of CAD models. This gross description is used for clustering the data-set into structurally homogeneous groups. Once the clusters are to hand, a geometric hashing strategy (Sengupta and Boyer, 1995b) is used to organise the models in a hierarchical manner.

The methods mentioned above find their mathematical foundation in the structure of the adjacency matrix. Using the properties of the adjacency matrix drawn from the theorem of Perron and Frobenius, Sarkar and Boyer (Sarkar and Boyer, 1998) have shown that the Raleigh quotient is closely related to a measure of cluster cohesiveness. Hence, if the adjacency matrix A is formed from two disjoint or quasi-disjoint blocks, the leading eigenvector can be used to characterise the principal sub-matrices of the matrix A. As a result, it is possible to factorise the matrix A into disjoint sub-matrices using the leading eigenvector of the adjacency matrix.

Shi and Malik (Shi and Malik, 1997), on the other hand, have posed the problem of pairwise clustering as that of finding the optimal partitioning of an attributed graph by minimising the graph cut. According to Shi and Malik (Shi and Malik, 1997), the cut cost is normalised by the sum of total edge connections of all nodes in the graph. The clustering problem is that of finding the partition that minimises the "normalised" cost. In their mathematical analysis, Shi and Malik pose the problem as a generalised eigensystem and arrive to an expression that is reminiscent of the Rayleigh quotient. They show that the optimal solution to the normalised cut problem is given by the second smallest eigenvector (i.e. the eigenvector corresponding to the second smallest eigenvalue) of the matrix $C = D^{-\frac{1}{2}}(D - A)D^{-\frac{1}{2}}$, where D is a degree matrix whose diagonal elements are the sum of the elements in the columns of the link weight matrix A.

Recently, there has been some work aimed at analysing the behaviour of graph-spectral clustering methods. For instance, Kannan, Vampala and Vetta (Kannan et al., 2000) have proposed a measure for assessing the quality of the results obtained when using a graph-spectral clustering algorithm. They build on the work of Azar *et al.* (Azar et al., 2000), where a unified approach for data mining (i.e. exploration and analysis of data to discover meaningful patterns and association rules) problems is presented. In doing this, they provide a theoretical justification for using spectral algorithms for tasks such as object classification and web site ranking. Adopting an empirical evaluation approach to the problem, Soundararajan and Sarkar (Soundararajan and Sarkar, 2001) have provided a performance study that assesses the output of several graph partitioning methods. The methods studied include Shi and Malik's normalised cut algorithm. Employing the quality measures developed by Borra and Sarkar (Borra and Sarkar, 1997), they are able to evaluate the clustering results by employing a composite quality criteria in conjunction with an analysis of variances (ANOVA). Another comparative study that takes into account quality corroboration is that of Weiss (Weiss, 1999). Here, the most representative graph-spectral clustering algorithms in the literature are examined using a unified mathematical framework which is based on the normalised proximity matrix.

One of the main criticisms that can be leveled at existing graph-spectral methods for pairwise clustering and grouping is that they are not statistical in nature. This has two consequences. First, it is not possible to associate the uncertainties with the cluster assignments. Second, the existing methods are not statistically robust. To overcome this shortcoming, in this thesis we intend to use graph-spectral methods in conjunction with the apparatus of statistical inference.

2.2.2 Perceptual Grouping

Perceptual grouping has its foundations in the Gestalt laws of psychology (Koffka, 1922). Gestalt psychology deals with part-whole relationships and is based upon several laws of perceptual organisation (Goldstein, 1989). Although straightforward in principle, the task of grouping a set of physically dis-connected elements in an image based on the Gestalt laws is a very difficult one to accomplish in practice. The reasons for this are twofold. Firstly, the Gestalt laws have to be embedded into the computational framework of the algorithm. Secondly, the raw image primitives must be reliably extracted from the raw image data.

The problem of perceptual grouping has been extensively studied using information theoretic and probabilistic methods. The first attempts at perceptual grouping in the literature were concerned with dot-patterns (Beck et al., 1989; Parent and Zucker, 1989). For instance, Beck, Rosenfeld and Ivry (Beck et al., 1989) have used pairwise properties such as co-linearity and proximity to locate dot-patterns that are perceived by the viewer as a straight line. Adopting a statistical approach, Parent and Zucker (Parent and Zucker, 1989) have modelled lines as a chain of trace points. They use an iterative relaxation scheme that imposes co-circularity and curvature consistency. Despite providing good results, the main argument leveled against these methods is that they do not use orientation or length.

Drawing upon psychophysics, Field (Field, 1993) has found that the relative angle and the gap to line-segment length ratio play an important role in the perceptual grouping process. Further, his work suggests that the relative angle breakpoint at which grouping ceases is close to 60 degrees. Based on evidence for the dominant role of local features over global ones, Parent and Zucker (Parent and Zucker, 1989) have used tangent and curvature information to estimate the line trace. Dickson (Dickson, 1991) has used Bayes nets to develop a hierarchical framework for splitting and merging groups of lines. Cox, Rehg and Hingorani (Cox et al., 1993) have developed a grouping method which combines evidence from the raw edge attributes delivered by the Canny edge detector. Guy and Medioni (Guy and Medioni, 1996) have used a local approach that employs a convolution mask to govern an extension field and recover a saliency map.

In a different development, Heitger and von der Heydt (Heitger and von der Heydt, 1993) have used anisotropic selective filters to recover occluded contour correspondances using the available end points and T-junctions of the pattern. Ullman (Ullman, 1976), was one of the first to use the framework of the Elastica (i.e. curves that minimise the energy over the trace connecting two points) for purposes of contour completion. He proposed a geometric approach to the problem, in which the curve joining two edge fragments is given by the pair of arcs that minimise the curvature intergral.

Adopting a Bayesian perspective, Castaño and Hutchinson (Castaño and Hutchinson, 1996) have developed a framework where straight lines are grouped if they participate in a common underlying geometric structure. The method is based on a frequentist approach over the set of partitions of the line-segments and is hence free of parameters. Castaño and Hutchinson base their work on that of Sarkar and Boyer (Sarkar and Boyer, 1993), where perceptual inference networks were developed using hierarchical Bayesian networks to handle spatial data. The main difference between the algorithm of Castaño and Hutchinson (Castaño and Hutchinson, 1996) and the one of Sarkar and Boyer (Sarkar and Boyer, 1993) resides in the fact that, in the later, conditional probabilities are computed using a voting method, while, in the former, the probabilities are computed using mathematical models of the image features.

Perceptual grouping has also been used to perform contour completion when occlusions are present. For instance, August, Siddiqi and Zucker (August et al., 1999) have developed a set of constraints for long-distance contour fragment grouping. Williams and Jacobs (Williams and Jacobs, 1997a; Williams and Jacobs, 1997b) have applied a stochastic form of the Elastica to contour fragment grouping. Taking a biologically plausible approach to the problem, Kalocsai (Kalocsai, 1997) has shown how the recognition rate of images in which the contour has been deleted can be improved by applying extension fields to the output of an array of Gabor filters.

There have been several attempts to group straight lines into splines. An example of this is the algorithm for computing extended geometric edges developed by Zucker *et. al.* (Zucker et al., 1988). Here, splines and tangent fields are used to develop an algorithm capable of connecting edge elements extracted from an image. The main advantage of the algorithm resides in the possibility of performing curvature estimation and detecting curvature singularities. This is due to the fact that the approach is capable of representing multiple orientations at the same point on the curve. Based on the approach proposed by Zucker *et. al.* (Zucker et al., 1988), Leite and Hancock (Leite and Hancock, 1997) have developed a spline fitting algorithm that employs the apparatus of the EM algorithm and least-squares minimisation techniques to compute the spline parameters. In a related development, Lüdtke (Lüdtke, 2002) has proposed an algorithm for computing spline representations from the responses of an array of Gabor filters. As an alternative to the spline representation, active contours have been used to represent complex traces obtained from a set of edge points. An example of this is the algorithm proposed by Cohen (Cohen, 2001), where the contour is modeled as a minimum cost path between two edge end-points. He builds upon the approach developed by Cohen and Kimmel (Cohen and Kimmel, 1997), where finite differences are used for computing closed contours when only one end-point is known.

Adopting a probabilistic approach, Crevier (Crevier, 1999) has recently developed an evidence combining framework for extracting chains of colinear line-segments. Amir and Lindenbaum (Amir and Lindenbaum, 1998) have a maximum likelihood method for grouping which relies on searching for the best graph partition. The method is two-step. First, grouping cues are used to construct the graph. Second, a greedy modification step is used to maximise the likelihood function. Turning our attention to information theoretic approaches, one of the best known methods is that of Hofmann and Buhmann (Hofmann and Buhmann, 1997), which uses mean-field theory to develop update equations for the pairwise cluster indicators. In related work, Gdalyahu, Weinshall and Werman (Gdalyahu et al., 1999) use a stochastic sampling method. These iterative processes have some features in common with the use of iterative relaxation style operators for edge grouping. This approach was pioneered by Shashua and Ullman (Shashua and Ullman, 1988) and later refined by Guy and Medioni (Guy and Medioni, 1996) among others. Parent and Zucker have shown how co-circularity can be used to gauge the compatibility of neighbouring edges (Parent and Zucker, 1989).

The main argument leveled against these methods draws from the fact that they rely on the assumption that a model of perception can be developed making use of association rules for the detection and characterisation of an organised structure. However, developing a computational analogue to the psychological rules that describe perception has proven an elusive task. As a result, the validity of these methods remains an open debate.

2.3 Shape-from-shading

There is important evidence that suggests that variations in the intensity of a surface represent an important cue for shape perception in the human vision system (Ramachandran, 1988; Mingolla and Todd, 1986). Moreover, psychophysical results suggest that shading can be used to compute an approximate depth map (Todd and Mingolla, 1983). Shading information may also be used to detect other shape cues. For instance, Forsyth and Zisserman (Forsyth and Zisserman, 1989) have shown how mutual shading information can be used to detect occluded contours. There is also important evidence that suggests shading is an important cue for surface interpolation. Based upon results of psychophysical experiments, Todd and Reichel (Todd and Reichel, 1989) have shown that the perceived relative depth provided by shading information is more reliable than the perceived absolute depth. The main argument leveled against the use of shading for shape recovery is that, as noted by Bülthoff and Mallot (Bülthoff and Mallot, 1988), it is not employed by the human vision system directly for building qualitative models of the depth of a surface. This can be attributed to the fact that, although shading is of central importance for recovering the 3D structure of an object, it is not a depth cue *per se*.

Despite the contradicting evidence from the psychology literature, in the computer vision community there has been considerable effort aimed at recovering the shape of an object from shading information. However, the computational recovery of shape from shading information has proved to be a non-trivial task. Further, since it is an under-constrained problem, shape-from-shading algorithms usually rely on assumptions that are not realistic when dealing with real-world imagery. This is because the intensity of a surface depends on a number of variables such as the direction of the light, the viewer and the surface normal. Hence, additional constraints have to be introduced in order to overcome the under-constrained nature of the shape-from-shading problem. One of the strongest assumptions made in order to constrain the problem is that of Lambertian surface reflectance. Under this assumption, the intensity depends solely on the angle between the surface normal and the light source direction (Horn and Brooks, 1986). As a result, only the zenith angle of the surface normal can be recovered directly from a single intensity value.

In an early attempt to perform shape-from-shading, Horn (Horn, 1975) was the first to address the problem of recovering the shape of the surface from its gray-scale image representation by making use of characteristic strips. The idea underpinning the algorithm is that, although the surface normal direction can not be determined from shading information alone, the magnitude of the surface slope can be found along an initially known direction. Hence, the algorithm defines a path on the image-lattice, usually around a singular point (i.e. a pixel-site on the image plane where the gray-scale gradient is zero), and recovers the surface height by traversing the path and incrementing the height using the measured slope. This method is prone to numerical inaccuracies and, due to its path-based nature, tends to amplify errors via back-propagation.

To overcome the noise-related problems of the algorithm developed by Horn (Horn, 1975), Horn and co-workers developed a variational approach to shape-from-shading. Ikeuchi (Ikeuchi, 1984) and Brooks and Horn (Brooks and Horn, 1985) recast the problem using the regularisation framework introduced by Poggio and Torre (Poggio et al., 1985). Regularisation is a general approach to surface fitting that seeks to minimise the error between the surface data and the fitted spline approximation under smoothness constraints. Ikeuchi (Ikeuchi, 1984) and Brooks and Horn (Brooks and Horn, 1985) note that, despite the fact that the irradiance equation provides insufficient information for recovering the surface normal directions, the solution may be found by applying a smoothness constraint during the shape-from-shading process. As a result, the surface normals can be assumed to vary smoothly over the surface and this smoothness condition can be enforced by making use of a regularisation approach. The main drawback of this approach is that the smoothing or regularisation term tends to dominate over the data-closeness term. This results in over-smoothing and lack of detail of the recovered surface.

To overcome this shortcoming, Frankot and Chellapa (Frankot and Chellappa, 1988; Frankot and Chellappa, 1990) have developed a method where the problem is solved by imposing an integrability constraint in the Fourier domain. This is a weak constraint that requires the reconstructed surface matches the image not only in terms of the local brightness but also in terms of the gray-scale gradient. Following Frankot and Chellapa, Leclerc and Bobick (Leclerc and Bobick, 1991) proposed an algorithm where the weight of the regularising term decreases as a function of iteration number. As a result, the weight is large in the early stages of the process, (i.e. when the computed surface normals are noisy and inconsistent) and small as the process approaches completion. More recently, Worthington and Hancock have developed a new framework for shape-from-shading (Worthington and Hancock, 1999) in which the image irradiance equation is treated as a hard constraint and curvature consistency constraints can be used to recover meaningful topographic surface structure (Worthington and Hancock, 1999). According to this geometric framework, the surface normals are constrained to fall on an irradiance cone whose axis is in the light source direction and whose apex angle is proportional to the inverse cosine of the measured image brightness. The azimuthal angle of the surface normal on the irradiance cone is determined using local smoothness contraints.

Although the approaches mentioned above make use of global shading information extracted from the image to recover the surface shape, there is an important body of work that aims to recover the shape of a surface using local shading information. For instance, Oliensis and Dupuis (Dupuis and Oliensis, 1992) have proposed an algorithm for which the under-constrained nature of the problem is solved using prior knowledge of the height of singular points on the surface. Their algorithm does not use regularisation and has been proven to reconstruct height information correctly from intensity data. The method propagates surface normals in the steepest downhill direction from singular points on the surface. Bichsel and Pentland (Bichsel and Pentland, 1992) have developed a fast variant of the method. Since these algorithms are not iterative, the computational cost is low. The main drawback of these local algorithms originates from the fact that the image irradiance equation is solved by employing a linear approximation to the relation between brightness and surface orientation. As a result, they are prone to errors.

A number of local shape-from-shading methods attempt to identify areas of constant curvature. For instance, Ferrie and Legarde (Ferrie and Legarde, 1990) have developed an iterative algorithm that employs curvature consistency for smoothing the recovered field of surface normals. The process uses the difference in Darboux frame orientation at adjacent image locations for improving the original estimate of the surface normal.

An alternative approach to shape-from-shading is to pose the problem as that of solving a system of differential equations. For instance, Kimmel and Bruckstein (Kimmel et al., 1995; Kimmel and Bruckstein, 1995) have shown how the apparatus of the level-set theory can be used to solve the differential equations that characterise the shape-from-shading problem. When cast into a level-set framework, the image iradiance equation can be solved as a boundary value problem. Stated succinctly, the level-set algorithm proceeds as follows. It commences initialising a height function using the iso-brightness contours on the image. Once the height function has been initialised, it is propagated over the pixel lattice depending on whether or not the normal to the contour corresponds well to the brightness of the corresponding pixel. In this manner, the surface height is recovered via the propagation of the level-curves (i.e. the level-sets) of the height function.

2.4 Graph Matching and Edit Distance

In this section, we review the literature on graph matching, edit distance and the link between them. These two research areas have produced a vast amount of literature that is not, by any means, straightforward to review. The section is organised in the following way. In Section 2.4.1, we commence by focusing our attention on the work related to inexact spectral-graph matching. In Section 2.4.2, due to its relevance to the material presented in this thesis, we survey the literature on edit distance and its relation to the maximum common subgraph.

2.4.1 Graph Matching

Graph-matching is a task of pivotal importance in high-level vision since it provides a means by which abstract pictorial descriptions can be matched to one-another. Some of the pioneering work on graph matching was undertaken by Barrow and Popplestone (Barrow and Poppplestone, 1971) and Fischler and Eischlager (Fischler and Eischlager, 1977) who addressed the task of recognising objects using high-level relational structures. These two studies provided a proof of concept that relational structures can be used for matching abstract pictorial descriptions. Since then, graph matching has attracted substantial interest in the areas of computer vision and pattern recognition.

Much of the early work on structural pattern recognition attempted to solve the problem of matching structural representations by optimising a measure of relational similarity. Some of the earliest work in the area was undertaken by Shapiro and Haralick (Shapiro and Haralick, 1985) and Fu and his co-workers (Sanfeliu and Fu, 1983; Eshera and Fu, 1984a; Eshera and Fu, 1984b) who showed how string edit distance could be extended to relational structures. The idea was to measure the similarity of graphs by counting the number of graph edit operations (i.e. node, edge insertions and deletions) required to transform a graph into another. More recently, Sebastian and Kimia (Sebastian and Kimia, 2001; Sebastian et al., 2002) have used a distance metric analogous to the string edit distance to perform object recognition from a data-set of shock graphs.

The main argument leveled against the work mentioned above is that it adopts a heuristic approach to the relational matching problem by making use of a goal-directed graph similarity measure. To overcome this problem, several authors have adopted a more general approach using ideas from information and probability theory. For instance, Wong and You (Wong and You, 1985) defined an entropic graph-distance for structural graph matching. Christmas, Kittler and Petrou (Christmas et al., 1995) have shown how a relaxation labeling approach can be employed to perform matching using pairwise attributes whose distribution is modeled by a Gaussian. In a recent series of papers, Hancock and co-workers have proposed several graph matching algorithms based on Bayesian models. For instance, Wilson and Hancock (Wilson and Hancock, 1997) have used a MAP (maximum *a posteriori*) estimation framework to accomplish purely structural graph matching. Adopting a different approach, Cross and Hancock (Cross and Hancock, 1998) developed an algorithm for matching triangulated point-sets under perspective geometry using a variant of the EM algorithm. Myers, Wilson and Hancock (Myers et al., 2000) have shown how the Levenshtein string edit distance can be used to render the method of Wilson and Hancock (Wilson and Hancock, 1997) more efficient.

Although the algorithms mentioned above make use of similarity metrics, a number of competing approaches pose the problem of structural graph matching as an optimisation one. Optimisation methods can be divided into two categories. The first of these include those methods that employ a discrete optimisation approach for locating optimal graph matches. Discrete optimisation methods include genetic search (Cross et al., 1997) and simulated annealing (Horaud and Skordas, 1989). The second approach is to employ continuous optimisation methods. For instance, Zhu, Yuille *et al.* (Yuille and Kosowsky, 1994; Yuille et al., 1994) and Gold and Rangarajan (Gold and Rangarajan, 1996) have posed the graph matching problem as that of recovering the permutation matrix. They have used mean field annealing to solve the problem.

Graph-spectral methods can also be used to recover correspondances via the estimation of a permutation matrix. Umeyama (Umeyama, 1988) has shown how a least-squares estimate of the permutation matrix may be computed by performing singular value decomposition on the adjacency matrix. The method finds the permutation matrix by taking the outer products of the eigenvectors for the two graphs being matched. Unfortunately, it is unable to cope with differences in graph-size. As a result, it cannot be used for subgraph isomorphism or for matching corrupted graphs. Scott and Loguett-Higgins (Scott and Longuet-Higgins, 1991) have shown how to recover correspondances between point-sets by performing singular value decomposition (SVD) on the between-image proximity matrix. However, the method fails when the rotation angle between the point-sets being matched becomes large. To overcome this problem, Shapiro and Brady (Shapiro and Brady, 1991) have developed an extension of the Scott and Longuett-Higgins method which finds correspondances by computing the eigenvectors of the proximity matrices of the two point-sets being matched. Adopting a purely structural approach to the recognition of line drawings, Horaud and Sossa (Horaud and Sossa, 1995) have shown how the immanental polynomials of the Laplacian matrix of the line-connectivity graph can be used for indexing a large data-base of line-drawings. In a related application, Sengupta and Boyer (Sengupta and Boyer, 1998) have developed a graph-spectral algorithm for indexing a data-base of line-drawings. The indexing algorithm proposed by Sengupta and Boyer employs the eigenvalues of a property matrix for dividing a large database of graphs into structurally homogeneous groups. Once the database has been partitioned, the clusters are organised using the information theoretic algorithm of Sengupta and Boyer (Sengupta and Boyer, 1995a). Shokoufandeh, Dickinson and Siddiqi (Shokoufandeh et al., 1998) have shown how graphs can be encoded using local topological spectra for shape recognition from large data-bases.

Although elegant due to their matrix formulation, the main limitation of these methods is their inability to cope with graphs of different sizes. As a result, they cannot be used when a large amount of structural corruption is present. In order to overcome this problem, Luo and Hancock (Luo and Hancock, 2001) have returned to the method of Umeyama (Umeyama, 1988) and have shown how it can be rendered robust to differences in graph-size and structural errors. Making use of a purely structural representation of the graph, they cast the problem into a matrix setting. The correspondance matching problem is posed as the maximum likelihood estimation of the matrix of correspondance indicators and solved using the apparatus of the EM algorithm. They also show how the correspondance indicators can be recovered using SVD. The algorithm is iterative in nature. Although the algorithm of Luo and Hancock is robust to structural errors and differences in graph-size, it is slow to converge. Furthermore, it is sensitive to initialisation.

2.4.2 Graph Edit Distance

An alternative approach to inexact graph matching is that of computing a metric that reflects the overall cost of the set of edit operations required to transform one graph into another. The aim in doing this is to extend the concept of edit distance from strings to graphs. String edit distance was first introduced by Wagner and Fisher in 1974 (Wagner and Fisher, 1974) for solving the “string-to-string” correction problem. The work develops on that of Morgan (Morgan, 1970), who showed how to correct spelling errors by making use of a set of character-string edit operations. Employing the definitions of edit operations introduced by Morgan, Wagner and Fisher proposed a general notion of "distance" between two strings. As defined by Wagner and Fisher, the edit distance is the minimum cost of all sequences of edit operations which transform one string into another. When dealing with strings, the edit operations are usually taken to be insertion, deletion and substitution. In order to model the structural distortion introduced by the string-to-string correction sequence, an edit cost is assigned to each of the edit operations. Since the selection of the edit costs affects the optimal matching of the input strings, recent work on sequence alignment has addressed the problem of estimating the costs (Rice et al., 1997). This work also shows the relationship between a particular set of cost functions and the longest common subsequence in the strings.

As noted earlier, the concept of string edit distance has been extended to graphs by Sanfeliu and Fu (Sanfeliu and Fu, 1983), Eshera and Fu (Eshera and Fu, 1984a; Eshera and Fu, 1984b) and Shapiro and Haralick (Shapiro and Haralick, 1982). Graph edit operations have also been used for performing graph reduction. For instance, Meilă and Jordan (Meilă and Jordan, 1997) have used edit operations to show how any directed graph can be converted into an undirected one whose siblings share common parents. This moralisation process is important because moral graphs can be easily reduced to junction trees, which are crucial for constructing Bayes Networks.

These methods, pose the problem of computing a metric for measuring the similarity between two graphs as that of finding a common underlying topological structure through graph edit operations. This is closely related to the problem of measuring structural affinity using the maximum common subgraph. The maximum common subgraph is an important subject in graph theory and has been used to solve relational matching problems (Horaud and Skordas, 1989; Levinson, 1992). The maximum common subgraph can be used to depict common topology. As a result, it is a powerful tool for graph matching and object recognition.

The arguments leveled against the use of the maximum common subrgraph are twofold. Firstly, it is not unique and does not depend on the index of the nodes of the graph. Secondly, and most importantly, the computation of the maximum common subgraph is an NP-complete problem. As a result, conventional algorithms for computing the maximum common subgraph have an exponential computational cost. Although these algorithms are simple to understand and code, their computational cost is high due to the fact they are usually based on maximal clique detection (Levi, 1972) or backtracking (McGregor, 1982).

To overcome this problem, practical algorithms for computing the maximum common subgraph often rely upon approximate constraints to overcome the exponential complexity problem and achieve polynomial cost. For instance, Messmer and Bunke (Messmer and Bunke, 1999) have proposed a graph-error correction algorithm that runs in polynomial time. The idea underpinning their approach is to decompose the graphs to be matched into subgraphs that can then be stored in a tree-structure. As a result of this tree indexing scheme, the subgraphs are matched only once and the computational complexity of the graph matching algorithm decreases significantly. Using this method, Shearer and Bunke (Shearer et al., 1998) have developed a video indexing technique that executes in almost real-time. However, the algorithm requires a pre-processing step in which the index tree is constructed.

The main argument leveled against graph edit distance is the lack of formal rigour when compared with string edit distance algorithms. However, recently Bunke (Bunke, 1997) has returned to the problem and has shown the relationship between graph edit distance and the size of the maximum common subgraph. The use of the edit distance for measuring graph similarity has a number of advantages. Firstly, in contrast with the maximum common subgraph, the edit distance is unique. Secondly, as shown by Bunke (Bunke, 1997), the graph edit distance is closely related to the size of the maximum common subgraph. This is not only practically useful, but also theoretically relevant since it suggests that the graph edit distance computation can be cast in a rigorously formal framework.

2.5 Conclusions

In this thesis, our main interest is to develop a family of graph-spectral algorithms for clustering, shape-from-shading and graph edit distance computation. Although elegant by virtue of their use of matrix factorisation to solve the underlying optimization problem, one of the criticisms which can be leveled at existing graph-spectral methods is that their foundations are not statistical or information theoretic in nature.

The aim in this thesis is to develop probabilistic eigenvector methods and to apply them to problems in low, intermediate and high-level vision. We commence in Chapter 3 by developing an EM algorithm which uses a Bernoulli distribution to model the cluster formation process. We show how the resulting expected log-likelihood function can be maximised using a graph-spectral method. Unfortunately, this method proves slow to converge. Hence, in Chapter 4, we develop a more efficient method which involves applying eigenvector analysis to the log-likelihood function for the Bernoulli cluster process. This allows non-overlapping clusters to be identified and noise in the link-weight matrix to be suppressed. This method proves faster to converge and more accurate than the EM algorithm.

In Chapters 5 and 6 we turn our attention to the application of eigenvector methods in intermediate and high-level vision. Here, the theoretical contribution is to show how eigenvector methods can be used to recover maximum weight paths in graphs. In Chapter 5, we use this result to develop a method for shape-from-shading. Rather than using conventional energy minimisation methods, we develop a graph-spectral method for recovering an integration path for surface height recovery. We show how the leading eigenvector of a curvature-based transition matrix can be used to locate a path through the pixel-sites such that the total curvature is minimum. Using the Perron-Frobenius theorem, we establish that the path is given by the leading eigenvector of the transition matrix.

In Chapter 6, we use the eigenvector property to convert nodes in the graph into a string order. Building upon the relation between graph matching and edit distance computation, we show how the path-finding eigenvector method developed in Chapter 5 may be used to map graphs onto strings. Once the graphs have been converted to a string order in this way, then they may be matched using classical string edit distance. We show how the required edit costs can be computed from the edge density using a simple probabilistic error model. We also show the utility of the resulting graph matching method by applying it to clustering and object recognition problems.

Chapter 3

Expectation-Maximisation Framework for Perceptual Grouping

The observation underpinning this chapter is that although considerable effort has been expended to develop algorithms for combining evidence from a raw grouping field (Williams and Jacobs, 1997b; Williams and Jacobs, 1997a), there has been little effort at casting eigenvector methods in a probabilistic setting. Moreover, the existing graph-based methods use static affinity relationships or relational abstraction as input (Boyer and Sarkar, 1999). The aim in this chapter is to develop a different approach to the problem which poses the recovery of the perceptual arrangement graph in a probabilistic evidence combining framework. We pose the problem as one of pairwise clustering which is parameterised using two sets of indicator variables. The first of these are cluster membership variables which indicate to which perceptual cluster a segmental entity belongs. The second set of variables are link weights which convey the strength of the perceptual relations between pairs of nodes in the same cluster. Our contribution is to show how to estimate both sets of indicator variables using the apparatus of the EM algorithm.

It is important to stress that although there have been some attempts at using probabilistic methods for grouping elsewhere in the literature (Castaño and Hutchinson, 1996), the method presented in this chapter has a number of unique features which distinguish it from these alternatives. First, although there have been successful attempts to develop probabilistic methods for grouping via graph partitioning, these do not use spectral information, and are instead based on search heuristics. Second, although our method relies on the iterative updating of cluster membership indicators it differs from the methods of Hofmann and Buhmann (Hofmann and Buhmann, 1997), and Shi and Malik (Shi and Malik, 1997) by virtue of the fact that the link-weight matrix is iteratively refined. Hence, when it comes to perform a maximum likelihood estimation via the EM algorithm, both, the cluster membership indicators and the link-weight matrix elements, are regarded as hidden variables.

The outline of this chapter is as follows. In Section 3.1, we develop a mixture model for the grouping problem. Section 3.2 shows how the parameters of this mixture model, namely the cluster membership probabilities and the pairwise link-weights can be estimated using the EM algorithm. We also describe how the number of components of the mixture model can be set using the eigenmodes of the link-weight matrix. We then provide two applications of the grouping algorithm. In Section 3.3 we address the problem of line-segment grouping. To commence, we describe a simple model which can be used to initialise the link-weights. We then evaluate the line-grouping method in a sensitivity study on synthetic data. We also furnish some examples on real world images. Section 3.4 presents a similar study for the problem of grey-scale image segmentation. Finally, in Section 3.5 the contributions of the work presented in this chapter are summarised.

3.1 Maximum Likelihood Framework

We pose the problem of perceptual grouping as that of finding the pairwise clusters which exist within a set of objects segmented from raw image data. These objects may be point-features such as corners, lines, curves or regions. However, in this chapter we focus on the specific problems of grey-level image segmentation and grouping line-segments. The process of pairwise clustering is somewhat different to the more familiar one of central clustering. Whereas central clustering aims to characterise cluster-membership using the cluster mean and variance, in pairwise clustering it is link-weights between nodes which are used to establish cluster membership. Although less well studied than central clustering, there has recently been renewed interest in pairwise clustering aimed at placing the method on a more principled footing using techniques such as mean-field annealing (Hofmann and Buhmann, 1997).

3.1.1 Model Ingredients

We abstract the problem in the following way. The raw perceptual entities are indexed using the set V. Our aim is to assign each node to one of a set of pairwise clusters which are indexed by the set Ω. To represent the state of organisation of the perceptual relation graph, we introduce some indicator variables. First, we introduce a cluster membership indicator which is unity if the node i belongs to the perceptual cluster $\omega \in \Omega$ and is zero otherwise, i.e.

$$s_{iw} = \begin{cases} 1 & \text{if } i \in \omega \\ 0 & \text{otherwise} \end{cases} \tag{3.1}$$

The second model ingredient is the link-weight $A_{i,j}$ between distinct pairs of nodes $(i, j) \in V \times V - \{(i, i|i \in V\}$. When the link-weights become binary in nature, they convey the following meaning

$$A_{i,j} = \begin{cases} 1 & \text{if } i \in \omega \text{ and } j \in \omega \\ 0 & \text{otherwise} \end{cases} \tag{3.2}$$

When the link-weights satisfy the above condition, then the different clusters represent disjoint subgraphs.

Our aim is to find the cluster membership variables and the link weights which partition the set of raw perceptual entities into disjoint pairwise clusters. We commence by assuming that there are putative edges between each distinct pair of nodes (i, j) belonging to the Cartesian self-product $\Phi = V \times V - \{(i, i)|i \in V\}$. Further suppose that $P(A_{i,j})$ is the probability density for the link weight appearing on the pair of nodes $(i, j) \in \Phi$.

Our aim is to locate disjoint subgraphs by updating the link weights until they are either zero or unity. Under the assumption that the link-weights on different pairs of nodes are independent of one-another, then the likelihood function for the observed arrangement of perceptual entities can be factorised over the set of putative edges as

$$P(A) = \prod_{(i,j)\in\Phi} P(A_{i,j}) \tag{3.3}$$

We are interested in partitioning the set of perceptual entities into pairwise clusters using the link weights between them. We must therefore entertain the possibility that each of the Cartesian pairs appearing under the above product, which represent putative perceptual relations, may belong to each of the pairwise clusters indexed by the set Ω. To make this uncertainty of association explicit, we construct a mixture model over the perceptual clusters and write

$$P(A_{i,j}) = \sum_{\omega\in\Omega} P(A_{i,j}|\omega)P(\omega) \tag{3.4}$$

According to this mixture model, $P(A_{i,j}|\omega)$ is the probability that the nodes i and j are connected by an edge with link weight $A_{i,j}$ which falls within the perceptual cluster indexed ω. The total probability mass associated with the cluster indexed ω is $P(\omega)$. In most of our experiments, we will assume that there are only two such sets of nodes; those that represent a foreground arrangement, and those that represent background clutter. However, for generality we proceed under the assumption that there are an arbitrary number of perceptual clusters. As a result, the probability of the observed set of perceptual entities is

$$P(A) = \prod_{(i,j)\in\Phi} \sum_{\omega\in\Omega} P(A_{i,j}|\omega)P(\omega) \tag{3.5}$$

In the next section we describe a simple model for the conditional probability density for the indicator variables given the current estimate of the link weight matrix elements i.e. $P(s_{i\omega}, s_{j\omega}|A_{i,j}$. The cluster membership indicators play the role of random variables, and the link-weights the role of distribution parameters.

3.1.2 Bernoulli Model

We now describe the generative model which underpins our pairwise clustering method. The model assumes that pairs of nodes associate to clusters as the outcome of a Bernoulli trial. The idea is that the the observed link structure of the pairwise clusters arises as the outcome of a series of Bernoulli trials. The probability that a link form between a pair of nodes if simply the link-weight between the nodes. To be more formal, let us consider the pair of nodes i and j. We are concerned with whether or not this pair of nodes both belong to the cluster indexed ω. The random variable that governs the outcome of the Bernoulli trial is the product of indicator variables $\vartheta_{i,j,\omega} = s_{i\omega}s_{j\omega}$. There of four combinations of the two indicator variables $s_{i\omega}$ and $s_{j\omega}$. For the single case when $s_{i\omega} = s_{j\omega} = 1$, then the two nodes have a pairwise association to the cluster indexed ω, and $\vartheta_{i,j\omega} = 1$. In the three cases when either $s_{i\omega} = 0$ or $s_{j\omega} = 0$, then the pair of nodes do not both associate to the cluster ω and $\vartheta_{i,j,\omega} = 0$. We model the cluster formation process as a series of independent Bernoulli trials over all pairs of nodes. According to this model, success is the event that both nodes belong to the same cluster, while failure is the event that they do not. The probability of success, i.e. the parameter of the Bernoulli trial is the link-weight $A_{i,j}$. Success is the event $\vartheta_{i,j,\omega} = 1$, and there is a single combination of cluster indicators $s_{i,\omega} = s_{j\omega} = 1$ that results in this outcome. The remaining probability mass $1 - A_{i,j}$ is assigned to the three cases which result in failure, i.e. those for which $\vartheta_{i,j,\omega} = 0$. This simple model is captured by the distribution rule

$$P(s_{i\omega}, s_{j\omega}|A_{i,j}) = \begin{cases} A_{i,j} & \text{if } s_{i\omega} = 1 \text{ and } s_{j\omega} = 1 \\ (1 - A_{i,j}) & \text{if } s_{i\omega} = 0 \text{ or } s_{j\omega} = 0 \end{cases} \tag{3.6}$$

This rule can be written in the more compact form,

$$P(s_{i\omega}, s_{j\omega}|A_{i,j}) = A_{i,j}^{s_{i\omega}s_{j\omega}}(1 - A_{i,j})^{1-s_{i\omega}s_{j\omega}} \tag{3.7}$$

We could clearly have adopted a more complicated model. by not distributing the probability uniformly among the three cases for which $\vartheta_{i,j,\omega} = 0$. Moreover, the model is developed under the assumption that the quantities $s_{i\omega}$ and $s_{j\omega}$, and hence $\vartheta_{i,j,\omega}$ are binary in nature. When we come to update the cluster indicators, then we relax this condition and the quantities no longer belong to the set $\{0, 1\}$, but instead belong to the interval $[0, 1]$.

3.2 Expectation-Maximisation

Our aim is to find the cluster-membership weights and the link-weights which maximize the likelihood function appearing in Equation (3.5). One way to locate the maximum likelihood perceptual relation graph is to update the binary cluster and edge indicators. This could be effected using a number of optimisation methods including simulated annealing and Markov Chain Monte Carlo. However, here we use the apparatus of the EM algorithm originally developed by Dempster, Laird and Rubin (Dempster et al., 1977). Our reason for doing this is that the cluster-membership variables $s_{i\omega}$ must be regarded as quantities whose distribution is governed by the link weights $A_{i,j}$. Since at the outset we know neither the associations between nodes and clusters nor the strength of the link weights within clusters, we treat these as the parameters of our model and make use of the EM algorithm to recover them.

3.2.1 Expected log-likelihood function

For the likelihood function appearing in Equation (3.5), the expected log-likelihood function is defined to be

$$Q(A^{(n+1)}|A^{(n)}) = \sum_{\omega \in \Omega} \sum_{(i,j) \in \Phi} P(s_{i\omega}, s_{j\omega}|A^{(n)}_{i,j}) \ln P(A^{(n+1)}_{i,j}|s_{i\omega}, s_{j\omega}) \qquad (3.8)$$

where $P(A^{(n+1)}_{i,j}|s_{i\omega}, s_{j\omega})$ is the probability distribution for the link-weights at iteration $n+1$ and $P(s_{i\omega}, s_{j\omega}|A^{(n)}_{i,j})$ is the a posteriori probability that the pair of nodes with link weight $A^{(n)}_{i,j}$ belong to the cluster indexed ω at iteration n of the algorithm. When the probability distribution function from Equation (3.7) is substituted, then the expected log-likelihood function becomes

$$Q(A^{(n+1)}|A^{(n)}) = \sum_{\omega\in\Omega}\sum_{(i,j)\in\Phi}\zeta^{(n)}_{i,j,\omega}\Big\{s^{(n+1)}_{i\omega}s^{(n+1)}_{j\omega}\ln A^{(n+1)}_{i,j}+ \\ (1-s^{(n+1)}_{i\omega}s^{(n+1)}_{j\omega})\ln\big[(1-A^{(n+1)}_{i,j}]\big)\Big\} \tag{3.9}$$

where we have used the shorthand $\zeta^{(n)}_{i,j,\omega} = P(s_{i\omega}, s_{j\omega}|A^{(n)}_{i,j})$ for the a posteriori cluster membership probabilities. After some algebra to collect terms, the expected log-likelihood function simplifies to

$$Q(A^{(n+1)}|A^{(n)}) = \sum_{\omega\in\Omega}\sum_{(i,j)\in\Phi}\zeta^{(n)}_{i,j,\omega}\Big\{s^{(n+1)}_{i\omega}s^{(n+1)}_{j\omega}\ln\frac{A^{(n+1)}_{i,j}}{1-A^{(n+1)}_{i,j}}+\ln(1-A^{(n+1)}_{i,j})\Big\} \tag{3.10}$$

3.2.2 Maximisation

In the maximisation step of the algorithm, we aim to recover the cluster and edge parameters $s_{i\omega}$ and $A_{i,j}$. The edge parameters are found by computing the derivatives of the expected log-likelihood function

$$\frac{\partial Q(A^{(n+1)}|A^{(n)})}{\partial A^{(n+1)}_{i,j}} = \sum_{\omega\in\Omega}\zeta^{(n)}_{i,j,\omega}\Big\{s^{(n+1)}_{i\omega}s^{(n+1)}_{j\omega}\frac{1}{A^{(n+1)}_{i,j}}(1-A^{(n+1)}_{i,j})-\frac{1}{1-A^{(n+1)}_{i,j}}\Big\} \tag{3.11}$$

and solving the equations

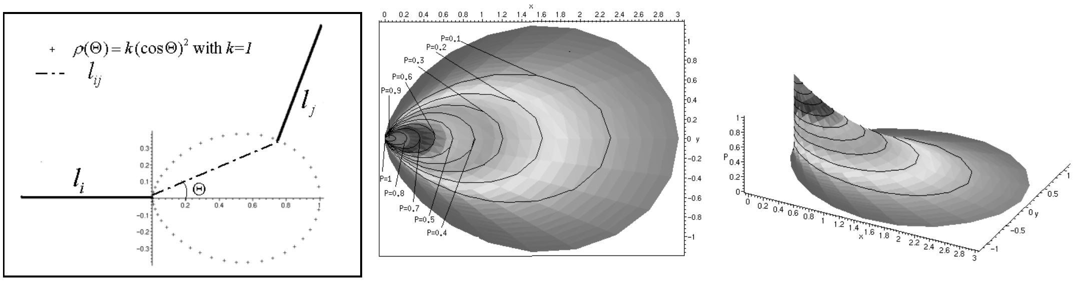
$$\frac{\partial Q(A^{(n+1)} \mid A^{(n)})}{\partial A^{(n+1)}_{i,j}} = 0 \tag{3.12}$$

As a result the updated link-weights are given by

$$A^{(n+1)}_{i,j} = \frac{\sum_{\omega\in\Omega}\zeta^{(n)}_{i,j,\omega}s^{(n+1)}_{i\omega}s^{(n+1)}_{j\omega}}{\sum_{\omega\in\Omega}\zeta^{(n)}_{i,j,\omega}} \tag{3.13}$$

where $i \in V$, $j \in V$ and $i \neq j$. In other words, the link-weight for the pair of nodes (i, j) is simply the average of the product of individual node cluster memberships over the different perceptual clusters.

To update the cluster membership variables, we compute the derivative of the expected



(a) (b) (c)

Figure 3.1: (a) Geometric meaning of the parameters used to obtain $P_{i,j}$; (b) Plot showing the level curves; (c) 3D plot showing $A_{i,j}$ on the z axis

log-likelihood function

$$\frac{\partial Q(A^{(n+1)}|A^{(n)})}{\partial s_{i\omega}^{(n+1)}} = \sum_{j \in V} \zeta_{i,j,\omega}^{(n)} s_{j\omega}^{(n+1)} \ln \frac{A_{i,j}^{(n+1)}}{1 - A_{i,j}^{(n+1)}} \tag{3.14}$$

However, the associated saddle-point equations are not tractable in closed form. Instead, we use the soft-assign ansatz of Bridle (Bridle, 1990) to update the cluster membership assignment variables. This involves exponentiating the partial derivatives of the expected log-likelihood function in the following manner

$$s_{i\omega}^{(n+1)} = \frac{\exp\left[\frac{\partial Q(A^{(n+1)}|A^{(n)})}{\partial s_{i\omega}^{(n+1)}}\right]}{\sum_{\omega \in \Omega} \exp\left[\frac{\partial Q(A^{(n+1)}|A^{(n)})}{\partial s_{i\omega}^{(n+1)}}\right]} \tag{3.15}$$

As a result, the update equation for the cluster membership indicator variables is

$$s_{i\omega}^{(n+1)} = \frac{\exp\left[\sum_{j \in V} \zeta_{i,j,\omega}^{(n)} s_{j\omega}^{(n)} \ln \frac{A_{i,j}^{(n+1)}}{1-A_{i,j}^{(n+1)}}\right]}{\sum_{\omega \in \Omega} \exp\left[\sum_{j \in V} \zeta_{i,j,\omega}^{(n)} s_{j\omega}^{(n)} \ln \frac{A_{i,j}^{(n+1)}}{1-A_{i,j}^{(n+1)}}\right]} = \frac{\prod_{j \in V} \left\{\frac{A_{i,j}^{(n+1)}}{1-A_{i,j}^{(n+1)}}\right\}^{\zeta_{i,j,\omega}^{(n)} s_{j\omega}^{(n)}}}{\sum_{\omega \in \Omega} \prod_{j \in V} \left\{\frac{A_{i,j}^{(n+1)}}{1-A_{i,j}^{(n+1)}}\right\}^{\zeta_{i,j,\omega}^{(n)} s_{j\omega}^{(n)}}} \tag{3.16}$$

where $i \in V$ and $\omega \in \Omega$.

3.2.3 Expectation

The *a posteriori* probabilities are updated in the expectation step of the algorithm. The current estimates of the parameters $s_{i\omega}^{(n)}$ and $A_{i,j}^{(n)}$ are used to compute the probability densities $P(A_{i,j}^{(n)}|s_{i\omega}, s_{j\omega})$ and the *a posteriori* probabilities are updated using the formula

$$P(\omega|A_{i,j}^{(n)}) = \frac{P(A_{i,j}^{(n)}|s_{i\omega}, s_{j\omega})\alpha^{(n)}(\omega)}{\sum_{\omega\in\Omega} P(A_{i,j}^{(n)}|s_{i\omega}, s_{j\omega})\alpha^{(n)}(\omega)} \tag{3.17}$$

where $\alpha^{(n)}(\omega)$ is the available estimate of the class-prior $P(\omega)$. This is computed using the formula

$$\alpha^{(n)}(\omega) = \frac{1}{|V|^2} \sum_{(i,j)\in\Phi} P(s_{i\omega}, s_{j\omega}|A_{i,j}^{(n)}) \tag{3.18}$$

Upon substituting for the probability density from Equation (3.7), the updated *a posteriori* probabilities are given by

$$P(s_{i\omega}, s_{j\omega}|A_{i,j}^{(n+1)}) = \zeta_{i,j,\omega}^{(n+1)} = \frac{{A_{i,j}^{(n)}}^{s_{i\omega}^{(n)} s_{j\omega}^{(n)}} (1 - A_{i,j}^{(n)})^{1-s_{i\omega}^{(n)} s_{j\omega}^{(n)}} \alpha^{(n)}(\omega)}{\sum_{(i,j)\in\Phi} {A_{i,j}^{(n)}}^{s_{i\omega}^{(n)} s_{j\omega}^{(n)}} (1 - A_{i,j}^{(n)})^{1-s_{i\omega}^{(n)} s_{j\omega}^{(n)}} \alpha^{(n)}(\omega)} \tag{3.19}$$

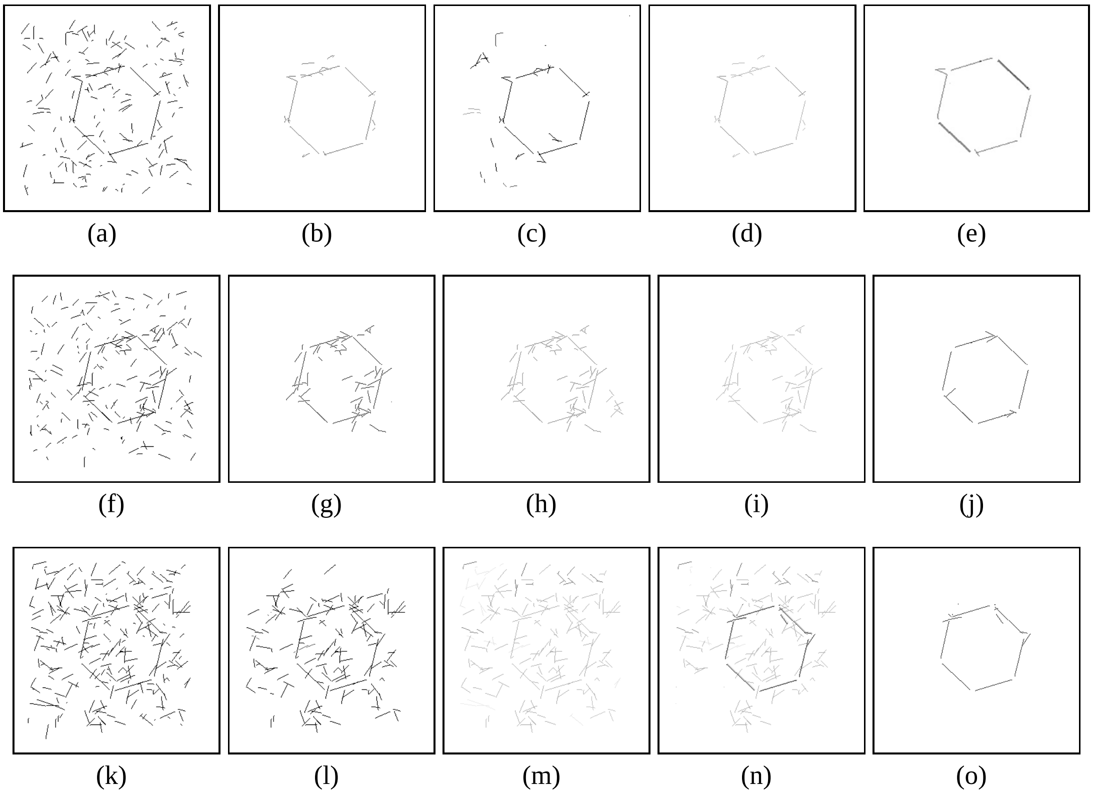
3.2.4 Selecting the Number of Modes

One of the practical difficulties associated with using a mixture model is that of selecting the number of components. There are several well established ways of addressing this problem. These frequently involve removing or splitting components so as to optimise a measure of model-order complexity (Ueda et al., 2000; Vlassis and Likas, 1999; Figueiredo et al., 1999). There are several possible choices of utility measures including the minimum description length (Risanen, 1989) and the Akaike information criterion (Akaike, 1969; Akaike, 1978; Sakamoto et al., 1988).

However, here we take a different route and use the modal structure of the initial affinity matrix to set the number of mixing components. Here we use a result due to Sarkar and Boyer (Sarkar and Boyer, 1998) who have shown how matrix factorisation methods can be used to locate the set of edges which partition the nodes into distinct perceptual clusters. One way of viewing this is as the search for the permutation matrix which reorders the elements of A into non-overlapping blocks. Unfortunately, when the elements of the matrix A are not binary in nature, then this is not a straightforward task. However, Sarkar and Boyer (Sarkar and Boyer, 1998) have shown how the same-sign eigenvectors of the matrix of link-weights can be used to assign nodes to perceptual clusters. Using the Rayleigh-Ritz theorem, they observe that the scalar quantity $\underline{x}^t A^{(0)} \underline{x}$, where $A^{(0)}$ is the initial weighted adjacency matrix, is maximised when $\underline{x}$ is the leading eigenvector of A. Moreover, each of the subdominant eigenvectors corresponds to a disjoint perceptual cluster. They confine their attention to the same-sign positive eigenvectors (i.e. those whose corresponding eigenvalues are real and positive, and whose components are either all positive or are all negative in sign). If a component of a positive eigenvector is non-zero, then the corresponding node belongs to the perceptual cluster associated with the associated eigenmodes of the weighted adjacency matrix. The eigenvalues $\lambda_1, \lambda_2 ....$ of $A^{(0)}$ are the solutions of the equation $|A^{(0)} - \lambda I| = 0$ where I is the $|V| \times |V|$ identity matrix. The corresponding eigenvectors $\underline{x}_{\lambda_1}, \underline{x}_{\lambda_2}, ...$ are found by solving the equation $A^{(0)} \underline{x}_{\lambda_i} = \lambda_i \underline{x}_{\lambda_i}$. Let the set of positive same-sign eigenvectors be represented by $\Omega = \{\omega | \lambda_\omega > 0 \wedge [(\underline{x}_\omega^*(i) > 0 \forall i) \vee \underline{x}_\omega^*(i) < 0 \forall i])\}$. Since the positive eigenvectors are orthogonal, this means that there is only one value of ω for which $\underline{x}_\omega^*(i) \neq 0$. In other words, each node i is associated with a unique cluster. We denote the set of nodes assigned to the cluster with modal index ω as $V_\omega = \{i | \underline{x}_\omega^*(i) \neq 0\}$. Hence each positive same-sign eigenvector is associated with a distinct mixing component. We use the eigenvectors of the initial affinity matrix to initialise the cluster membership variables. This is done using the magnitudes of the modal co-efficients and we set

$$s_{iw}^{(0)} = \frac{|\underline{x}_\omega^*(i)|}{\sum_{i \in V_\omega} |\underline{x}_\omega^*(i)|} \tag{3.20}$$

At this point it is worth pausing to review the way in which Sarkar and Boyer (Sarkar and Boyer, 1998) use the eigenvectors of the adjacency matrix to assign nodes to clusters. In an attempt to render the clustering method robust to noise, they allow up to 5% of


(a) (b) (c) (d) (e)

(f) (g) (h) (i) (j)

(k) (l) (m) (n) (o)

Figure 3.2: Results on a sequence of noisy synthetic images.

the components of the eigenvectors to flip sign. To be more formal let $U_\omega^+ = \{i | \lambda_\omega > 0 \wedge \mathbf{x}_\omega^*(i) > 0\}$ be the set of components of $\mathbf{x}_\omega$ that are of positive sign and let $U_\omega^- = \{i | \lambda_\omega > 0 \wedge \mathbf{x}_\omega^*(i) < 0\}$ be the set of components that have negative sign. The set of eigenmodes used by Sarkar and Boyer to represent the set of perceptual clusters is $\Omega_{SB} = \{\omega | \frac{|U_\omega^+|}{|V|} > t \vee \frac{|U_\omega^-|}{|V|} > t\}$. In other words the positive eigenvectors are acceptable as modal clusters provided that the fraction of same-sign components exceeds the threshold t. In practice 5% of the components are allowed to flip sign, i.e. $t = 0.95$. One of the problems introduced by this procedure is that by allowing flipping, nodes may be ambiguously assigned to more than one cluster.

We have recently developed an alterntive way of rendering the non-iterative eigendecomposition method robust to noise (Robles Kelly and Hancock, 2000). This involves thresholding the link-weight matrix. Prior to performing eigendecomposition, we set insignificant elements, i.e. those which are smaller than 0.001, to zero.

In our experiments, we will compare the results delivered by the EM algorithm with those obtained with both sign-flipping and the thresholded adjacency matrix.

3.2.5 Algorithm description

Finally, to summarise, the iterative steps of the algorithm are as follows:

1.- *Initialisation*: Compute the eigenvectors of the initial current link-weight matrix $A^{(0)}$. Each same-sign eigenvector whose eigenvalue is positive is used to seed a different component of the mixture model.

2.- *Expectation*: Compute the updated cluster-membership variables using the E-step.

3.- *Maximisation*: Update the link-weights using the M-step to compute the updated link weight matrix $A^{(n)}$.

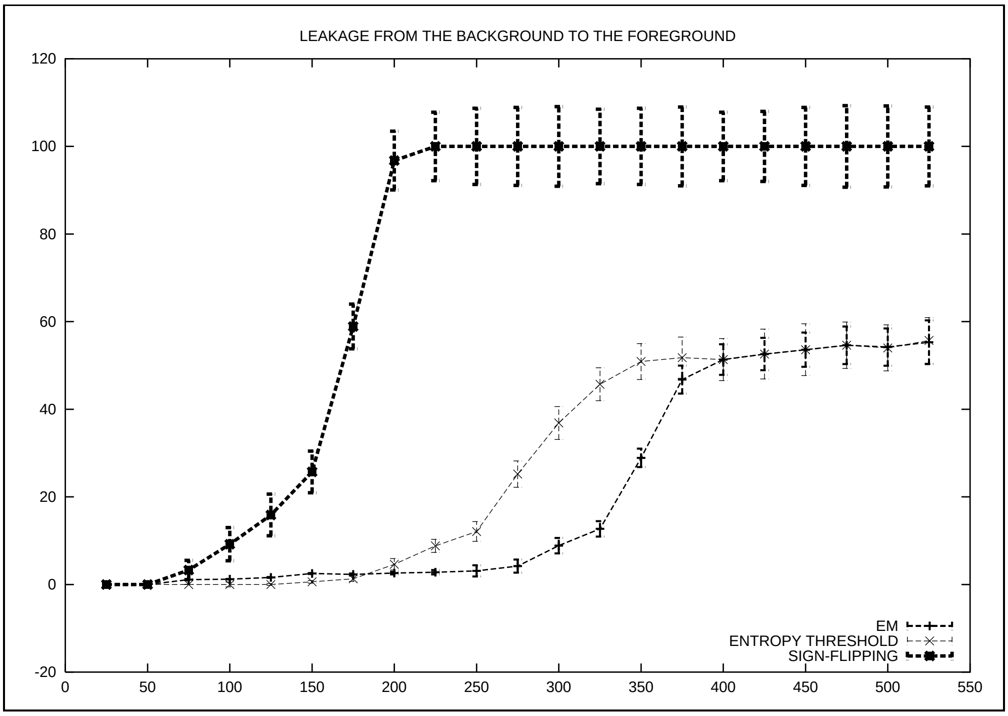
4.- Repeat steps (2) and (3) until convergence is reached.

3.3 Line Grouping

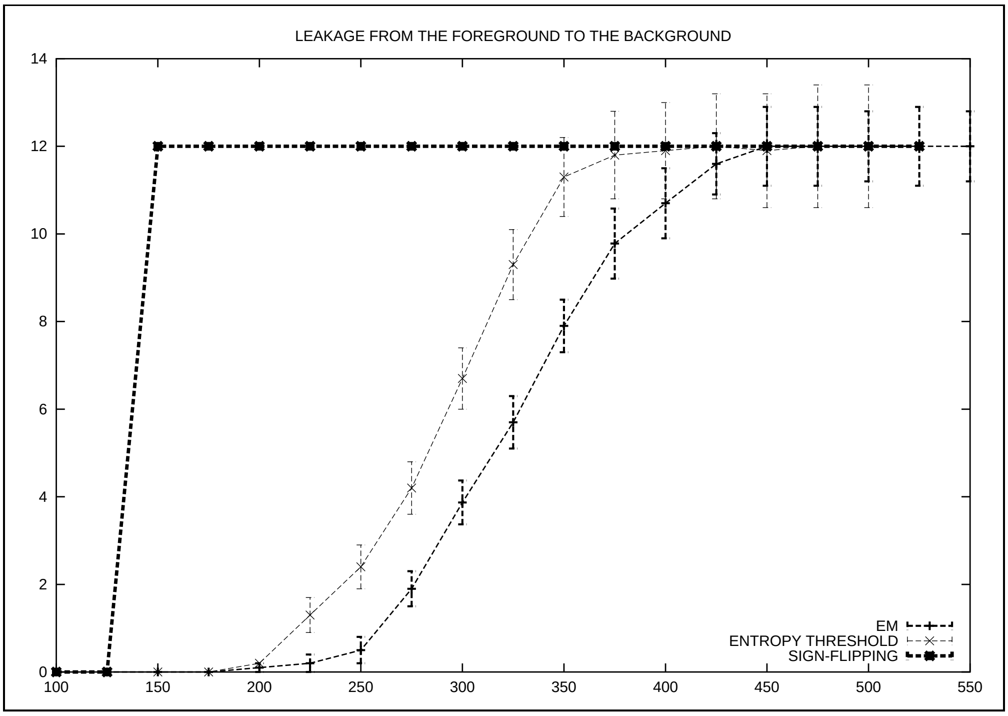
In this section we provide the first example application of our new clustering method. This involves the grouping or linking of line-segments.

3.3.1 Initial Line-Grouping Field

We are interested in locating groups of line-segments that exhibit strong geometric affinity to one-another. In this section we provide details of a probabilistic linking field that can be used to gauge geometric affinity. To be more formal suppose we have a set of line-segments $\Lambda = \{\Lambda_i; i = 1, ..., n\}$. Consider two lines Λ_i and Λ_j drawn from this set. Their respective lengths are l_i and l_j. Our model of the linking process commences by constructing the line $\Lambda_{i,j}$ which connects the closest pair of endpoints for the two lines. The geometry of this connecting line is represented using its length $\rho_{i,j}$ and the polar



(a) Number of nodes leaked into the foreground versus number of distractors added for sign-flipping (top curve), thresholded link-weight matrix (middle curve) and EM (lower curve).



(b) Number of nodes leaked into the foreground versus number of distractors added for sign-flipping (top curve), thresholded link-weight matrix (middle curve) and EM (lower curve).

Figure 3.3: Comparison between the EM algorithm and the method of Sarkar and Boyer

angle $\theta_{i,j}$ of the line $\Lambda_{i,j}$ with respect to the base-line Λ_i. We measure the overall scale of the arrangement of lines using the length of the shorter line $\hat{\rho}_{i,j} = \min[l_i, l_j]$.

The relative length of the gap between the two line-segments is represented in a scale-invariant manner using the dimensionless quantity $\xi_{i,j} = \frac{\rho_{i,j}}{\hat{\rho}_{i,j}}$.

Following Heitger and Von der Heydt (Heitger and von der Heydt, 1993) we model the linking process using an elongated polar grouping field. To establish the degree of geometric affinity between the lines we interpolate the end-points of the two lines using the polar lemniscate

$$\xi_{i,j} = k \cos^2 \theta_{i,j} \tag{3.21}$$

The value of the constant k is used to measure the degree of affinity between the two lines. For each linking line, we compute the value of the constant k which allows the polar locus to pass through the pair of endpoints. The value of this constant is

$$k = \frac{\rho_{i,j}}{\hat{\rho}_{i,j} \cos^2 \theta_{i,j}} \tag{3.22}$$

The geometry of the lines and their relationship to the interpolating polar lemniscate is illustrated in Figure 3.1a. It is important to note that the polar angle is defined over the interval $\theta_{i,j} \in (0, \pi]$ and is rotation invariant.

We use the parameter k to model the linking probability for the pair of line-segments. When the lemniscate envelope is large, i.e. k is large, then the grouping probability is small. On the other hand, when the envelope is compact, then the grouping probability is large. To model this behaviour, we assign the linking probability using the exponential distribution

$$A^{(0)}_{i,j} = \exp[-\delta k] \tag{3.23}$$

where δ is a constant whose best value has been found empirically to be unity. As a result, the linking probability is large when either the relative separation of the endpoints is small i.e. $\rho_{i,j} << \hat{\rho}_{i,j}$ or the polar angle is close to zero or π, i.e. the two lines are colinear or parallel. The linking probability is small when either the relative separation of the endpoints is large i.e. $\rho_{i,j} >> \hat{\rho}_{i,j}$ or the polar angle is close to $\frac{\pi}{2}$, i.e. the two lines are perpendicular. In Figures 3.1 b and c we show a plot of the linking probability as a function of $\frac{\rho_{i,j}}{\hat{\rho}_{i,j}}$ and $\theta_{i,j}$.

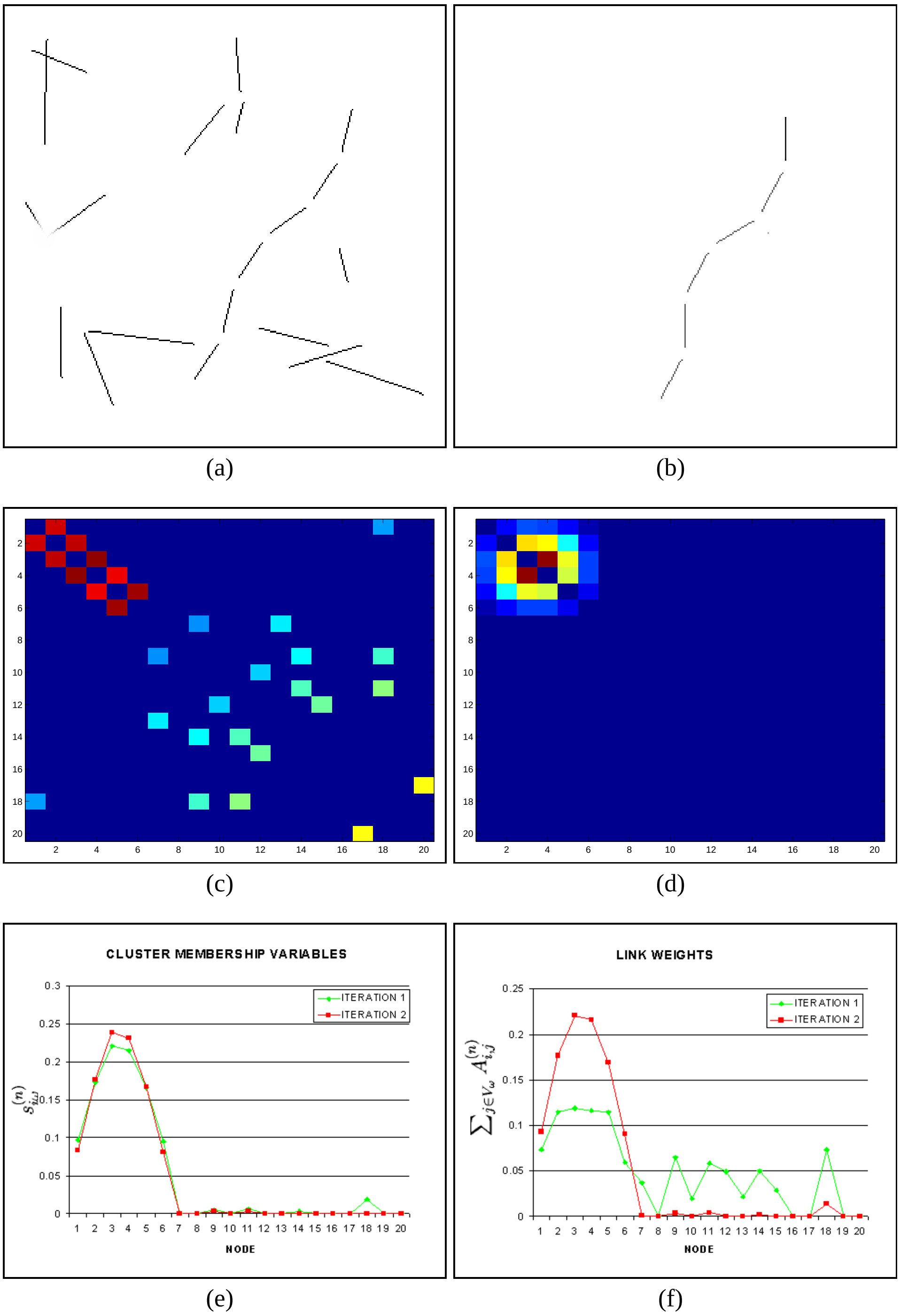


Figure 3.4: Input pattern (a); output pattern (b); initial adjacency matrix (c); final adjacency matrix (d); distribution of the cluster membership variables (e); distribution of the link weight mass (f).

3.3.2 Experiments

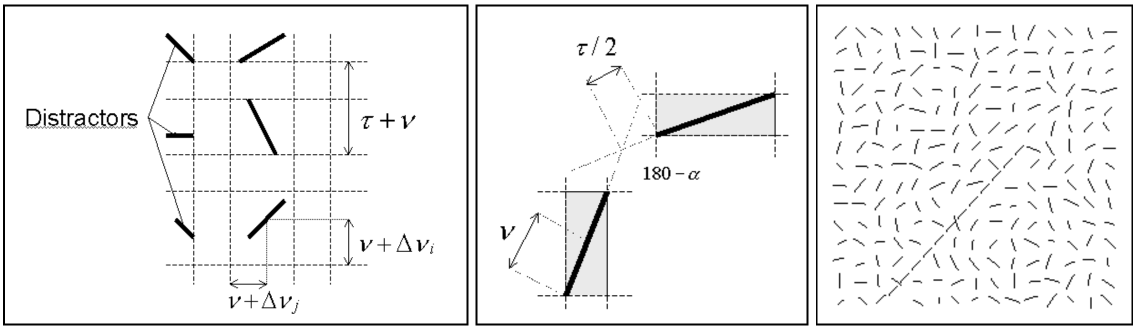
In this Section, we provide some experiments to illustrate the utility of our new perceptual grouping method. There are two aspects to this study. We commence by providing some examples for synthetic images. Here we investigate the sensitivity of the method to clutter and compare it with an eigen-decomposition method. The second aspect of our study focuses on real world images with known ground-truth.

Synthetic Images

The first sequence of synthetic images is shown in Figure 3.2. Here the foreground structure is an approximately circular arrangement of line-segments. In the first column of Figure 3.2 we show the arrangement of lines with increasing amounts of added clutter. In the subsequent columns we show the results of grouping. The second column shows the pattern of foreground line segments extracted by applying the eigendecomposition method described in (Robles Kelly and Hancock, 2000) to the grouping field already detailed in Section 3.3.1. The third, fourth and fifth columns show the results obtained with the first, second and third iterations of the EM algorithm. Here the line segments are coded according to the value of the cluster membership weights $s_{i\omega_f}$ where ω_f is the foreground cluster label. As the EM algorithm iterates, then so the foreground cluster weights tend to unity. In each case, the foreground cluster located by the EM algorithm contains less noise contamination than the result delivered by non-iterative eigendecomposition. Moreover, none of the line segments leaks into the background.

Next, we perform a more quantitative study on the robustness to noise of the algorithm. To this end, we have repeated the experiments described above for a sequence of synthetic images in which the density of distractors increases. For each image in turn we have computed the number of distractors merged with the foreground pattern and the number of foreground line-segments which leak into the background. Figures 3.3 a and b respectively show the fraction of nodes merged with the foreground and the fraction of nodes which leak into the background as a function of the number of distractors. Each plot contains three curves. In both cases, the dotted (leftmost) curve is the result of applying the Sarkar and Boyer (Sarkar and Boyer, 1998) method to the raw link-weight matrix. This method assigns lines to clusters based on the same-sign eigenvectors as described in Section 3.3. However, the method attempts to accommodate noise by allowing 5% of the components to flip sign. The dashed (middle) curve is the result of applying the Sarkar and Boyer method to the thresholded link-weight matrix (Robles Kelly and Hancock, 2000). Here we set the elements of the matrix to zero if they fall below an entropy threshold. Finally, the dot-dashed (rightmost) curve is the result obtained on covergence of the EM algorithm (in practice 3-4 iterations). In both cases, the shoulder response curve for the EM algorithm occurs at a significantly higher error rate than those for the eigen-decomposition method of Sarkar and Boyer when applied with either sign-flipping tolerance or thresholding of the link-weight matrix.

At this point, we turn our attention to the link weights $A_{i,j}$. To illustrate how the link weights behave under iteration of the EM algorithm, we have taken an arrangement of line-segments with distractors of variable length and orientation. In Figure 3.4a we show the pattern of line-segments used as input to the EM algorithm. Figure 3.4b shows the foreground cluster obtained after 2 iterations of the algorithm. Figures 3.4c and 3.4d show the initial and final link weight matrices. Here the foreground line-segments occupy the top-lefthand corner of the matrix. Initially, there are a significant number of non-zero



(a) (b) (c)

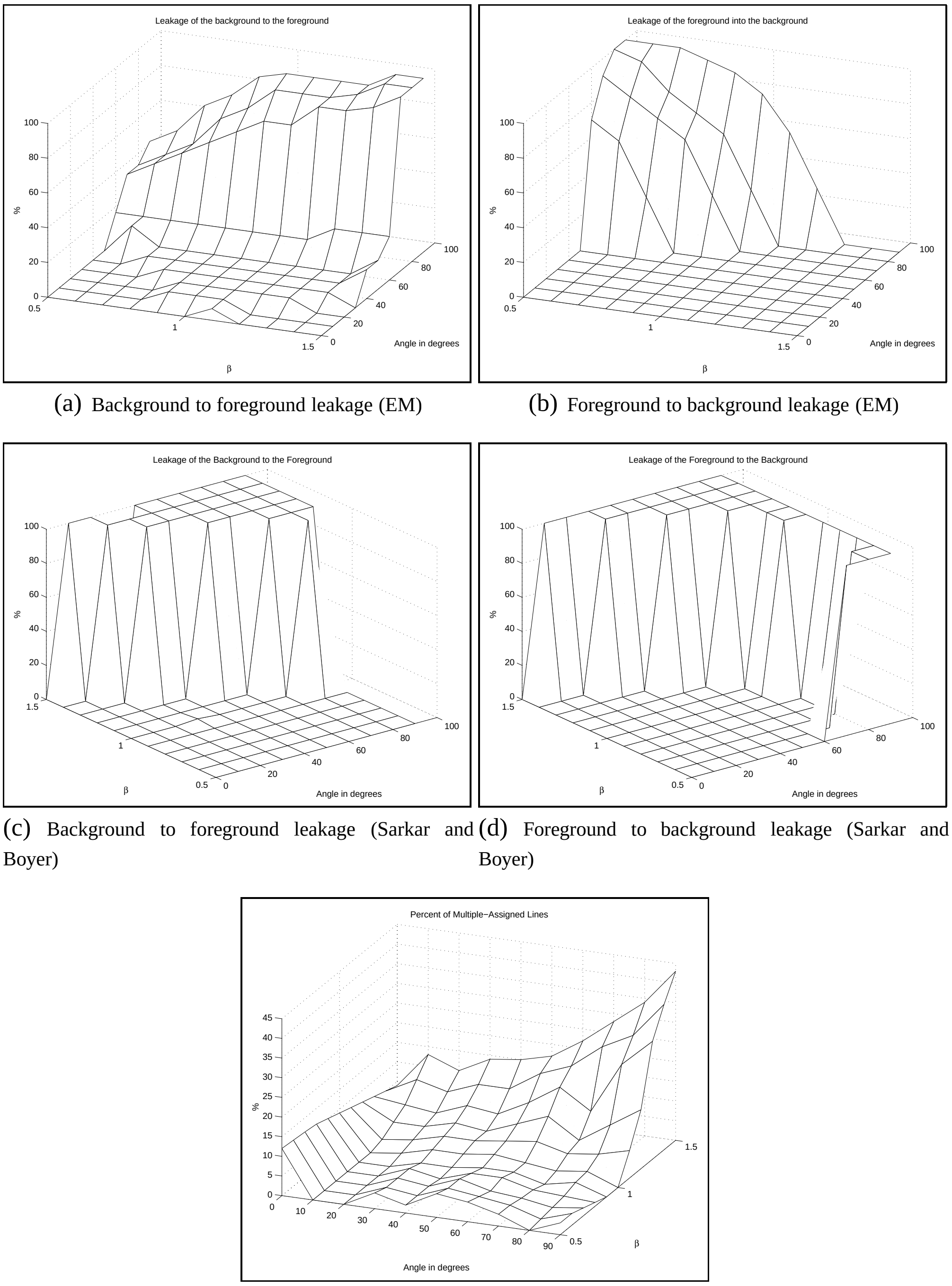
Figure 3.5: Geometric meaning of the parameters α, υ and τ in the line pattern generation (a) and in the distractors generation (b);(c) Sample test pattern ($\beta = 1.5$ and $\alpha = 0$ deg.).

entries outside this corner of the matrix. After the EM algorithm has converged, these entries are zero.

In Figures 3.4e and 3.4f, we illustrate how the distributions of cluster membership variables and link-weights change with iteration number. In the two plots the green curve shows the initial distribution and the red curve shows the distribution after one iteration of the algorithm. It is clear that in both cases the algorithm sharpens the distributions. However, re-distribution of link weight is more strongly marked.

Finally, we provide some sensitivity analysis for our grouping algorithm. These experiments have been performed using the distractor grid proposed by Field (Field, 1993). The algorithm has been tested by embedding a line pattern consisting of 12 tokens in a background of 220 distractors. The distractor pattern was generated using a grid of 15×15 pixel squares each of which contained a randomly orientated line. The centre point of the distractor lines were positioned as illustrated in Figure 3.5a, where $\Delta \upsilon_{i,j}$ is a random positional jitter in the centre-point position which is sampled from the interval $[0, \tau - \upsilon - 1]$. Here 2τ is the distance from the closest endpoint of the line to the intersection point, 2υ is the line-length and α is the complement of the opening angle. The parameter υ was fixed to 12 pixels and τ was varied between 8 and 24 pixels. The 12 line foreground pattern was created using the geometric parameters τ, α and υ as shown in Figure 3.5b. If an endpoint of a distractor line fell inside a square of the grid occupied by the centre-point of another distractor, then it was deleted. Figure 3.5c shows an example pattern generated in this way.

The results of the performance analysis are shown in Figure 3.6. Here we have investigated the effect of varying the opening angle α of the foreground pattern and increasing the gap length τ. We measure the gap-length using the ratio $\beta = \frac{\upsilon}{\tau}$. Each point in the plot is averaged over 4 randomly generated distractor patterns. Figures 3.6 a and b show the results obtained with the EM algorithm. In Figure 3.6a we show the fraction of background contamination of the foreground as a function of the gap-length ratio β and the line opening angle α. The contamination is greatest for large opening angles and large gap-lengths (i.e. small values of β). The contamination becomes severe once the open-



(a) Background to foreground leakage (EM)

(b) Foreground to background leakage (EM)

(c) Background to foreground leakage (Sarkar and Boyer)

(d) Foreground to background leakage (Sarkar and Boyer)

(e) Fraction of multiply assigned lines (Sarkar and Boyer)

Figure 3.6: Sensitivity study results

ing angle exceeds 60 degrees. In Figure 3.6b, we show the fraction of foreground lines that leak into the background. The leakage is again shown as a function of the opening angle and gap-length ratio. Here the method again fails for large opening angles and large gap-lengths. However, the onset of algorithm failure is now more abrupt. Of par-

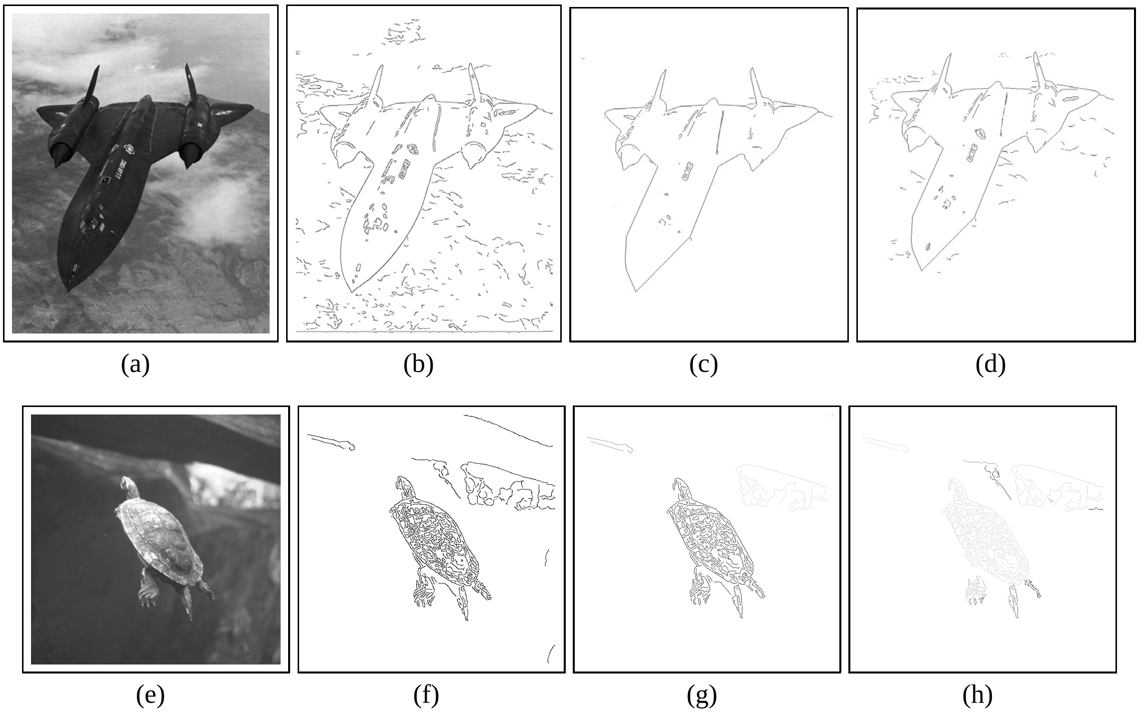


Figure 3.7: Real world images in (a) and (e);(b) and (f) results for the canny edge detector;(c),(g) results for the EM algorithm; (d),(h) results for the entropy threshold approach.

ticular importance is the fact that the onset of grouping errors is dependent only on the opening angle and not upon the gap-length ratio β. These results are in accord with the psychophysical results obtained by Field (Field, 1993).

Figures 3.6 c and d repeat this analysis for the algorithm of Sarkar and Boyer (Sarkar and Boyer, 1998) with sign-flipping tolerance. Now the angle at which there is onset of grouping errors is dependent on the gap-length ratio β. In particular, for small gap-lengths then the onset angle for grouping errors becomes small. In other words, the method is more fragile than the EM algorithm. Finally, it is important to stress that the Sarkar and Boyer method allows line-segments to be assigned to more than one eigencluster since it tolerates sign-flips. In Figure 3.6 e we plot the fraction of ambiguously assigned line-segments as a function of the opening angle and the gap-length ratio β. The fraction of ambiguously assigned lines is greatest for large gap-lengths and large angles.

Real-World Images

Finally, we present results on real-world images. We have used the airplane and turtle images shown in Figure 3.7a and e. The edges shown in Figure 3.7b and f have been extracted from the raw images using the Canny edge-detector. Straight-line segments have been extracted using the method of Yin (Peng-Yeng, 1998). The resulting clusters obtained with the EM algorithm can be seen in Figure 3.7c and g. Here the EM algorithm typically converged in 3-4 iterations. For comparison, in Figure 3.7d and h we show the results obtained with the eigendecomposition method of Sarkar and Boyer (Sarkar and Boyer, 1998). In both cases the grouping obtained by the EM algorithm is cleaner and contains less spurious clutter.

3.4 Grey Scale Image Segmentation

The second application of our new pairwise clustering method involves segmenting grey-scale images into regions. To compute the initial affinity matrix, we use the difference in grey-scale values at different pixel sites. Suppose that ν_i is the grey-scale value at the pixel indexed i and ν_j is the grey-scale value at the pixel indexed j. The corresponding entry in the affinity matrix is

$$A^{(0)}_{i,j} = \exp[-k(\nu_i - \nu_j)^2] \tag{3.24}$$

where k is a heuristically set constant. If we are segmenting an $R \times C$ image of R rows and C columns, then the affinity matrix is of dimensions $RC \times RC$. Hence, the affinity matrix is rather large (i.e. typically $RC \times RC > 1600$). This initial characterisation of the affinity matrix is similar to that used by Shi and Malik (Shi and Malik, 1997) in their normalised-cut method of image segmentation It is important to stress that this is an extremely simplistic method that does not, for instance, include information concerning the spatial proximity of the pixels.

By applying the clustering method to this initial affinity matrix we iteratively segment

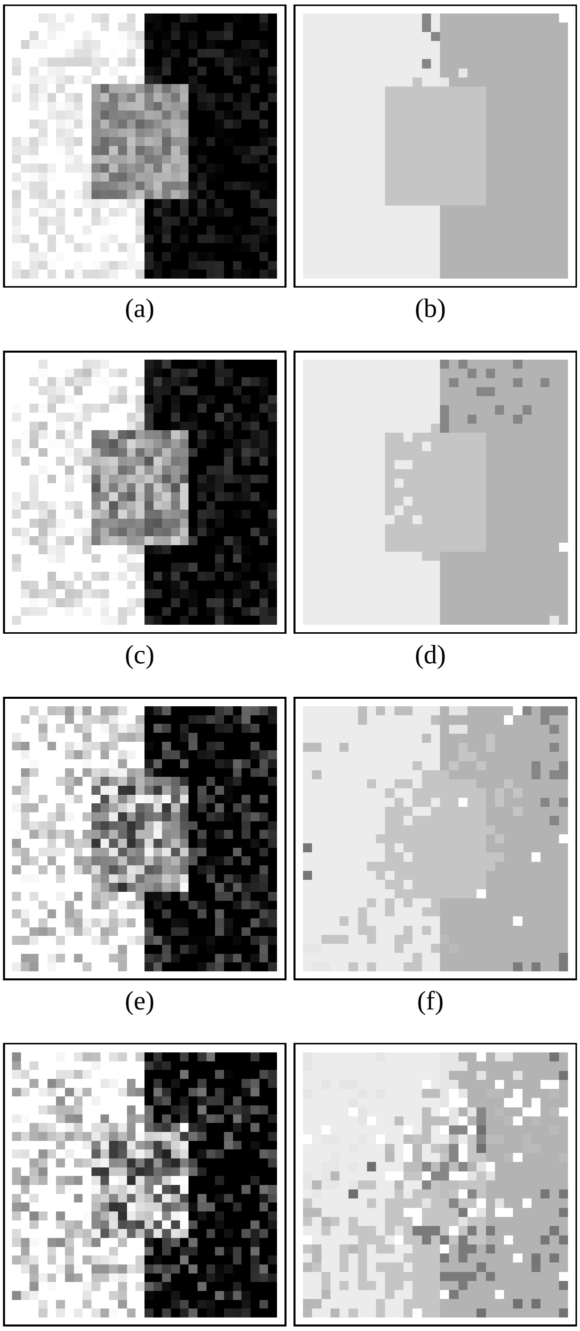


Figure 3.8: Results obtained on synthetic images using the EM algorithm.

the image into regions. Each pairwise cluster corresponds to a distinct region.

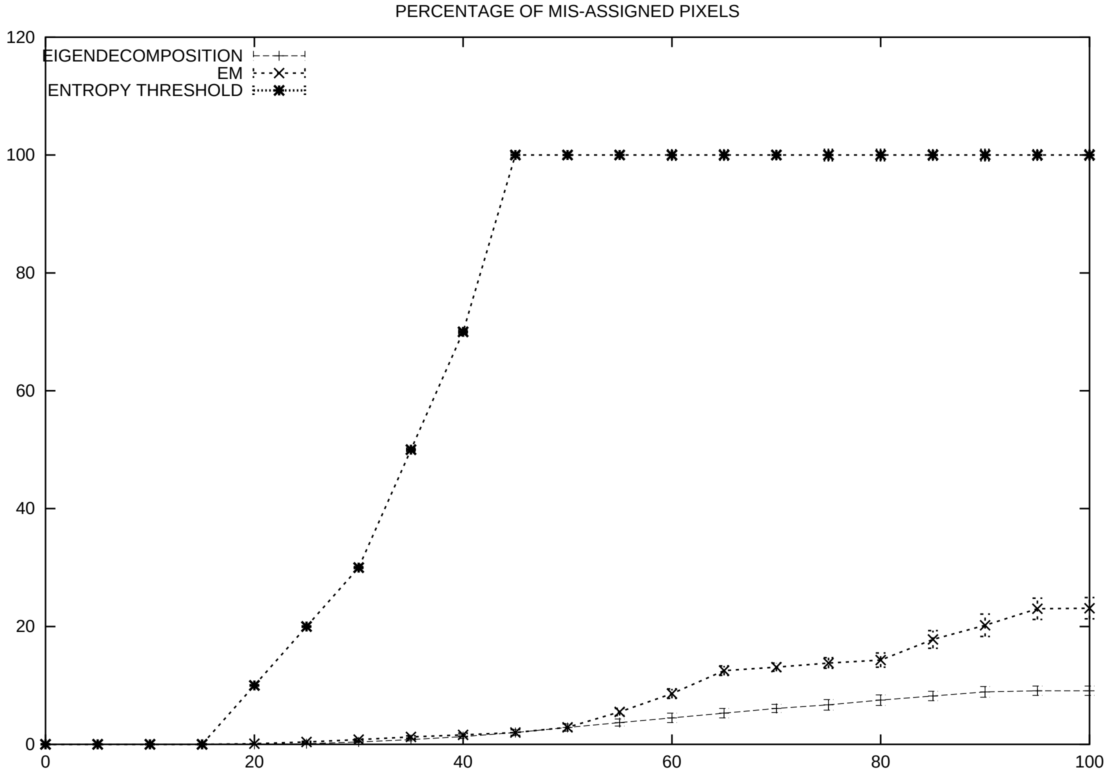


Figure 3.9: Number of segmentation errors versus noise standard deviation with sign-flipping (top curve), thresholded link-weight matrix (middle curve) and EM (lower curve).

3.4.1 Experiments

We have conducted experiments on both synthetic and real images. We commence with some examples on synthetic images aimed at establishing some of the properties of the resulting image segmentation method.

We commence by considering the effect of added random noise. In Figure 3.8 a), c), e), and g) we show a sequence of images in which we have added Gaussian noise of zero mean and known standard deviation to the grey-scale values in an image containing three rectangular regions. In the sequence, each pair of images represents the original noise corrupted image (on the left) and the resulting segmentation (on the right) as the standard deviation of the noise in increased. For the different image pairs, standard deviation of the added Gaussian noise is 35%, 50%, 65% and 95% of the grey-scale difference between the regions. The final segmentations are obtained with an average of 2.3 iterations. The

Table 3.1: Details of the results shown in fig.3.10.

Image	Iterations	No. of Clusters	k
(a)	1.4	7	3
(c)	1.8	5	3
(e)	4.5	16	3
(g)	4.8	7	3

final segmentations contain 3, 3, 4 and 4 clusters respectively. The method begins to fail once the noise exceeds 60%. It is also worth noting that the region boundaries and corners are well reconstructed.

Figure 3.9 offers a more quantitative evaluation of the segmentation capabilities of the method. Here we compute the fraction of mislabelled pixels in the segmented images as a function of the standard deviation of the added Gaussian noise. The plot shows three performance curves obtained with a) the eigendecomposition algorithm method of Sarkar and Boyer with sign-flipping (leftmost curve), b) the Sarkar and Boyer method applied to the thresholded link-weight matrix (middle curve) and c) the EM method (rightmost curve). The two variants of the Sarkar and Boyer method fail abruptly at low noise-levels. However, thresholding of the link-weight matrix is slightly more robust than using sign-flip tolerance. The EM algorithm performs much better.

To conclude this section, in Figure 3.10 we provide some example segmentations on real-world images. In the left-hand column we show the original image, while the right-hand column shows the final segmentation. Table 3.1 lists the value of the parameter k, the number of clusters and the average number of iterations taken to obtain each segmentation. On the whole the results are quite promising. The segmentations capture the main region structure of the images. Moreover, they are not unduly disturbed by brightness variations or texture. It should be stressed that these results are presented to illustrate the scope offered by our new clustering algorithm and not to make any claims concerning its utility as a tool for image segmentation. To do so would require comparison and sensitivity analysis well beyond the scope of this chapter.

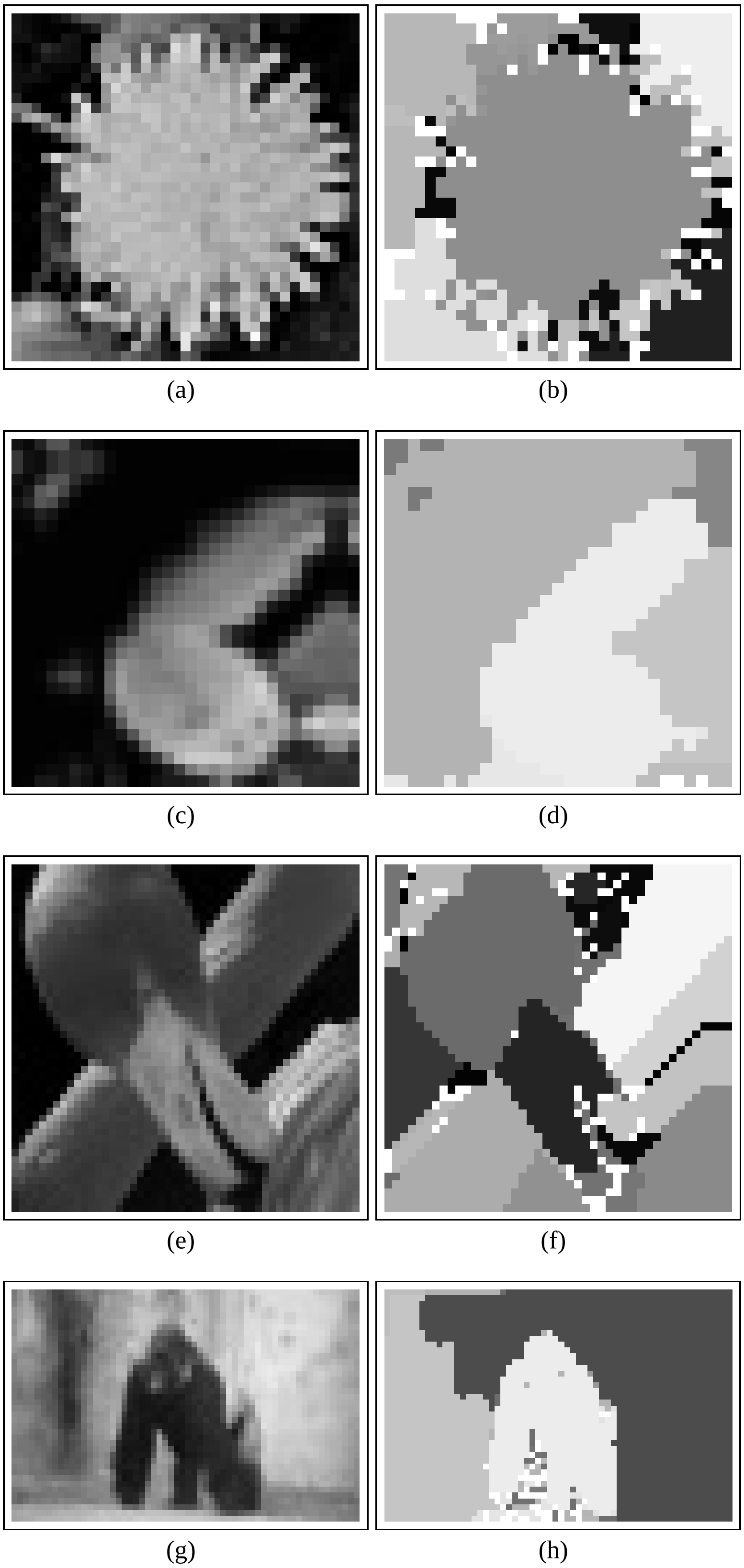


Figure 3.10: Segmentations of grey-scale images.

3.5 Conclusions

In this chapter, we have presented a new perceptual clustering algorithm which uses the EM algorithm to estimate link-weights and cluster membership probabilities. The method is based on a mixture model over pairwise clusters. The cluster membership probabilities are modeled using a Bernoulli distribution for the link-weights. We apply the method to the problems of region segmentation and of line-segment grouping. In the case of line-segment grouping, the method appears robust to severe levels of background clutter. Although more preliminary, the results obtained for region segmentation are promising and underline the flexibility of the new method.

At this point, it is important to note that, once the initial link-weight matrix has been computed, the dual update steps which constitute the algorithm are decoupled from the raw image data. The two steps are aimed at improving the structure of the link-weight matrix and the pairwise clusters that can be extracted from it. One way of viewing this process is that of applying a kind of relaxation process which smooths the link-weight matrix by re-enforcing adjacent elements within a block.

There are two drawbacks of the method presented in this chapter. First, the method is relatively slow to converge. Second, the clusters are overlapped, which results in poor performance. In the next chapter, we attempt to overcome this problems by developing a more direct segmentation method based on a maximum likelihood approach.

Chapter 4

Maximum Likelihood Framework for Segmentation and Grouping

Despite being effective, the method developed in the previous chapter is prone to errors due to cluster overlap and is relatively slow to converge. Here, we aim to overcome these problems by using direct matrix decomposition methods to develop a maximum likelihood framework for segmentation and grouping. The method is reminiscent of the EM algorithm presented in Chapter 3. However, here we adopt a more sophisticated and direct way of updating the link-weight matrix.

Our starting point is to take our analysis further and investigate in more detail the Bernoulli model developed in the previous chapter. We commence by showing how the log-likelihood function for this model can be cast in a matrix setting. This allows us to use matrix factorisation methods to refine the structure of the updated link-weight matrix in order to remove noisy link weights. To do this, we decompose the updated link-weight matrix into components that originate from the different clusters. For each cluster-component of the link-weight matrix, we compute the leading eigenvector. We compute a revised link weight matrix by taking the sum of the outer products of the leading eigenvectors of the independent cluster link-weight matrices.

4.1 Maximum Likelihood Framework

In this section, we develop our maximum likelihood framework. The method makes use of the same formalism as introduced in Chapter 3, and also utilises the Bernoulli distribution to model the cluster formation process.

4.1.1 Joint Likelihood Function

Our grouping process aims to estimate the cluster-membership indicators S and to obtain an improved estimate of the link-weight matrix A. It will be convenient to work with a matrix representation of the cluster membership indicators. Hence, we introduce a vector of indicator variables for the cluster indexed ω

$$\underline{s}_\omega = (s_{1\omega}, s_{2\omega}, ...)^T$$

The vectors are used as the columns of the $|V| \times |\Omega|$ cluster membership matrix

$$S = (\underline{s}_1 | \underline{s}_2 | ..... | s_{|\Omega|}) \tag{4.1}$$

whose rows are indexed by the set of nodes and whose columns are indexed by the set of clusters.

We pose these problems of estimating the link-weight matrix A and the cluster-membership indicators S in terms of the conditional likelihood $P(S|A)$. The problem of recovering indicator variables is one of maximum a posteriori probability estimation of S given the current link-weight matrix A. Here the link weight matrix plays the role of fixed data. The re-estimation of A is posed as maximum likelihood estimation, with the cluster membership indicators playing the role of fixed data.

To develop the two update steps, we turn our attention to the conditional likelihood function

$$P(S|A) = P(\underline{s}_1, \underline{s}_2, ..., \underline{s}_{|\Omega|} | A) \tag{4.2}$$

To simplify the likelihood function, we make a number of independence assumptions. We commence by applying the chain rule of conditional probability to rewrite the likelihood function as a product of conditional probabilities

$$\begin{aligned} P(S|A) &= P(\underline{s}_1|\underline{s}_2, ..., \underline{s}_{|\Omega|}, A) \\ &\times P(\underline{s}_2|\underline{s}_3, ..., \underline{s}_{|\Omega|}, A) \\ &\times P(\underline{s}_i|\underline{s}_{i+1}, ..., \underline{s}_{|\Omega|}, A) \\ &\times ..... \\ &\times P(\underline{s}_{|\Omega|}|A) \end{aligned} \tag{4.3}$$

To simplify this product, we assume that vectors of class indicators are conditionally independent of one-another given the matrix of link-weights. Hence

$$P(\underline{s}_i|\underline{s}_{i+1}, ..., \underline{s}_{|\Omega|}, A) = P(\underline{s}_i|A)$$

Using this simplification we can write the conditional likelihood as a product over the cluster indices

$$P(S|A) = \prod_{\omega \in \Omega} P(\underline{s}_\omega|A) \tag{4.4}$$

Next we apply the definition of conditional probability to re-write terms under the product in the following manner

$$P(S|A) = \prod_{\omega \in \Omega} \frac{P(A|\underline{s}_\omega)P(\underline{s}_\omega)}{P(A)} \tag{4.5}$$

To further develop the expression for the likelihood we turn our attention to the conditional probability for the link-weight matrix given the vector of cluster indicators for the cluster ω, i.e. $P(A|\underline{s}_\omega)$. Again applying the chain rule of conditional probability, we can perform the following factorisation over the non-diagonal elements of the link-weight matrix

$$P(A|\underline{s}_\omega) = \prod_{(i,j) \in \Phi} P(A_{i,j}|A_{k,l}, k > i, l > j, \underline{s}_\omega) \tag{4.6}$$

where $\Phi = V \times V - \{(i,i)|i \in V\}$ is the set of non-diagonal elements of A. To simplify the factorisation, we assume that the element $A_{i,j}$ is conditionally dependant only on the cluster indicators for the nodes indexed i and j. Hence, we can write

$$P(A_{i,j}|A_{k,l}, k > i, l > j, \underline{s}_\omega) = P(A_{i,j}|s_{i\omega}, s_{j\omega})$$

Under this simplification

$$P(A|\underline{s}_\omega) = \prod_{(i,j)\in\Phi} P(A_{i,j}|s_{i\omega}, s_{j\omega}) \tag{4.7}$$

Substituting this expression into that for the joint-likelihood we have that

$$P(S|A) = \prod_{\omega\in\Omega} \left\{ \frac{P(\underline{s}_\omega)}{P(A)} \prod_{(i,j)\in\Phi} \left[\frac{P(s_{i\omega}, s_{j\omega}|A_{i,j})P(A_{i,j})}{P(s_{i\omega}, s_{j\omega})} \right] \right\} \tag{4.8}$$

As stated earlier, we aim to recover revised estimates of both the link-weight matrix and the cluster indicators. These estimates are realised using dual interleaved update operations. The recovery of the revised link weight matrix is posed as the maximum likelihood parameter estimation problem

$$A^* = \arg\max_A P(S|A) \tag{4.9}$$

The recovery of the cluster membership indicators, on the other hand, is posed as the maximum a posteriori probability estimation problem

$$S^* = \arg\max_S P(S|A) \tag{4.10}$$

Since, we are interested in the joint dependence of the link-weight matrix A and the cluster membership indicators S, we turn our attention to the maximisation of the quantity

$$\mathcal{L}(A,S) = \sum_{\omega\in\Omega} \sum_{(i,j)\in\Phi} \ln P(s_{i\omega}, s_{j\omega}|A_{i,j}) \tag{4.11}$$

4.1.2 Bernoulli Model

The generative model underpinning the clustering process is identical to that used in Chapter 3. We assume that the cluster membership process is governed by a series of independent Bernoulli trails and write

$$P(s_{i\omega}, s_{j\omega}|A_{i,j}) = A_{i,j}^{s_{i\omega}s_{j\omega}}(1 - A_{i,j})^{1-s_{i\omega}s_{j\omega}} \tag{4.12}$$

After substituting this distribution into the log-likelihood function, we find that

$$\mathcal{L}(A,S) = \sum_{\omega\in\Omega}\sum_{(i,j)\in\Phi}\left\{s_{i\omega}s_{j\omega}\ln A_{ij} + (1 - s_{i\omega}s_{j\omega})\ln(1 - A_{i,j})\right\} \tag{4.13}$$

Performing algebra to collect terms, the log-likelihood function simplifies to

$$\mathcal{L}(A,S) = \sum_{\omega\in\Omega}\sum_{(i,j)\in\Phi}\left\{s_{i\omega}s_{j\omega}\ln\frac{A_{ij}}{1 - A_{ij}} + \ln(1 - A_{i,j})\right\} \tag{4.14}$$

The structure of the log-likelihood function deserves further comment. The first term, which depends on the cluster membership indicators, is closely related to the association measure for the configuration of clusters. Classically, the association of the cluster indexed ω is defined to be $Assoc(\omega) = \sum_{i\in V}\sum_{j\in V} s_{i\omega}s_{j\omega}A_{i,j} = \vec{s}_\omega^T A\vec{s}_\omega$. Hence, our log-likelihood function is the sum of the invidual cluster associations for the logarithmically transformed link weight matrix. Our method hence does not take into account the cut-measure between clusters. The cut between the clusters indexed ω_a and ω_b is defined to be $Cut(\omega_a, \omega_b) = \sum_{i\in\omega_a}\sum_{j\in\omega_b} s_{i\omega_a}s_{j\omega_b}A_{i,j}$. There is considerable debate in the spectral clustering literature concerning the choice of utility measure. Maximising the association is widely thought to work well with compact well separated clusters, and is at the heart of the Sarkar and Boyer method. Minimising the cut, on the other hand can remove outliers from an otherwise well defined cluster. To strike a ballance between these two behaviours has lead to the development of more sophisticated measures such as the normalised cut (Shi and Malik, 1997). As evidenced by the recent paper of Kannan, Vempala and Vetta

(Kannan et al., 2000), the debate on the optmimal choice of utility measure continues. However, as noted by Weiss (Weiss, 1999) the postprocessing of the spectral representation can play a pivotal role in determining the quality of the clusters recoverable. This is not suprising. For instance, techniques such as relaxation labelling have proved to be very effective in improving the results of otherwise limited initial labellings. However, the aim here is to commence from a principal starting point.

There are a number of additional points concerning the structure of the log-likelihood function. First, when the cluster membership indicators are initialised using the components of the same-sign eigenvectors of A (as described later), it gauges only the within-cluster structure of the link-weight matrix. There are no contributions from between cluster links. The second feature is that the structure of the log-likelihood function is reminiscent of that underpinning the expectation-maximisation algorithm. The reason for this is that the product of cluster-membership variables $s_{i\omega}s_{j\omega}$ plays a role similar to that of the *a posteriori* measurement probability in the EM algorithm, and weight contributions of the link-weights to the likelihood function.

4.1.3 Maximising the likelihood function

In this section we focus on how the log-likelihood function can be maximised with respect to the link-weights and the cluster membership variables. This is a three-step process. We commence by showing how the maximum likelihood link-weight matrix can be located by taking the outer-product of the vectors of cluster membership indicators. Second, we show how to remove noise from the link-weight matrix using a process which we refer to as modal sharpening. This involves decomposing the link-weight matrix into components corresponding to the same-sign eigenvectors. For each component or cluster there is an individual link-weight matrix. For each such matrix, we compute the leading eigenvector. The modal sharpening process involves reconstructing the overall link-weight matrix by summing the outer products of the leading eigenvectors of the cluster link-weight matrices, The third, and final component of the update process is to update the link-weights.

This is done by applying a naive mean field method to the likelihood function.

Updating the Link-Weight Matrix

Our aim is to explore how the log-likelihood function can be maximised with respect to the link-weights and the cluster membership indicators. In this section, we turn our attention to the first of these. To do this we compute the derivatives of the expected log-likelihood function with respect to the elements of the link-weight matrix

$$\frac{\partial \mathcal{L}}{\partial A_{ij}} = \sum_{\omega \in \Omega} \left\{ s_{i\omega} s_{j\omega} \frac{1}{A_{ij}(1 - A_{ij})} - \frac{1}{1 - A_{ij}} \right\} \tag{4.15}$$

Followin the procedure already adopted in Chapter 3, the matrix of updated link weights $\hat{A}_\omega$ may be found by setting the derivatives to zero and solving the equation

$$\frac{\partial \mathcal{L}}{\partial A_{ij}} = 0 \tag{4.16}$$

The derivative vanishes when the following condition is satisfied

$$A_{i,j} \sum_{\omega \in \Omega} s_{i\omega} s_{j\omega} = |\Omega| \tag{4.17}$$

Re-ordering terms, the solution is

$$\hat{A}_{ij} = \frac{1}{|\Omega|} \sum_{\omega \in \Omega} s_{i\omega} s_{j\omega} \tag{4.18}$$

In other words, the link-weight for the pair of nodes (i, j) is simply the average of the product of individual node cluster memberships over the different perceptual clusters. This differs from the result presented in Chapter 3 only by virtue of the fact that there is no logner any weighting. We can make the structure of the updated link-weight matrix clearer if we make use of the vector of membership variables for the cluster indexed ω, i.e. $\underline{s}_\omega = (s_{1\omega}, s_{2\omega}, ...)^T$. With this notation the updated link-weight matrix is

$$\hat{A} = \frac{1}{|\Omega|} \sum_{\omega \in \Omega} \underline{s}_\omega \underline{s}_\omega^T \tag{4.19}$$

Hence, the updated link weight matrix is simply the average of the outer-products of the vectors of cluster membership indicators. We can make the cluster-structure of the link weight matrix clearer if we introduce the link-weight matrix $\hat{A} = \underline{s}_\omega \underline{s}_\omega^T$ for the cluster indexed ω. With this notation, we can write the updated link weight matrix as the sum of contributions from different clusters, i.e.

$$\hat{A} = \frac{1}{|\Omega|} \sum_{\omega \in \Omega} \hat{A}_\omega \tag{4.20}$$

Finally, we note that the updated link-weight matrix can be written in the compact form

$$\hat{A} = \frac{1}{|\Omega|} S S^T \tag{4.21}$$

Modal Sharpening of the Link-weight Matrix

In practice, the link-weight matrix may be noisy and hence the cluster structure may be subject to error. In an attempt to overcome this problem, in this section we turn our attention to how the updated link-weight matrix may be refined with a view to improving its block structure. The aim here is to suppress structure which is not associated with the principal modes of the matrix.

We commence by focussing in more detail on the significance of the update process described in the previous section. To do this, we return to the expression for the log-likelihood function. The component of the log-likelihood which depends on the cluster indicators is

$$\hat{\mathcal{L}}(A, S) = \sum_{\omega \in \Omega} \sum_{(i,j) \in \Phi} \left\{ s_{i\omega} s_{j\omega} \ln \frac{A_{ij}}{1 - A_{ij}} \right\} \tag{4.22}$$

We can rewrite this component of the log-likelihood function using matrix notation as

$$\hat{\mathcal{L}}(A,S) = Tr[S^T T S] \tag{4.23}$$

where S is the cluster membership matrix defined earlier and T is the $|V| \times |V|$ matrix whose elements are given by

$$T_{ij} = \ln \frac{A_{ij}}{1 - A_{i,j}} \tag{4.24}$$

Since the trace of a matrix product is invariant under the cyclic permutation of the matrices, we have that

$$\hat{\mathcal{L}}(A,S) = Tr[S S^T T] \tag{4.25}$$

From the previous section of this paper, we know that the matrix SS^T is related to the updated link-weight matrix in the following manner

$$SS^T = |\Omega| \hat{A} \tag{4.26}$$

Hence,

$$\hat{\mathcal{L}}(A,S) = |\Omega| Tr[\hat{A} T] \tag{4.27}$$

As shown by Scott and Longuet-Higgins (Scott and Longuet-Higgins, 1990) in a study of correspondance matching, this quantity may be maximised by performing an eigendecomposition on the matrix T and setting the columns of $\hat{A}$ equal to the eigenvectors of T. It has been shown by Dieci (Dieci, 1996) that the eigenvectors of the matrices A and $\ln A$, have identical directions. This suggests a means by which we might refine our estimate of the link-weight matrix.

To pursue this idea further we note that the eigenvector expansion of the matrix A is

$$A = \sum_{k=1}^{|V|} \lambda_k \mathrm{x}_k \mathrm{x}_k^T \tag{4.28}$$

This matrix may be approximated by the same-sign eigenvectors, i.e.

$$A \simeq \sum_{\omega \in \Omega} \lambda_\omega \mathrm{x^*}_\omega \mathrm{x^*}^T_\omega \tag{4.29}$$

We can exploit this property to develop a means of refining the structure of the updated link-weight matrix, with the aim of improving its block structure. To commence, let the rank-one matrix

$$\hat{A}_\omega = \mathrm{s}_\omega \mathrm{s}_\omega^T \tag{4.30}$$

represent the component of the updated link-weight matrix which results from the cluster of nodes indexed ω. We can write

$$A = \sum_{\omega \in \Omega} \hat{A}_\omega + E \tag{4.31}$$

where E is an error matrix. Since $\hat{A}_\omega$ is rank one, it has only one non-zero eigenvalue λ^*_ω. Let the eigenvector corresponding to this eigenvalue be ϕ^*_ω. With this notation, we can write

$$A = \sum_{\omega \in \Omega} \lambda^*_\omega \phi^*_\omega (\phi^*_\omega)^T + E \tag{4.32}$$

Hence, provided that the error matrix E is small, then we can approximate the updated link-weight matrix $\hat{A}$ by the matrix

$$A^* = \sum_{\omega \in \Omega} \frac{\lambda^*_\omega}{|\Omega|} \phi^*_\omega (\phi^*_\omega)^T \tag{4.33}$$

Thus, we construct a new link-weight matrix from the leading eigenvectors of the cluster link-weight matrices $\hat{A}_\omega = \mathrm{s}_\omega \mathrm{s}_\omega^T$. The eigenvalues and eigenvectors of the new link-weight matrix are the leading eigenvalues and eigenvectors of the individual cluster adjacency matrices. We refer to this process as modal sharpening. The effect is to impose a strong block structuring on the link-weight matrix. For each cluster or mode of the link-weight matrix, we effectively partition the nodes into foreground and background. The background nodes for each cluster are then removed from further consideration. This can be viewed as a form of noise removal.

The modal decomposition of the link-weight matrix also suggests an initialisation. Following Chapter 3, we assign cluster-membership probabilities so that they are close to the eigenmodes of the raw adjacency matrix. To do this we use the same sign eigenvectors by setting

$$s_{iw} = \frac{|\mathbf{x}^*_\omega(i)|}{\sum_{i \in V_\omega} |\mathbf{x}^*_\omega(i)|} \tag{4.34}$$

Since each node is associated with a unique cluster, this means that the updated affinity matrix is composed of non-overlapping blocks. Moreover, the link-weights are guaranteed to be in the interval $[0, 1]$. Finally, it is important to note that the updating of the link-weights is a unique feature of our algorithm which distinguishes it from the pairwise clustering methods of Hoffman and Buhmann (Hofmann and Buhmann, 1997) and Shi and Malik (Shi and Malik, 1997).

Updating the Cluster Membership Variables

We can repeat the gradient-based analysis of the log-likelihood function to develop update equations for the cluster-membership variables. Following the procedure in Chapter 3, we commence by computing the derivatives of the expected log-likelihood function with respect to the cluster-membership variable

$$\frac{\partial \mathcal{L}}{\partial s_{i\omega}} = \sum_{j \in V_\omega} s_{j\omega} \ln \frac{A_{ij}}{1 - A_{ij}} \tag{4.35}$$

After simplifying the argument of the exponential, the update formula reduces to

$$\hat{s}_{i\omega} = \frac{\prod_{j \in V_\omega} \left\{ \frac{A_{i,j}}{1-A_{ij}} \right\}^{s_{j\omega}}}{\sum_{i \in V_\omega} \prod_{j \in V_\omega} \left\{ \frac{A_{ij}}{1-A_{ij}} \right\}^{s_{j\omega}}} \tag{4.36}$$

It is worth pausing to consider the structure of this update equation. First, the updated link weights are an exponential function of the current ones. Second, the exponential constant is greater than unity, i.e. there is re-enforcement of the cluster memberships, provided that $A_{i,j} > \frac{1}{2}$.

We can take the analysis of the cluster membership update one step further and establish a link with the eigenvectors of the updated adjacency matrix. To this end we make use of the matrix T defined in Equation (4.24). We turn our attention to the argument of the exponential appearing in Equation (4.36) and write

$$\sum_{j \in V_\omega} s_{j\omega} \ln \frac{A_{i,j}}{1 - A_{i,j}} = \Big(T \underline{s}_\omega \Big)_i \tag{4.37}$$

In other words, the argument of the exponential is simply the ith component of the vector obtained by the matrix multiplication $T\underline{s}_\omega$.

Next, consider the case when the vector $\underline{s}_\omega$ is an eigenvector of the matrix T. The eigenvector equation for the matrix T is

$$T \underline{z}_\omega = \hat{\lambda}_\omega \underline{z}_\omega$$

where $\hat{\lambda}_\omega$ is the ω^{th} eigenvalue and $\underline{z}_\omega$ is the corresponding eigenvector. Hence, when the vector of cluster memberships $\underline{s}_\omega$ is an eigenvector of T, then we can write

$$\Big(T \underline{s}_\omega \Big)_i = \underline{z}_\omega(i)$$

where $\underline{z}_\omega(i)$ is the ith component of the vector $\underline{z}_\omega$. If this is the case, then we can identify the pairwise clusters with the eigenmodes of T, and the update equation becomes

$$\hat{s}_{i\omega} = \frac{\exp\Big[\hat{\lambda}_\omega z_\omega(i) \Big]}{\sum_{i \in V_\omega} \exp\Big[\hat{\lambda}_\omega z_\omega(i) \Big]} = \frac{\tilde{\lambda}_\omega^{z_\omega(i)}}{\sum_{i \in V_\omega} \tilde{\lambda}_\omega^{z_\omega(i)}} \tag{4.38}$$

where

$$\hat{\lambda}_\omega = \ln \tilde{\lambda}_\omega$$

This update process becomes particularly simple when it is applied to the adjacency matrix obtained by modal sharpening. Let the elements of matrix T^* be given by $T^*_{i,j} = \ln\left(\frac{A^*_{i,j}}{1-A^*_{i,j}}\right)$. Since the logarithm of matrix T is a polynomial in T (i.e. a primary matrix function) (Horn and Johnson, 1991) and matrix T is positive, definite and invertible, the directions of the eigenvectors of the matrices T and $\ln T$ are identical (Dieci, 1996). Hence, we can compute the eigenvectors of T^* by the eigenvectors of T. As a result, the updated cluster membership variables can be computed directly from the eigenvectors of the matrix T^*, i.e. the leading eigenvector $\tilde{\phi}_\omega$. Thus, we can write

$$\hat{s}_{i\omega} = \frac{\tilde{\lambda}_\omega^{\tilde{\phi}_\omega(i)}}{\sum_{i \in V_\omega} \tilde{\lambda}_\omega^{\tilde{\phi}_\omega(i)}} \tag{4.39}$$

In this way, by computing the eigenmodes of the matrix T^*, we can update the individual cluster membership indicators.

4.1.4 Algorithm description

We use the update steps developed in Section 4.1 to develop an iterative grouping algorithm. The steps of the algorithm are as follows:

- Step 0: The algorithm is initialised using the initial link weight matrix A. This is computed from raw image data and is domain specific. Some examples of how this is done are provided later on in Section 4.4 for line-segment grouping and in Section 4.5 for motion segmentation.

- Step 1: The same-sign eigenvectors are extracted from the current link-weight matrix A. These are used to compute the cluster-membership matrix S using Equation (4.39). The number of same-sign eigenvectors determines the number of clusters for the current iteration. This number may vary from iteration to iteration. In our experiments, the complexity of computing the first eigenvector using the power method was on average $5.8n^2$, where n is the order of matrix A.

- Step 2: For each cluster we compute the link-weight matrix $A_\omega = \underline{s}_\omega \underline{s}_\omega^T$. We perform an eigen-decomposition on each cluster link-weight matrix to extract the leading eigenvalue λ_ω^* and the leading eigenvector ϕ_ω^*. Since the matrix A_ω is defined as the product of two vectors, the computation of the first eigenvector can be regarded for computational purposes as a normalisation of the vectors $\underline{s}_\omega$. Therefore, the complexity can be reduced to approximately $3n$ for each cluster.

- Step 3: We perform modal sharpening by applying Equation (4.33) to the leading eigenvalues and eigenvectors of the cluster link-weight matrices $\hat{A}_\omega$. The resulting revised link-weight matrix is A^*. The complexity of computing matrix A^* is n^2.

- Step 4: An updated matrix of cluster membership variables $\hat{S}$ is computed. This is done by applying Equation (4.36) to the revised link-weight matrix obtained by modal sharpening, i.e. A^*, and the current cluster membership matrix S. Making use of the fact that the denominator of Equation (4.36) is the sum of the quantities in the numerator for every $\hat{s}_{i\omega}$, the complexity of this step can be reduced to approximately $5n$.

- Step 5: The updated cluster membership matrix $\hat{S}$ is used to compute the updated link-weight matrix $\hat{A} = \frac{1}{|\Omega|} S S^T$. This revised link-weight matrix is passed to Step 1. The average complexity of this step in our experiments was $2.4n^2$.

Steps 1 to 5 are iterated in sequence until convergence is reached.

The iterative process described above clearly has features which are reminiscent of the EM algorithm outlined in Chapter 3. However, there are important differences. These mainly stem from our use of the modal sharpening process to improve the block and cluster structure of the link-weight matrix. We have recently reported the use of an EM algorithm based on mixtures of Bernoulli distributions. We will compare the method described in this thesis with this EM algorithm in our experimental evaluation. The algorithm commences from the initial set of cluster memberships defined by the eigenmodes of the raw affinity matrix A. In the E or expectation step we compute a matrix of *a posteriori* cluster membership probabilities Q. This matrix is found by taking the expectation of the cluster-membership matrix i.e. $Q = E(S)$. In the M or maximisation step we perform two updates. First, we compute the updated link-weight matrix using the formula $\hat{A} = E(SS^T)$. Second, we compute a revised matrix of cluster-membership indicators $\hat{S}$ using a variant of the soft-assign method outlined in Chapter 3. Hence, the main differences are that the number of clusters is set at the outset of the algorithm, and that there is no modal sharpening of the link-weight matrix.

4.2 Convergence Analysis

In this section we provide some analysis of the convergence properties of the new clustering algorithm. We are interested in the relationship between this modal analysis and the updated cluster membership variables. Making use of the shorthand defined in Equation (4.24) and substituting in Equation (4.14) the update formulae for the link-weight matrix and the cluster membership indicators given in Equations (4.39) and (4.18), it is a straightforward matter to show that the corresponding updated log-likelihood function is given by

$$\mathcal{L}(\mathcal{A},\mathcal{S}) = \sum_{\omega\in\Omega} \sum_{(i,j)\in|V_\omega\times V_\omega|} \left\{ T_{ij} \frac{\tilde{\lambda}_\omega^{z_\omega(i)+z_\omega(j))}}{\sum_{i'\in V_\omega} \tilde{\lambda}_\omega^{z_\omega(i')} \sum_{j'\in V_\omega} \tilde{\lambda}_\omega^{z_\omega(j')}} + \ln(1-A_{ij}) \right\} \tag{4.40}$$

We would like to understand the conditions under which the likelihood is maximised by the update process. We hence compute the partial derivative of $\mathcal{L}$ with respect to $\tilde{\lambda}_\omega$. After collecting terms and some algebra we find

$$\begin{aligned} \frac{\partial \mathcal{L}(\mathcal{A},\mathcal{S})}{\partial \tilde{\lambda}_\omega} = & \sum_{\omega\in\Omega} \sum_{(i,j)\in|V_\omega\times V_\omega|} \left\{ \frac{T_{ij}\tilde{\lambda}_\omega^{z_\omega(i)+z_\omega(j)}}{\tilde{\lambda}_\omega \big(\sum_{i'\in V_\omega} \tilde{\lambda}_\omega^{z_\omega(i')}\big)^2} \Bigg(z_\omega(i) + z_\omega(j) \right. \\ & \left. - 2 \frac{\sum_{i'\in V_\omega} z_\omega(i')\tilde{\lambda}_\omega^{z_\omega(i')}}{\sum_{i'\in V_\omega} \tilde{\lambda}_\omega^{z_\omega(i')}} \Bigg) \right\} \end{aligned} \tag{4.41}$$

Since the natural logarithm function is strictly increasing, the maximum of the likelihood will occur at the same point as the maximum of the log-likelihood function. Hence, we set the partial derivative to zero. This condition is satisfied when

$$z_\omega(i) + z_\omega(j) = 2\frac{\sum_{i' \in V_\omega} z_\omega(i')\tilde{\lambda}_\omega^{z_\omega(i')}}{\sum_{i' \in V_\omega} \tilde{\lambda}_\omega^{z_\omega(i')}} \tag{4.42}$$

Unfortunately, this condition is not always guaranteed to exist. However, from Equation (4.41) we can conclude that the following will always be the best approximation

$$\tilde{\lambda}_\omega^{z_\omega(i)+z_\omega(j)} << \tilde{\lambda}_\omega \left(\sum_{i' \in V_\omega} \tilde{\lambda}_\omega^{z_\omega(i')} \right)^2 \tag{4.43}$$

If T^* is a non-negative irreducible symmetric matrix then the coefficients of the leading eigenvector z$_*$ associated with the eigenvalue $\hat{\lambda}_\omega = \ln \tilde{\lambda}_\omega$ are each positive (Varga, 2000). As a result, the quantity $\sum_{i' \in V_\omega} \tilde{\lambda}_\omega^{z_\omega(i')}$ will be maximized when $\hat{\lambda}_\omega$ is maximum. Hence, $\mathcal{L}$ will be maximized by the first (maximum) eigenvalue of T^*.

Further, since the matrix T^* is symmetric and non-negative, the leading eigenvalue of every principal minor of matrix T^* does not exceed the value of its maximal eigenvalue (Gantmacher, 1971). From this monotonicity principle, we can conclude that the maximal eigenvalue of matrix T^* can be used for measuring the degree of convergence and the stopping point. To do this, it is enough to compare the maximal eigenvalue of the matrix T^* when it is passed on to step 1 from step 5 (see algorithm description). If the leading eigenvalue of the matrix T^* starts increasing over iteration number, then the stopping point has been reached.

4.3 Algorithm Behaviour

In this section we illustrate how the cluster-membership variables and the link-weight matrix evolve over the steps described in the previous section. To this end, we have generated a set of four point-patterns consisting of 190 points corresponding to two clusters. The first cluster consists of 50 points distributed normally around a fixed center point. The second cluster is an anullus consisting of 150 normally distributed points. For both clusters, the variance of the Gaussian kernel was set to 1.5.

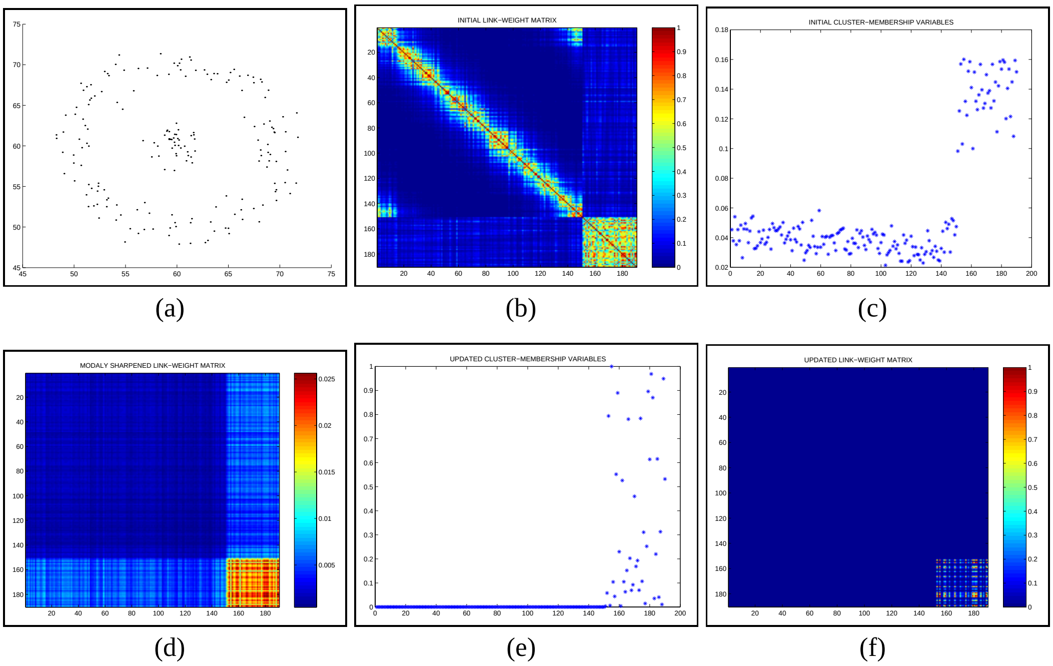
We assign the linking probability between the point indexed i and the point indexed j using the exponential distribution

$$A_{ij} = \exp(-kr_{ij}^2) \tag{4.44}$$

where r_{ij}^2 is the squared Euclidean distance on the x-y plane and $k \in (0, \infty]$ is a constant.

We focus our attention on the evolution of the link-weights and the cluster-membership variables corresponding to the first cluster. In the top row of Figure 4.1 we show an example of the point-patterns under study, the initial adjacency matrix A and the corresponding leading eigenvector. From the affinity matrix, it is clear that there are two clusters. The strongest cluster is in the bottom right-hand corner, and corresponds to the cluster of points in the centre. The weaker cluster is in the top left-hand corner and corresponds to the annulus of surrounding points. In the bottom row of Figure 4.1, from left-to-right we show the matrix A^*, the updated cluster-membership variables and the matrix $\hat{A}$. There are three important effects of the algorithm steps on the cluster variables. First, the first eigenvector of the adjacency matrix A presents two well defined groups of coefficient-values that correspond to the two clusters in the point-pattern. Second, when the matrix A^* is computed a strong block structure is imposed by the modal sharpening step. Finally, the soft-assign step nulls all the cluster-membership variables that correspond to the elements in the second cluster (top left-hand corner of A).

To conclude this section, we study the effect of varying k in the output of our clustering algorithm. To perform this study, we have computed the adjacency matrices of our four point-patterns varying k from 0.1 to 0.5. In Figure 4.2 we show the clustering results with $k = 0.3, 0.4$ and the fraction of points that are mis-assigned by the clustering algorithm. From Figure 4.2c, we can conclude that the output is stable and reliable over the $[0.25, 0.35]$ interval, with $k = 0.3$ as its optimum.



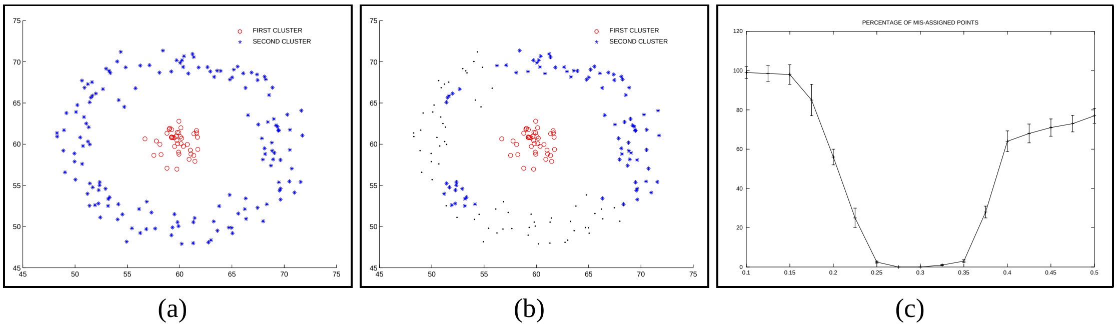
(a) (b) (c)

(d) (e) (f)

Figure 4.1: (a) Example of the point-patterns under study; (b) Initial adjacency matrix ($k = 0.275$);(c),(d),(e),(f) evolution of the clustering variables (see text).

4.4 Line Grouping

In this section we provide the first example application of our new clustering method. This involves the grouping or linking of line-segments. There are two aspects to this study. We commence by providing some examples for synthetic images. Here we investigate the sensitivity of the method to clutter and compare it with an eigen-decomposition method. The second aspect of our study focuses on real world images with known ground-truth.



(a) (b) (c)

Figure 4.2: (a),(b) Clustering results with $k = 0.3, 0.4$; (c) fraction of mis-assigned points as a function of k.

The initial line-linking probability is assigned using the probabilistic line-grouping field described in earlier Section 3.3.1.

In our experiments we provide comparison with three different algorithms. The first of these is the EM algorithm described in Chapter 3, which uses a mixture of Bernoulli distributions. This algorithm does not however use modal decomposition of the link-weight matrix. The second method is that of Sarkar and Boyer (Sarkar and Boyer, 1998). Thirdly, there is the normalised cut (recursive bisection) method of Shi and Malik (Shi and Malik, 1997).

Our first experiment concerns a hexagonal arrangement of lines to which increasing numbers of randomly distributed distractors have been added. The positions, orientations and lengths of the distractors have been drawn from uniform distributions. The images used in this study are shown in the first column of Figure 4.3. The distractor density increases from top to bottom in the first column of the figure. From the arrangement

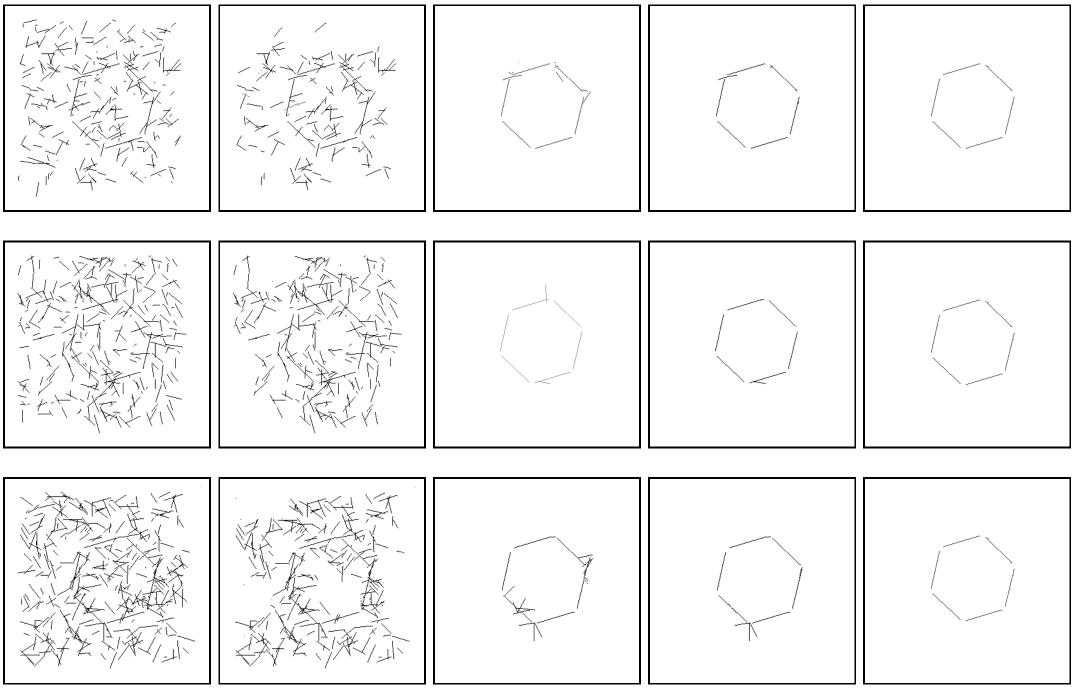


Figure 4.3: Left-hand column: patterns containing 250, 300 and 350 randomly positioned background lines; each subsequent column shows the result obtained with the Sarkar and Boyer algorithm (second column), the results when a standard EM algorithm is used (third column), the cluster memberships obtained using the normalized cut (fourth column) and the cluster-memberships obtained using our new method (last column) for each of the images shown in the first column.

of lines in each panel of the figure, we compute the link-weight matrix using Equation (3.23).

In the second column of the figure, we show the results obtained using the Sarkar and Boyer method. The third column shows the result obtained using the EM algorithm described in Chapter 3. In the fourth column, we show the results obtained using the normalised cuts method. Finally, the fifth column shows the results obtained using our new method.

To display the results of our method we first label the lines according to the cluster associated with the largest membership variable. For the line indexed i, the cluster-label is $\Lambda_i = \arg\max_{\omega \in \Omega} \hat{s}_{i\omega}$. Next, we identify the cluster which contains the largest number of lines from the hexagonal pattern. Suppose that the index of this cluster is denoted by ω_p. The lines displayed are those belonging to the set $\Lambda_{\omega_p} = \{\Lambda_i | \Lambda_i = \omega_p\}$.

There are a number of conclusions that can be drawn from these examples. First, the quality of the results obtained increases as we move from left-to-right across the figure. In the case of the Boyer and Sarkar method, little of the distractor structure is removed. In the case of the EM algorithm and the normalised cuts method, most of the background is removed, but a few distractors remain attached to the heaxagonal pattern of lines.

We have repeated the experiments described above for a sequence of synthetic images in which the density of distractors increases. For each image in turn we have computed the number of distractors merged with the foreground pattern and the number of foreground line-segments which leak into the background. Figures 4.4 a and b respectively show the fraction of nodes merged with the foreground and the fraction of nodes which leak into the background as a function of the number of distractors. The four curves shown in each plot are for the non-iterative eigendecomposition method of Sarkar and Boyer, the EM algorithm presented in Chapter 3, the Shi and Malik normalised cuts method, and for the new method described in this chapter.

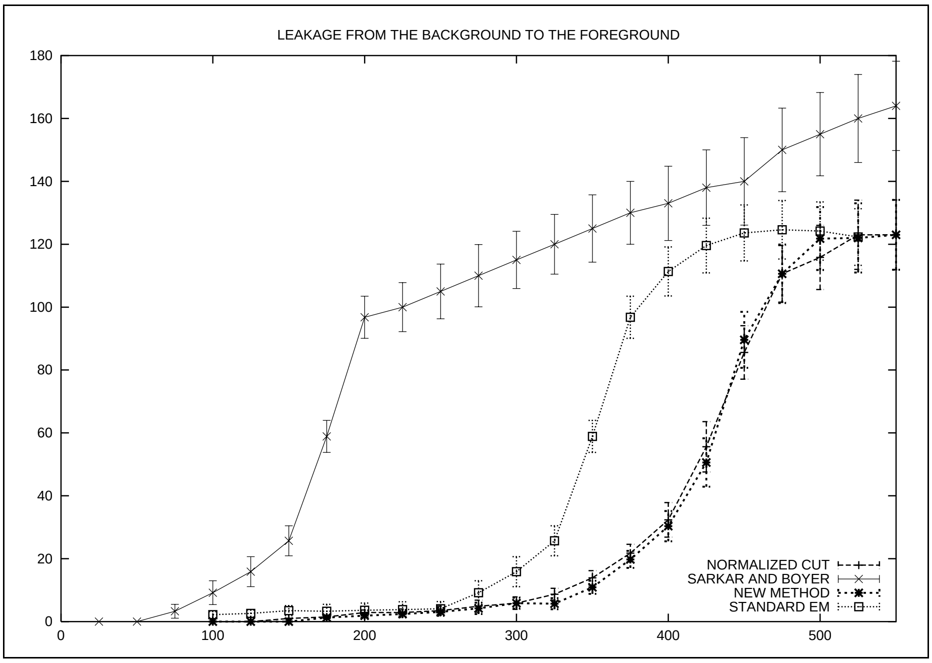
Next, we turn our attention to the fraction of foreground lines which leak into the background (i.e. those which are erroneously identified as distractors). From Figure 4.4b a similar pattern emerges to that in Figure 4.4a. In other words, the worst performance is delivered by the Sarkar and Boyer method (Sarkar and Boyer, 1998), and the EM-like algorithm presented in Chapter 3 gives intermediate performance. However, now the new method gives a margin of improvement over the Shi and Malik normalised cuts method.

Finally, we present results on a real-world image in Figure 4.5. The edges shown in Figure 4.5b have been extracted from the raw image using the Canny edge-detector. Straight-line segments have been extracted using the method of Yin (Peng-Yeng, 1998). The resulting groupings obtained with our new method are shown in Figure 4.5c.

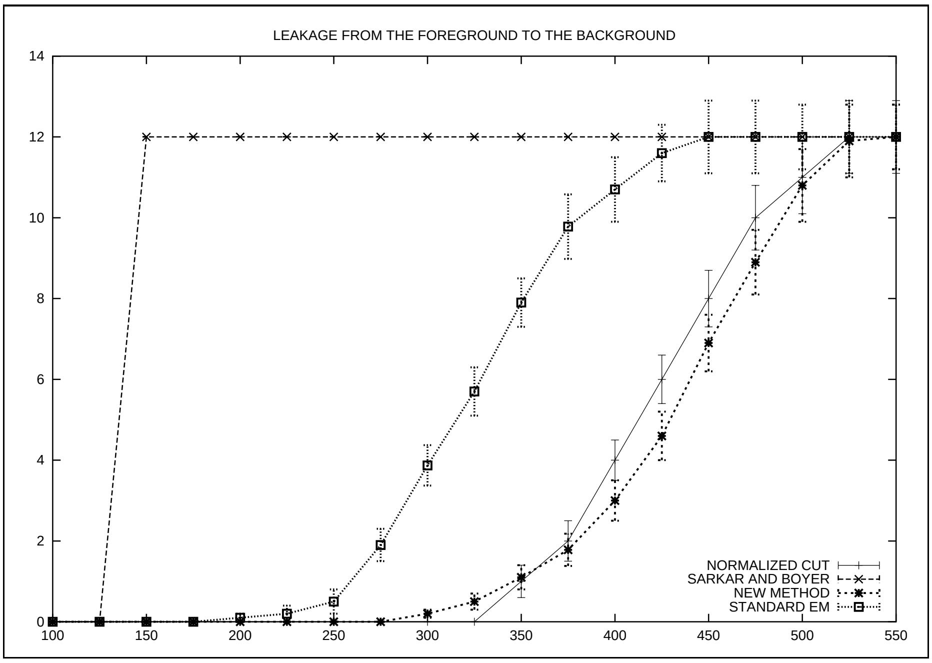
4.5 Motion Segmentation

The second application of our pairwise clustering method focusses on the segmentation of independently moving objects from image sequences. The motion vectors used in our analysis have been computed using a single resolution block matching algorithm (Hsieh et al., 1990). The method measures the similarity of motion blocks using spatial correlation and uses predictive search to efficiently compute block-correspondences in different frames. The block matching algorithm assumes that the translational motion from frame to frame is constant. The current frame is divided into blocks that will be compared with the next frame in order to find the displaced coordinates of the corresponding block within the search area of the reference frame. Since the computational complexity is much lower than the optical flow equation and the pel-recursive methods, block matching has been widely adopted as a standard for video coding and hence it provides a good starting point.

However, the drawback of the single resolution block-matching scheme is that while the high resolution field of motion vectors obtained with small block sizes captures fine detail, it is susceptible to noise. At low resolution, i.e. for large block sizes, the field of motion vectors is less noisy but the fine structure is lost. To strike a compromise between low-resolution noise suppression and high resolution recovery of fine detail, there have been several attempts to develop multi-resolution block matching algorithms. These methods have provided good predictive performance and also improvements in speed. However, one of the major problems with the multi-resolution block matching method is



(a) Number of nodes leaked into the foreground versus number of distractors added for the method of Sarkar and Boyer (top curve), the EM algorithm (middle curve), the normalized cut (lower curve) and the maximum likelihood framework (bottom curve).



(b) Number of nodes leaked into the foreground versus number of distractors added for the method of Sarkar and Boyer (top curve), the EM algorithm (middle curve), the normalized cut (lower curve) and the maximum likelihood framework (bottom curve).

Figure 4.4: Comparison between a non-iterative eigendecomposition approach, the normalized cut and the two pairwise clustering algorithms presented in this thesis.

that random motions can have a significant degradational effect on the estimated motion field. For these reasons, we have used a single high-resolution block matching algorithm to estimate the raw motion field. This potentially noisy information is refined in the motion segmentation step, where we exploit hierarchical information.

We pose the problem of grouping motion blocks into coherent moving objects as that of finding pairwise clusters. The 2D velocity vectors for the extracted motion blocks are characterised using a matrix of pairwise similarity weights. Suppose that $\hat{\mathbf{n}}_i$ and $\hat{\mathbf{n}}_j$ are the unit motion vectors for the blocks indexed i and j. The elements of the initial link-weight matrix are given by

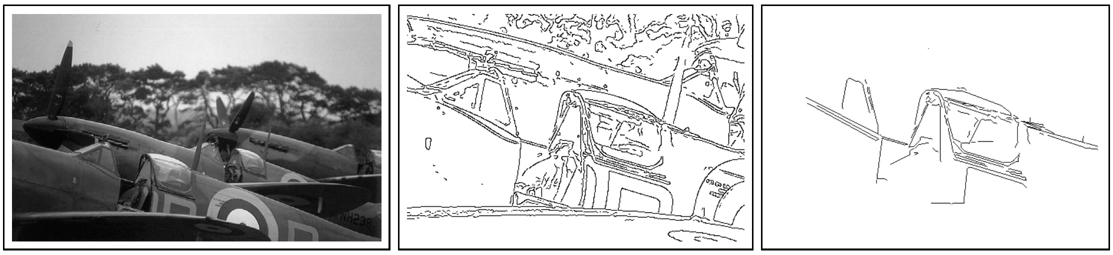
$$A^{(0)}_{i,j} = \begin{cases} \frac{1}{2}(1 + \hat{\mathbf{n}}_i \cdot \hat{\mathbf{n}}_j) & \text{if } i \neq j \\ 0 & otherwise \end{cases} \tag{4.45}$$

4.5.1 Hierarchical Motion Segmentation

As mentioned earlier, we use a single-level high-resolution block-matching method to estimate the motion field. The resulting field of motion vectors is therefore likely to be noisy. To control the effects of motion-vector noise, we have developed a multi-resolution extension to the clustering approach described above.

The adopted approach is as follows.

- We obtain the a high resolution field of motion vectors U_H using blocks of size k-pixels and a low-resolution motion field U_L using blocks of size $2k$ pixels.



(a) (b) (c)

Figure 4.5: Real world images: (a) raw image; (b) results for the Canny edge detector; (c) result of applying the Eigendecomposition algorithm.

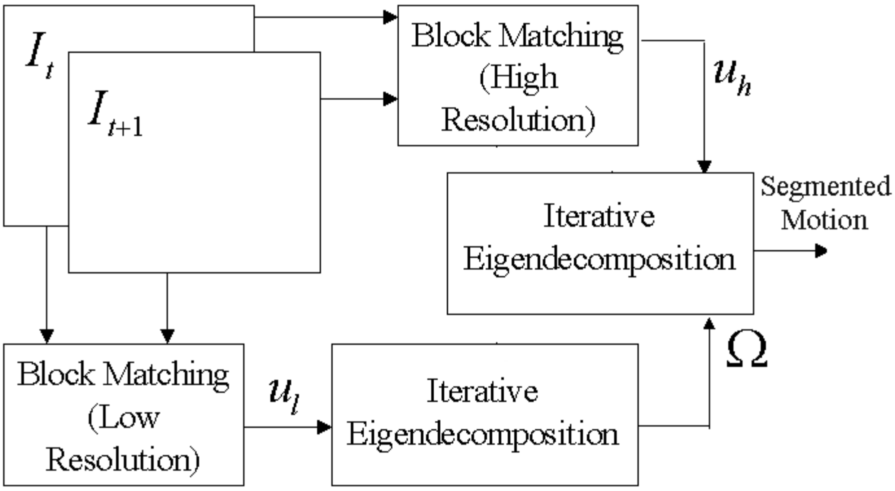

I_t
I_{t+1}
Block Matching (High Resolution)
u_h
Iterative Eigendecomposition
Segmented Motion
Ω
Block Matching (Low Resolution)
u_l
Iterative Eigendecomposition

Figure 4.6: Diagram of the hierarchical motion segmentation system

- We apply our clustering algorithm to the low resolution motion field U_L. We note the number of clusters $\mid \Omega_L \mid$ detected.
- We make a second application of our clustering algorithm to the hight-resolution motion field U_H. Here we select only the first $\mid \Omega_L \mid$ eigenvalues of the motion-vector similarity matrix as cluster centres.

In this way we successively perform the motion estimation at low and high resolution. The number of clusters detected at low resolution is used to constrain the number of permissible high resolution clusters. This allows the high-resolution clustering process to deal with fine detail motion fields without succumbing to noise. There is scope to extend the method and develop a pyramidal segmentation strategy. The structure of the hierarchical system can be seen in Figure 4.6.

4.5.2 Motion Experiments

We have conducted experiments on motion sequences with known ground truth. In Figure 4.7 we show some results obtained with five frames of the well-known “Hamburg Taxi” sequence. The top row shows the hand-labelled ground-truth segmentation for the motion sequence. The second row shows the corresponding image frames from the motion sequence. In the third and fourth rows we respectively show the low resolution and high resolution block motion vectors. At low-resolution we use 16×16 pixel blocks to perform motion correspondence and compute the motion vectors; for the high resolution motion field the block size is 8×8 pixels. The fifth row in the figure shows the moving objects segmented from the motion field using the normalised cut method. The sixth row shows the motion segmentation obtained using the new method described in this paper. Turning our attention to the results delivered by our new method (i.e. the sixth row of the figure)

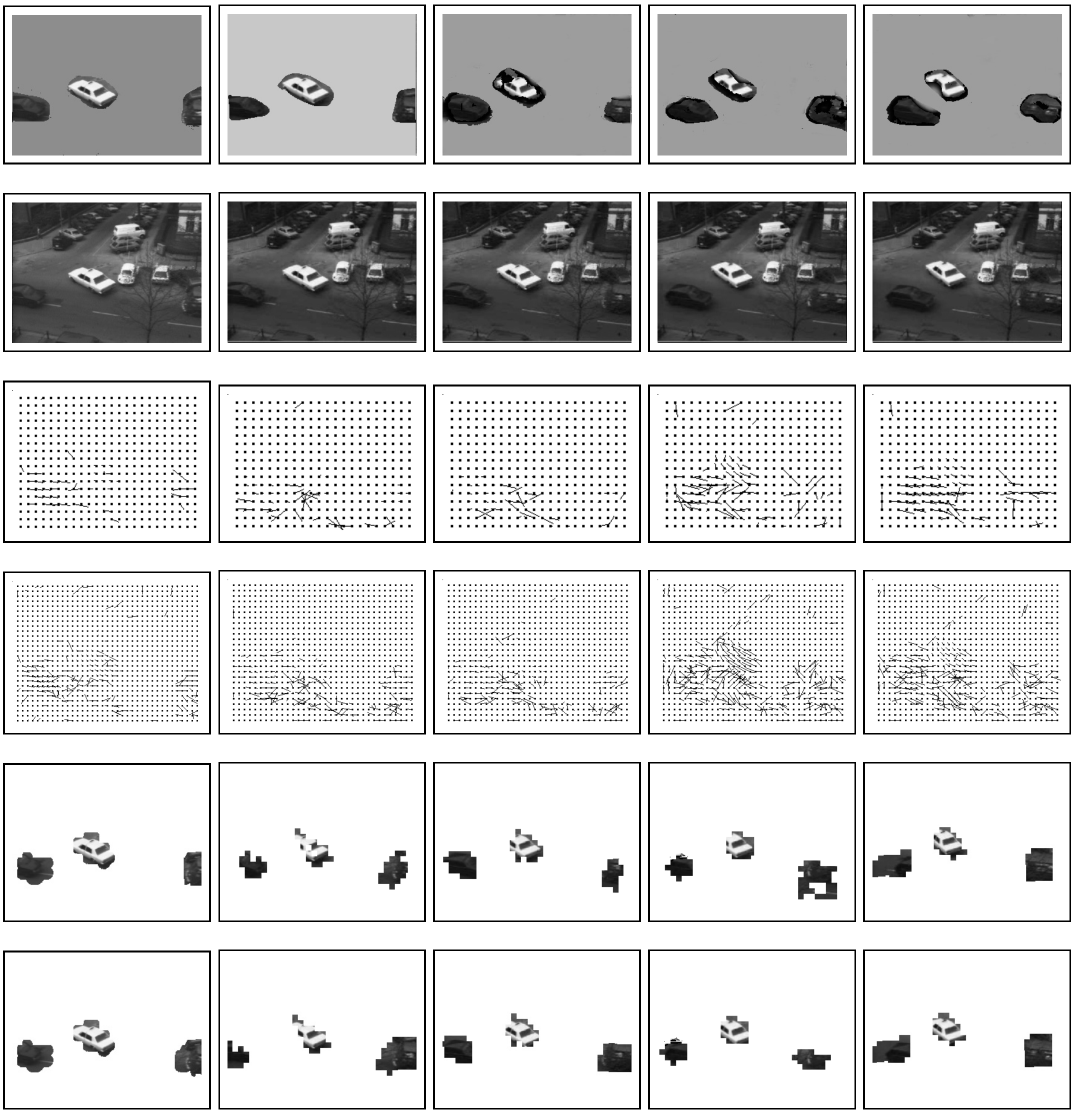


Figure 4.7: Top row: ground truth for the 1st, 4th, 8th, 12th and 16th frame of the ”Hamburg Taxi” sequence; second row: original frames ; third and fourth rows: low and high resolution motion fields; fifth row: Final motion segmentation obtained using the normalized cut; bottom row: motion segmentation using our new method.

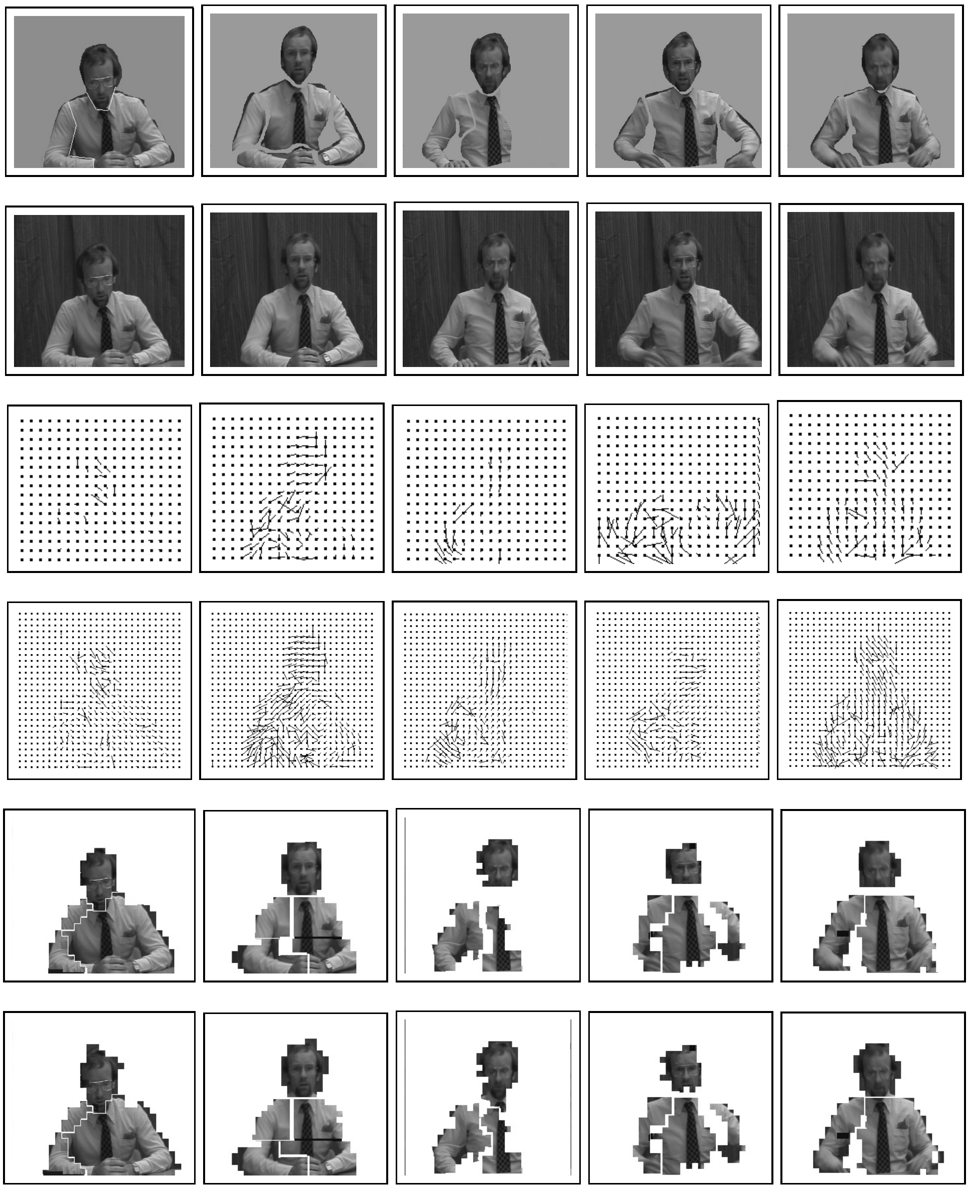


Figure 4.8: Top row: ground truth for the 1st, 4th, 8th, 12th and 16th frame of the “Trevor White” sequence; second row: original frames; third and fourth rows: low and high resolution motion fields; fifth row: motion segmentation obtained using the normalized cut; bottom row: motion segmentation obtained using our new method.

in each frame there are 3 clusters which match closely to the ground truth data shown. In fact, the three different clusters correspond to distinct moving vehicles in the sequence. These clusters again match closely to the ground-truth data. The results obtained using the normalised cut are good, but some of regions are slightly undersegmented.

Figure 4.8 repeats these experiments for the “Trevor White” sequence. The sequence

Sequence	Cluster	% Error (Normalized Cut)	% Error (Our approach)
Trevor White	Right arm	7.2 %	8 %
Trevor White	Chest	7.4 %	6 %
Trevor White	Head	13.6%	12%
Ham. Taxi	Taxi	6 %	4 %
Ham. Taxi	Far Left Car	4 %	3 %
Ham. Taxi	Far Right Car	14 %	10 %

Table 4.1: Error percentage for the two image sequences.

of rows is the same as in Figure 4.7. Here the block sizes are respectively 24×24 and 12×12 pixels. There are three motion clusters which correspond to the head, the right arm, and the chest plus left arm. These clusters again match closely to the ground-truth data. The normalised cuts method again under-segments the motion regions when compared with our new method.

In Table 4.1 we provide a more quantitative analysis of these results. The table lists the fraction of the pixels in each region of the ground truth data which are mis-assigned by the clustering algorithm. There are different columns in the table for the normalised cuts method and our new method. Turning our attention to our new method, the best results are obtained for the chest-region, the taxi and the far-left car, where the error rate is a few percent. For the far-right car and the head of the newsreader, the error rates are about 10%. The problems with the far-right car probably relate to the fact that it is close to the periphery of the image. The normalised cuts method consistently gives error rates which are about 2% worse than our new method.

4.6 Conclusions

In this chapter, we have developed a maximum likelihood framework for pairwise clustering. The method commences from a specification of the pairwise clustering problem in terms of a matrix of link-weights and a set of cluster membership indicators. The likelihood function underpinning our method is developed under the assumption that the cluster membership indicators are random variables which are drawn from a Bernoulli distributions. The parameter of the Bernoulli distributions are the link-weights. Based on this model, we develop an iterative process for updating the link-weights and the cluster membership indicators in interleaved steps.

To understand the relationships between our probabilistic method and the graph-spectral approach to pairwise clustering, we present a variational analysis for the expected log-likelihood function. This reveals two important results. First, we demonstrate a relationship between the updated cluster membership variables resulting from our variational analysis and the eigenvectors of the link-weight matrix. Second, we show that the log-likelihood function is maximised by the leading eigenvector of the link-weight matrix.

We apply the resulting pairwise clustering process to a number of image segmentation and grouping problems. These include grey-scale region segmentation, motion segmentation and line-segment grouping. A sensitivity analysis reveals that our new clustering method outperforms existing ones in terms of its ability to reduce the effects of foreground-background and background-foreground leakage in the segmentation and grouping processes.

The advantages of the method are twofold. First, here we provide a more direct way of updating the link-weight matrix. As a result, the method is faster to converge than the EM algorithm developed in Chapter 3. Second, we remove noisy link-weights by refining the structure of the updated link-weight matrix. This provides a margin of improvement over competing methods.

Chapter 5

Shape-from-shading

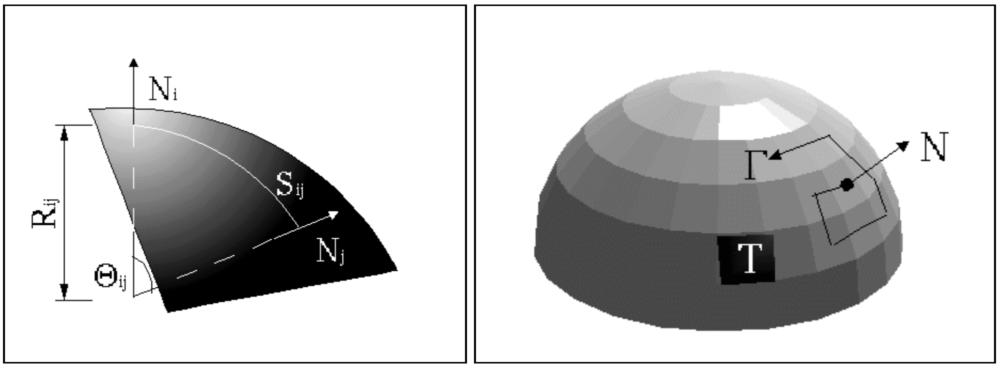
In Chapters 3 and 4, we developed two eigenvector methods and applied them to segmentation and grouping problems. However, eigenvector methods can also be used for solving path-based problems. In this chapter and in Chapter 6, we turn our attention to path-based problems in intermediate and high-level vision. Here, we consider how path-based spectral methods can be used for shape-from-shading. This involves the recovery of field of surface normals from intensity data and the reconstruction of the surface by integration.

Our contribution in this chapter is the development of a graph-spectral method for recovering an integration path for surface height recovery in shape-from-shading. Here the abstract structure under study is a weighted graph. The weight matrix represents the curvature dependant transition probability between pairs of surface normals. The aim is to recover the integration path that minimises the sum of curvature weights across the field of surface normals. This can be viewed as a problem of graph-seriation (Atkins et al., 1998), which involves ordering the set of nodes in a graph in a sequence such that strongly correlated elements are placed next to one another. The seriation problem can be approached in a number of ways. Clearly the problem of searching for a serial ordering of the nodes, which maximally preserves the edge ordering is one of exponential complexity. As a result approximate solution methods have been employed. These involve casting the problem in an optimisation setting. Hence techniques such as simulated annealing and mean field annealing have been applied to the problem. It may also be formulated using semidefinite programming, which is a technique closely akin to spectral graph theory since it relies on eigenvector methods. However, recently a graph-spectral solution has been found to the problem. Atkins, Bowman and Hendrikson (Atkins et al., 1998) have shown how to use the Fiedler eigenvector of the Laplacian matrix to sequence relational data. The method has been successfully applied to the consecutive ones problem and a number of DNA sequencing tasks.

Unfortunately, the analysis of the seriation problem presented in the paper by Atkins, Bowman and Hendrikson (Atkins et al., 1998) is not directly applicable to our surface reconstruction problem. We hence provide an analysis which shows how the problem of recovering the optimal seriation path corresponds to maximising a Rayleigh quotient. This in turn establishes the relationship between the seriation path and the leading eigenvector of the transition weight matrix. The sites visited by this path constitute a patch on the surface.

The eigenvector analysis also allows us to recursively segment the surface into patches. The sites visited by the integration path can be viewed as constituting a single surface patch. By taking the outer product of the leading eigenvector, we can compute the transition weight matrix associated with the sites belonging to the patch. The elements of the transition weight matrix associated with the patch may be set to zero, and the leading eigenvector of the residual weight matrix computed. This process may be repeated until the remaining pixels form patches of negligible size.

The graph-spectral shape-from-shading algorithm which we develop from this spectral treatment is as follows. We commence with the surface normals residing on their irradiance cones and pointing in the direction of the smoothed image gradient. By recursively removing the leading eigenvectors from the transition weight matrix, we identify surface patches and recover patch integration paths and hence surface height. Local smoothing is then performed by fitting a quadric patch to the height estimates for the sites contained within each patch. At each image location, we use the quadric patches to adjust the surface normal directions on the local irradiance cone. To do this we rotate the sur-



(a) (b)

Figure 5.1: Geometric meaning of the surface height recovery process

face normals so that they point in the direction of the surface height gradient. From the updated surface normal directions a new transition weight matrix may be estimated. The processes of height estimation and surface normal adjustment are interleaved and iterated until a stable height function is recovered.

The outline of this chapter is as follows. In Section 5.1 we briefly review Worthington and Hancock's geometric framework for Lambertian shape-from-shading. Section 5.2 describes our curvature-based characterisation of the transition weight matrix for sites on the integration path. In Section 5.3 we describe how ideas from spectral graph theory can be used to recover an integration path from the transition weight matrix. Section 5.4 describes the relationship between the leading eigenvectors and surface patches. Sections 5.5 and 5.6 respectively address the problems of height recovery, surface fitting and surface normal adjustment. In Section 5.7 we provide an overview of the resulting iterative algorithm for surface recovery. In Section 5.8, we illustrate how curvature variations affect the recovered path across the surface. We also address the impact of variations in the illuminant direction on the stability of the recovered surface. Section 5.9 provides a sensitivity study of the surface reconstruction process and of the algorithm as a whole making use of synthetic data. Experiments on real-world data are described in Section 5.10. Finally, Section 5.11 provides some conclusions and suggested directions for further investigation.

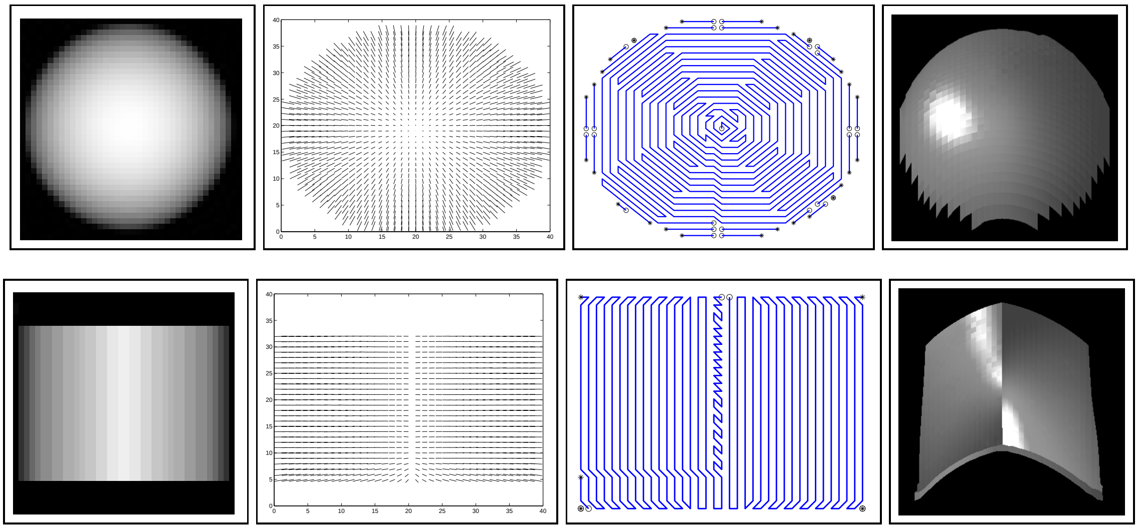


Figure 5.2: Impact of surface curvature variations on the height recovery algorithm : Input images (left-hand-column); needlemaps (second column); plot of the integration path across the surface (third column); view of the reconstructed surfaces (right-hand column)

5.1 Lambertian Reflectance

The starting point for the development of our graph-spectral approach to shape-from-shading is the geometric framework recently reported by Worthington and Hancock (Worthington and Hancock, 1999). Underpinning this work is the observation that for Lambertian reflectance from a matte surface, the image irradiance equation defines a cone of possible surface normal directions (Worthington and Hancock, 1999). The axis of this cone points in the light-source direction and the opening angle is determined by the measured brightness. If the recovered needle-map is to satisfy the image irradiance equation as a hard constraint, then the surface normals must each fall on their respective irradiance cones. Initially, the surface normals are positioned so that their projections onto the image plane point in the direction of the image intensity gradient. Subsequently, there is iterative adjustment of the surface normal directions so as to improve the consistency of the needle-map. In other words, each surface normal is free to rotate about its reflectance cone in such a way as to improve its consistency with its neighbours.

To be more formal let $\vec{s} = [s_x, s_y, s_z]^T$ be a unit vector in the light source direction and let ν_i be the normalised brightness at the image location indexed i. Further, suppose that

$\vec{N}_i^{(k)}$ is the corresponding estimate of the surface normal at iteration k of our algorithm. The image irradiance equation is

$$\nu_i = \vec{N}_i^{(k)} \cdot \vec{s}$$

Hence, the surface normal is constrained for fall on a cone, whose axis is the direction on the light source $\vec{\mathbf{L}}$ and whose opening angle is $\cos^{-1} \nu_i$.

Our aim is to adjust the surface normals by first smoothing them and then projecting them back onto the local irradiance cone. This smoothing is achieved by fitting a quadric patch to reconstructed height data, and estimating the local normal to the fitted surface patch. Details of this process are described later in this chapter. However, suppose that after local smoothing, the off-cone surface normal is $\bar{\vec{N}}_i^{(k)}$. The updated on-cone surface normal which satisfies the image irradiance equation as a hard constraint is obtained via the rotation

$$\vec{N}_i^{(k+1)} = \Phi^{(k)} \bar{\vec{N}}_i^{(k)}$$

The matrix $\Phi^{(k)}$ rotates the smoothed off-cone surface normal estimate by the angle difference between the apex angle of the cone, and the angle subtended between the off-cone

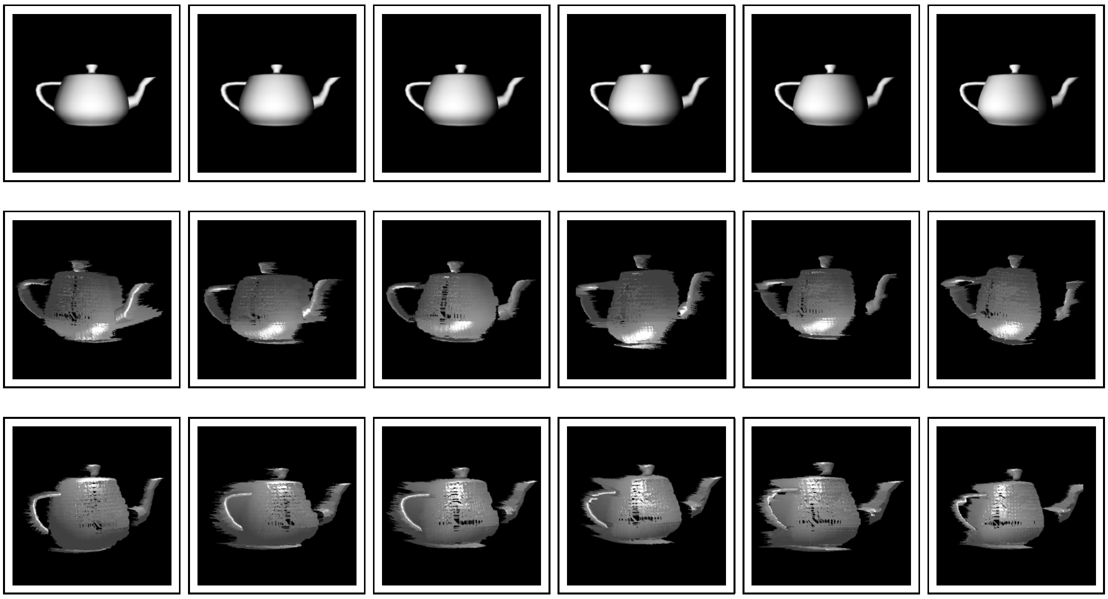


Figure 5.3: Effect of varying the light source direction on the shape-from-shading algorithm: Input images (top row); Views of the recovered surfaces (Bottom rows)

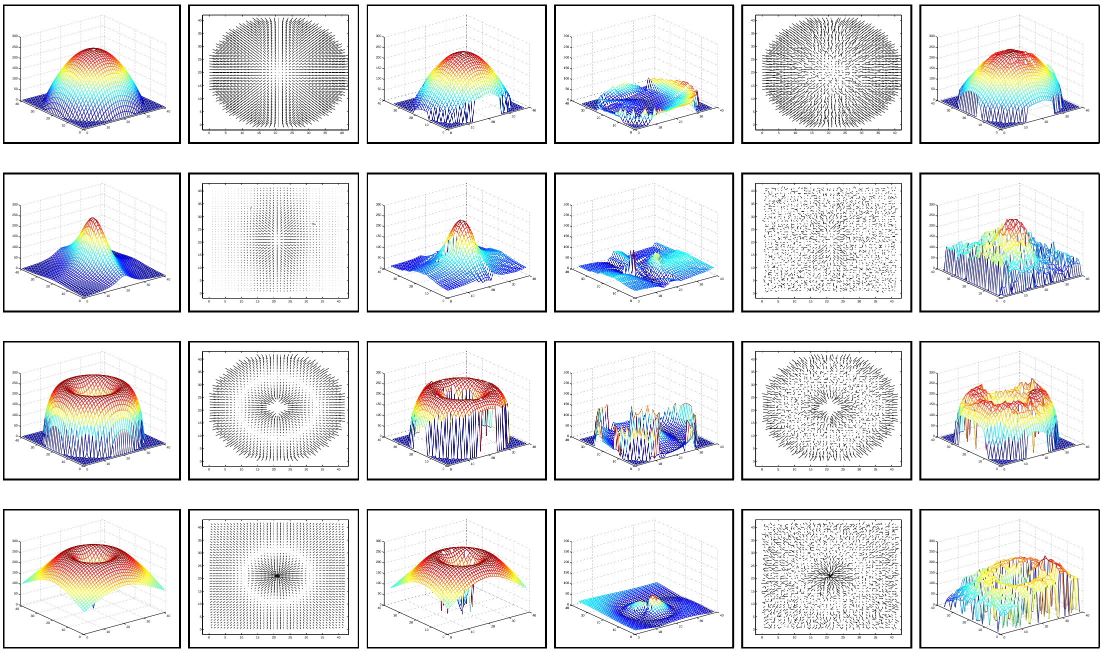


Figure 5.4: Surface reconstruction. From left-to-right the columns show a) the original surface, b) the needle map, c) the reconstructed surface, d) the height error, e) noise currupted needle-maps, f) surfaces reconstructed from the noisy needle-maps.

normal and the light source direction. This angle is equal to

$$\theta_i^{(k)} = \cos^{-1}\nu_i - \cos^{-1}\frac{\vec{N}_i^{(k)}\cdot\vec{s}}{||\vec{N}_j^{(k)}||\cdot||\vec{s}||}$$

This rotation takes place about the axis whose direction is given by the vector $[u,v,w]^T = \vec{N}_i^{(k)}\times\vec{\mathbf{L}}$. This rotation axis is perpendicular to both the light source direction and the off-cone normal. Hence, the rotation matrix is

$$\Phi^{(k)} = \begin{pmatrix} c+u^2c' & -ws+uvc' & vs+uwc' \\ ws+uvc' & c+v^2c' & -us+vwc' \\ -vs+uwc' & us+vwc' & c+w^2c' \end{pmatrix}$$

where $c=\cos\theta_i^{(k)}$, $c'=1-c$, and $s=\sin\theta_i^{(k)}$.

This simple approach has a number of shortcomings. For instance, it needs known light source direction and copes neither with changes in albedo nor the presence of specularities. Much of its success can be attributed to the use of the image gradient direction to initialise the positions of the surface normals on the irradiance cone. This initialisation has the effect of biassing the recovered needle-map in the favour of convex surfaces.

5.2 Affinity Matrix

The method outlined in the previous section provides us with estimates of the field of surface normals. Our aim is to reconstruct the surface using constraints on the surface normal directions provided by the local irradiance cones. To do this we use a graph-spectral method to locate a curvature minimising path through the field of surface normals.

Stated formally, our goal is the recovery of height information from the field of surface normals. The field is obtained by translating the surface normals from each point on a curved surface to a projection plane. From a computational standpoint the aim is to find a path on the projection plane along which simple trigonometry may be applied to increment the estimated height function. To be more formal suppose that the surface under study is $\mathcal{S}$ and that the field of surface normals is constructed on the plane Π. Our aim here is to find a curve Γ_Π across the plane Π that can be used as an integration path to reconstruct the height-function of the surface $\mathcal{S}$. The projection of the curve Γ_Π onto the surface $\mathcal{S}$ is denoted by $\Gamma_\mathcal{S}$. Further, suppose that $\kappa(l)$ is the sectional curvature of the curve $\Gamma_\mathcal{S}$ at the point q with parametric co-ordinate l. We seek the path $\Gamma_\mathcal{S}$ that minimises the total squared curvature

$$\mathcal{E}(\Gamma_\mathcal{S}) = \int_{\Gamma_\mathcal{S}} \kappa(l)^2 dl \tag{5.1}$$

Dropping the iteration dependant superscript used in the previous section, for the surface $\mathcal{S}$ sampled on the plane Π the field of unit surface normals consists of the set $\vec{N}_q$ where $q \in \Pi$. Accordingly, and following do Carmo (Do Carmo, 1976), we let $\Pi_q(\mathcal{S})$ represent the tangent plane to the surface $\mathcal{S}$ at the point q which belongs to the curve $\Gamma_\mathcal{S}$. To compute the sectional curvature $\kappa(l)$ we require the differential of the surface or Hessian matrix $d\vec{N}_q : \Pi_q(\mathcal{S}) \to \Pi_q(\mathcal{S})$. The maximum and minimum eigenvectors λ_1 and λ_2 of $d\vec{N}_q$

are the principal curvatures at the point q. The corresponding eigenvectors $\vec{e}_1 \in \Pi_q(\mathcal{S})$ and $\vec{e}_2 \in \Pi_q(\mathcal{S})$ form an orthogonal basis on the tangent plane $\Pi_q(\mathcal{S})$. At the point q the unit normal vector to the curve $\Gamma_{\mathcal{S}}$ is $\vec{n}$ and the unit tangent vector is $t_q \in \Pi_q(\mathcal{S})$. The sectional curvature of Γ at q is given by

$$\kappa(l) = \frac{(\vec{t}_q \cdot \vec{e}_1)^2(\lambda_1 - \lambda_2) + \lambda_2}{\vec{n} \cdot \vec{N}_q} \tag{5.2}$$

where $(\vec{t}_q \cdot \vec{e}_1)^2(\lambda_1 - \lambda_2) + \lambda_2$ is the normal curvature and $\alpha = \arccos \vec{n} \cdot \vec{N}_q$ is the angle between the curve normal and the surface normal.

In practice, we will be dealing with points which are positioned at discrete positions on the pixel lattice. Suppose that i and j are the pixel indices of neighbouring points sampled on the pixel lattice along the path $\Gamma_{\mathcal{S}}$. With this discrete notation, the cost associated with the path is given by

$$\mathcal{E}(\Gamma_{\mathcal{S}}) = \sum_{(i,j)\in\Gamma_{\mathcal{S}}} \mathcal{E}_{i,j} = \sum_{(i,j)\in\Gamma_{\mathcal{S}}} \kappa_{i,j}^2 l_{i,j} \tag{5.3}$$

where $\kappa_{i,j}$ is an estimate of the curvature based on the surface normal directions at the pixel locations i and j, and $l_{i,j}$ is the path distance between these points. The energy associated with the transition between sites i and j is $\mathcal{E}_{i,j} = \kappa_{i,j}^2 l_{i,j}$.

In order to compute the path curvature appearing in the expression for the transition energy, we make use of the surface normal directions. To commence, we note that

$$|\kappa_{i,j}| = \frac{1}{\mathcal{R}_{i,j}}$$

where $\mathcal{R}_{i,j}$ is the radius of the local circular approximation to the integration curve on the surface. Suppose that the surface normal directions at the pixel locations i and j are respectively $\vec{N}_i$ and $\vec{N}_j$. The approximating circle connects the points i and j, and has the path segment $l_{i,j}$ as the connecting chord. The change in direction of the radius vector of the circle is $\theta_{i,j} = \arccos \vec{N}_i \cdot \vec{N}_j$, and hence $\cos\theta_{i,j} = \vec{N}_i \cdot \vec{N}_j$. If the angle $\theta_{i,j}$ is small, then we can make the Maclaurin approximation

$$\cos\theta_{i,j} \simeq 1 - \frac{\theta_{i,j}^2}{2} = \vec{N}_i \cdot \vec{N}_j \tag{5.4}$$

Moreover, the small angle approximation to the radius of curvature of the circle is $\mathcal{R}_{i,j} = \frac{l_{i,j}}{\theta_{i,j}}$. Hence,

$$\kappa_{i,j}^2 = \frac{2(1 - \vec{N}_i \cdot \vec{N}_j)}{l_{i,j}^2} \tag{5.5}$$

The geometry outlined above is illustrated in Figure 5.1a.

In the next section of this chapter, we present a graph spectral technique for locating the integration path $\Gamma_{\mathcal{S}}$. The starting point for this analysis is a weight matrix that can be used to represent the affinity of pixels based on the curvature of the connecting path, or the difference in surface normal directions. To compute the elements of the affinity matrix, we associate with the pair of pixels a cost or energy that is equal to the product of

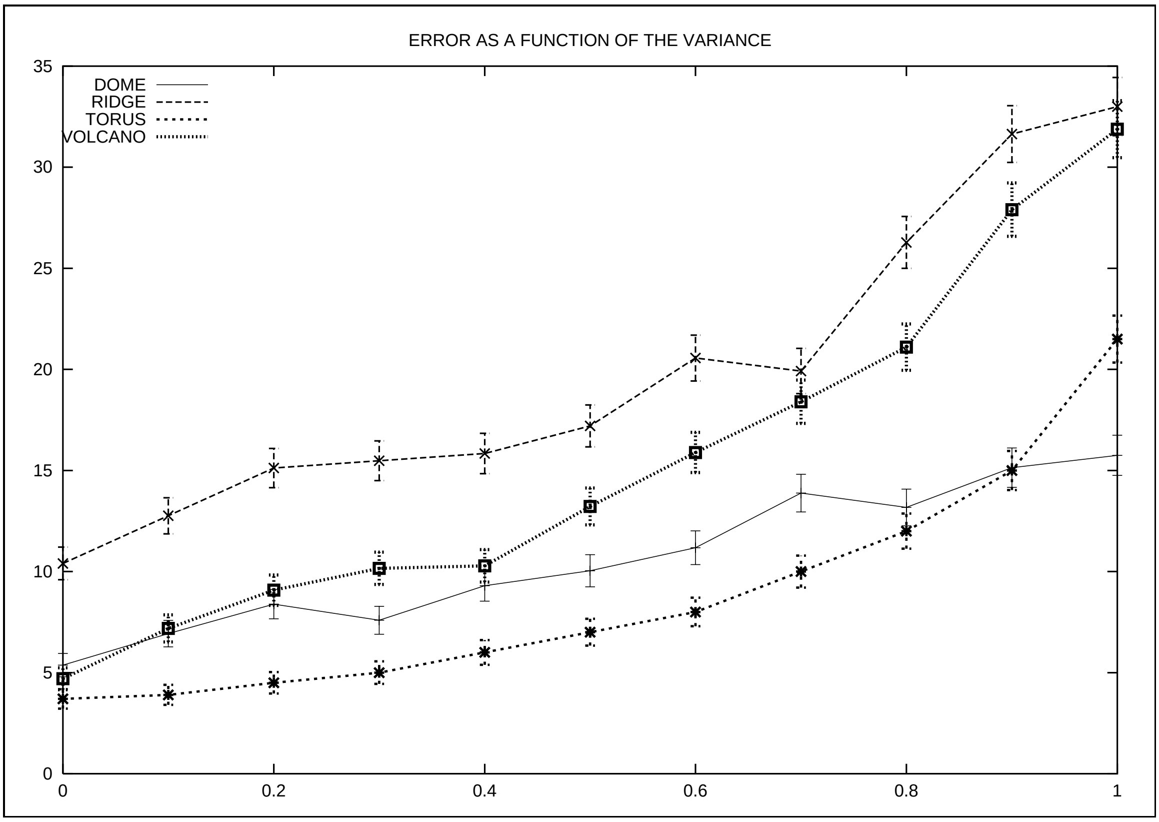


Figure 5.5: Surface reconstruction error as a function of the noise variance

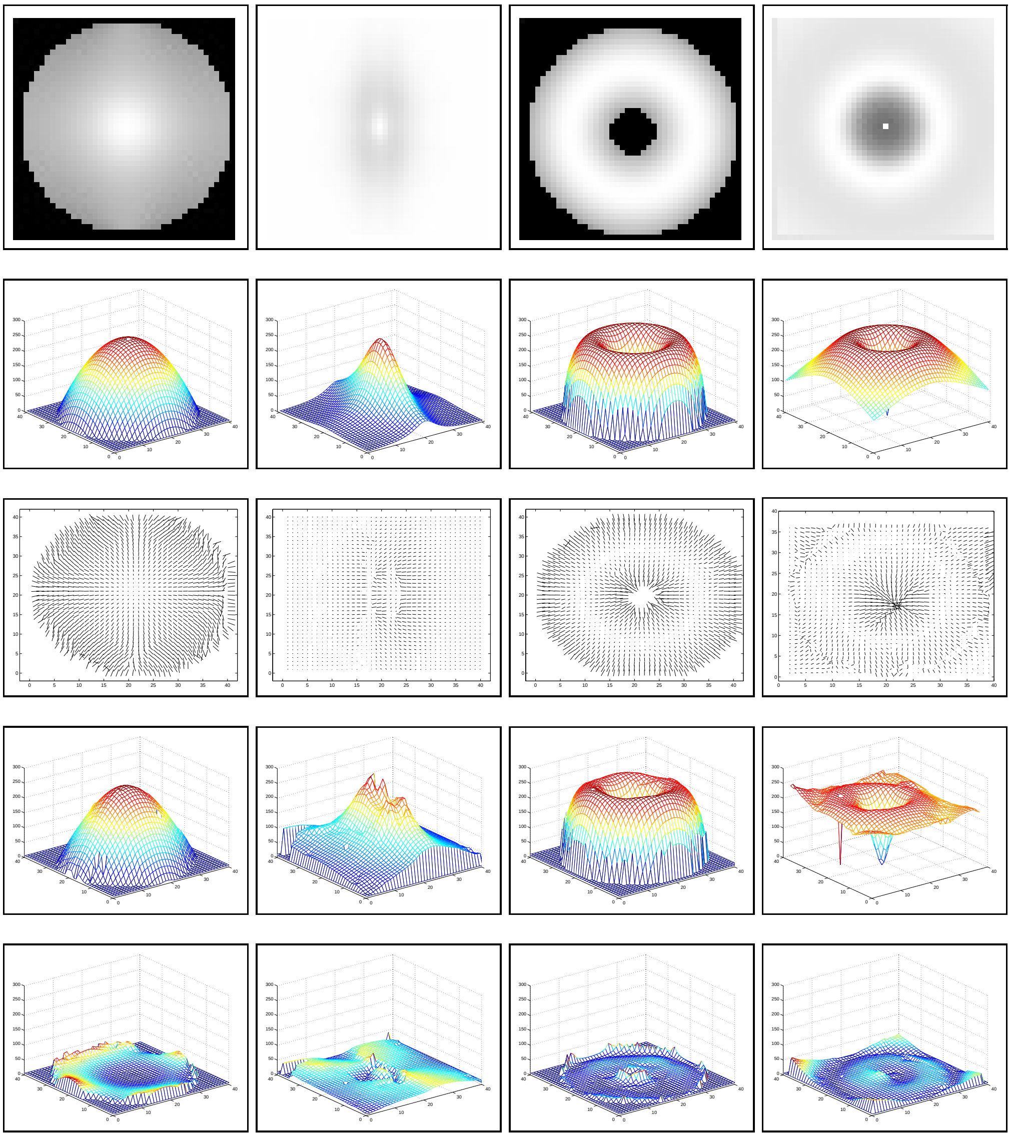


Figure 5.6: Shape-from-shading results on noise-free data: Initial images (top row); ground-truth surfaces (second row); needle-maps (third row); reconstructed surfaces (fourth row); absolute difference between ground truth and reconstructed surfaces

the distance between the sites and the squared sectional curvature of the connecting path, i.e. $\mathcal{E}_{i,j} = \hat{\kappa}_{i,j}^2 l_{i,j}$.

Using the approximation to the sectional curvature, we find that the cost associated with the step from the pixel i to the pixel j is

$$\mathcal{E}_{i,j} = \frac{2}{l_{i,j}}(1 - \vec{N}_i \cdot \vec{N}_j) \tag{5.6}$$

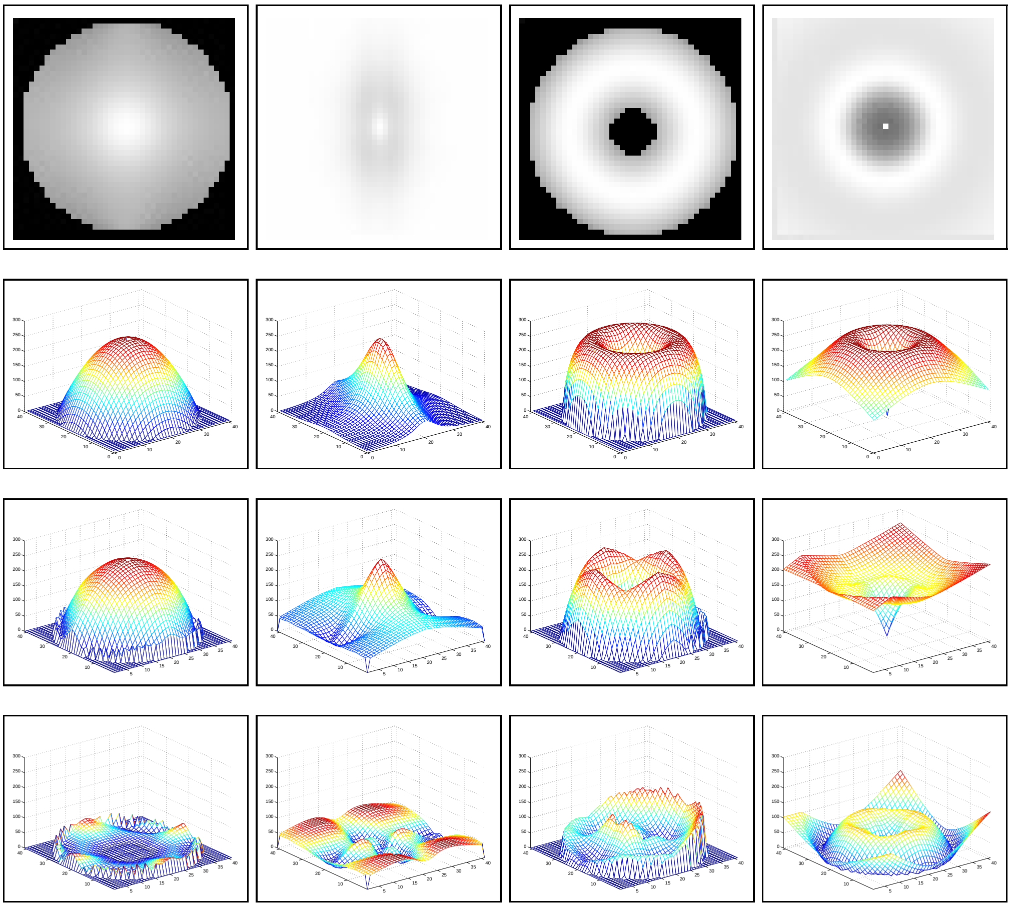


Figure 5.7: Results on noise-free image data for the algorithm of Bichsel and Pentland: Initial images (top row); ground-truth surfaces (second row); reconstructed surfaces (third row); absolute difference between ground truth and reconstructed surfaces (bottom row)

To pursue the graph-spectral analysis of the field of surface normals, we require a transition weight matrix which reflects the connectivity of the pixel lattice. For the pixels indexed i and j we define the transition weight matrix to have elements

$$W(i,j) = \begin{cases} \exp[-\beta \mathcal{E}_{i,j}] & \text{if } j \in \mathcal{N}_i \\ 0 & \text{otherwise} \end{cases} \tag{5.7}$$

where $\mathcal{N}_i$ is the set of pixels-neighbours of the pixel i and β is a constant. Hence, the curvature dependent weight is only non-zero if pixels abut one-another. Moreover, the weights are unity if the sectional curvature is zero and tend to zero as the curvature increases.

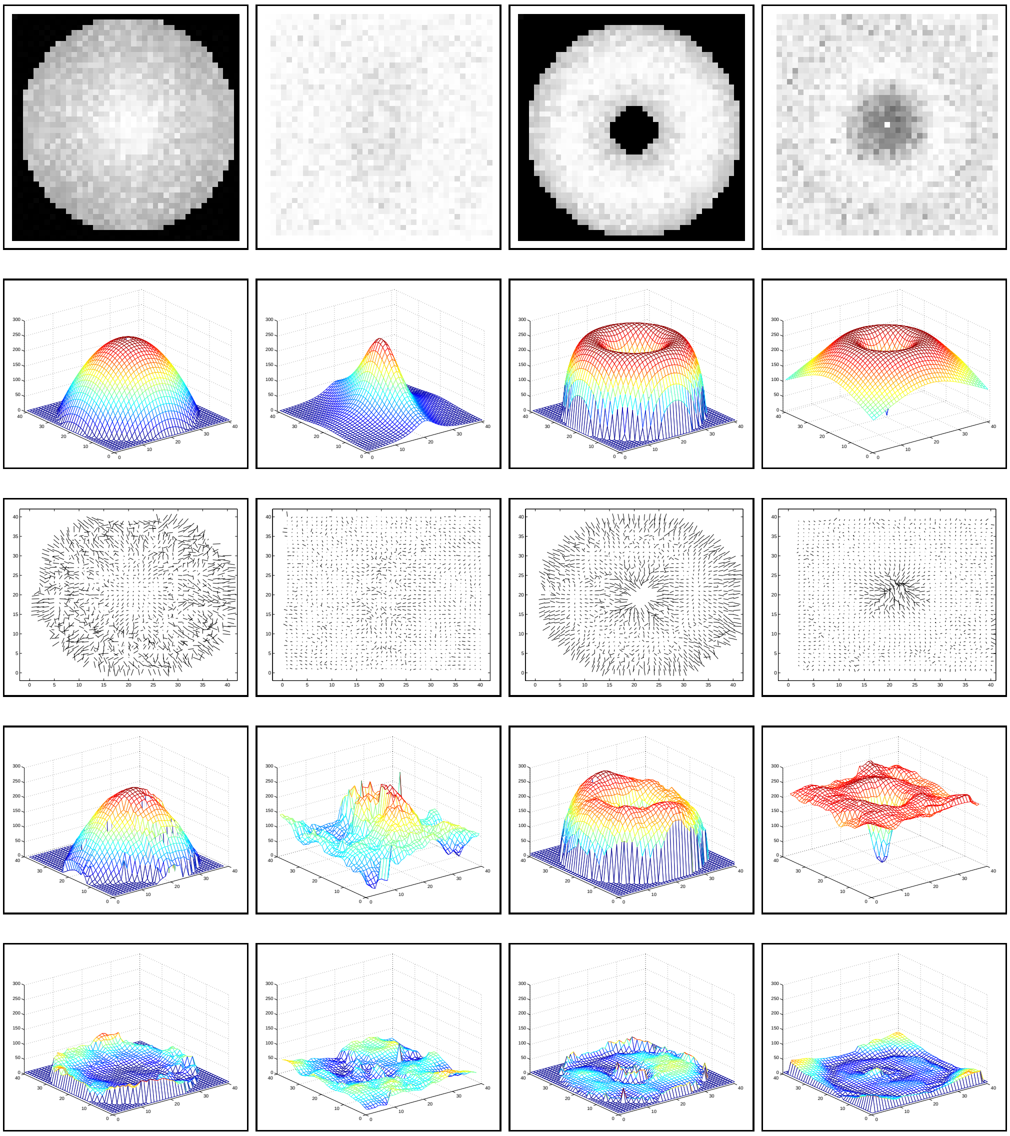


Figure 5.8: Shape-from-shading results on noisy image data: Initial images (top row); ground-truth surfaces (second row); needle-maps (third row); reconstructed surfaces (fourth row); absolute difference between ground truth and reconstructed surfaces

5.3 Graph Seriation

In the previous section, we showed how the change in surface normal directions could be used to compute the elements of a transition weight matrix. In this section, we describe how the leading eigenvector of the transition weight matrix can be used to locate an integration path that maximises the total curvature weight. We pose this as a process of graph-spectral seriation.

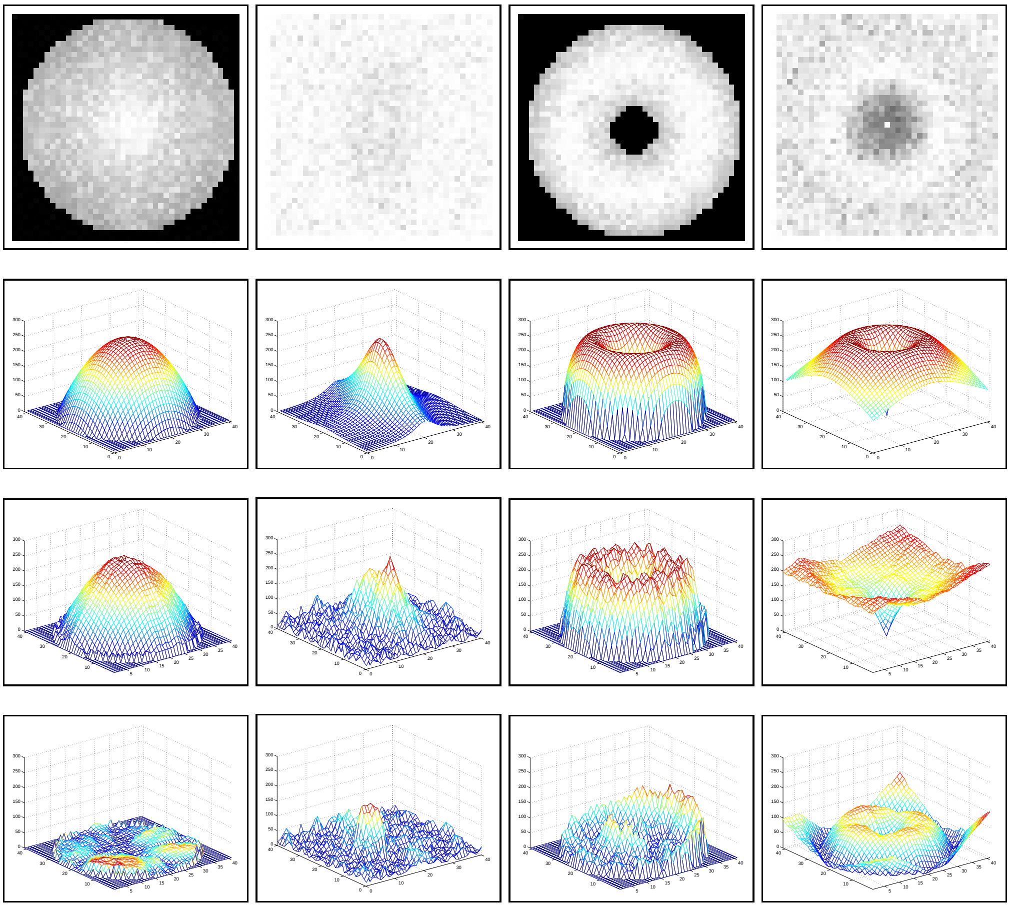


Figure 5.9: Results on noisy image data for the algorithm of Bichsel and Pentland: Input images (top row); ground-truth surfaces (second row); reconstructed surfaces (third row); absolute difference between ground truth and reconstructed surfaces (bottom row)

To commence, we pose the problem in a graph-based setting. The set of pixel sites can be viewed as a weighted graph $G = (V, E, W)$ with index-set V, edge-set $E = \{(i, j)|(i, j) \in V \times V, i \neq j\}$ and weight function $W : E \to [0, 1]$. Here, $i \in V$ refers to the index of the pixel at the ith position on the path.

The seriation problem as stated by Atkins, Boman and Hendrickson (Atkins et al., 1998) is as follows. The aim is to find a path sequence for the nodes in the graph using a permutation ϱ. Hence the permutation of the pixel index gives the order of the pixel in the path. The sequence is such that the edge weight decreases as the path is traversed. Hence, if $\varrho(k) < \varrho(l) < \varrho(i)$, then $W(k, l) > W(k, i)$ and $W(l, m) > W(k, i)$. This behaviour can be captured using the penalty function

$$g(\varrho) = \sum_{k=1}^{|V|} \sum_{l=1}^{|V|} W(k,l)(\varrho(k) - \varrho(l))^2$$

By minimising $g(\varrho)$ it is possible to find the permutation that minimises the difference in edge weight between adjacent nodes in the path, and this in turn sorts the edge weights into magnitude order. Unfortunately, minimising $g(\varrho)$ is potentially NP complete due to the combinatorial nature of the discrete permutation ϱ. To overcome this problem, a relaxed solution is sought that approximates the structure of $g(\varrho)$ using a vector $\vec{x} = [x_1, x_2, \ldots, x_{|V|}]^T$ of continuous variables x_i. When the coefficients of $\vec{x}$ are sorted in decreasing order, then the rank of the component x_i is the same as that of the permutation element $\varrho(i)$ Hence, the penalty function considered is

$$\hat{g}(\vec{x}) = \sum_{i=1}^{|V|} \sum_{l=1}^{|V|} W(i,l)(x_i - x_l)^2$$

The minimiser of the penalty function above is not unique, this is due to the fact that the value of $\hat{g}(\vec{x})$ does not change if a constant amount is added to each of the components x_i. Hence, the minimisation problem must be subject to constraints on the components of the vector $\vec{x}$. The constraints are that

$$\sum_{i=1}^{|V|} x_i^2 = 1 \quad \text{and} \quad \sum_{i=1}^{|V|} x_i = 0 \tag{5.8}$$

Atkins, Bowman and Atkinson show that the solution to this relaxed problem may be obtained from the Laplacian matrix. Let D be the diagonal matrix with elements $D(i,i) = \sum_{j=1}^{|V|} W(i,j)$ equal to the total weight of the edges connected to the node i. The Laplacian matrix is $L = D - A$. If $\mathbf{e} = [1, 1, 1..., 1]^T$ is the all-ones vector, then the solution to the minimisation problem is the vector

$$\vec{x} = \arg \min_{\vec{x}_*^T \cdot \vec{e} = 0, \vec{x}_*^T \vec{x}_* = 1} \vec{x}_*^T L \vec{x}_* = \arg \min_{\vec{x}_*^T \cdot \vec{e} = 0, \vec{x}_*^T \vec{x}_* = 1} \sum_{i>l} W(i,l)(x_{*i} - x_{*l})^2$$

When W is positive definite, then the solution is the Fiedler vector, i.e. the vector associated with the smallest non-zero eigenvalue of L. In fact, the associated eigenvalue minimises the Rayleigh quotient

$$\lambda = \arg\min_{x_*} \frac{\vec{x}_*^T L \vec{x}_*}{\vec{x}_*^T \vec{x}_*}$$

Unfortunately, the procedure described above does not meet our requirements since it will not result in a path that minimises the curvature. To overcome this problem, we can turn instead to maximising the quadratic quantity

$$\hat{g}_E(\vec{x}) = \vec{x}^T W \vec{x} = \sum_{i=1}^{|V|} \sum_{j=1}^{|V|} W(i,j) x_i x_j \tag{5.9}$$

As noted earlier, Sarkar and Boyer (Sarkar and Boyer, 1998) have shown that the maximiser of $\hat{g}_E(\vec{x})$ is the leading eigenvector ϕ_* of the matrix W. The leading eigenvector satisfies the equation

$$\lambda_* \phi_* = W \phi_* \tag{5.10}$$

where λ_* is the leading eigenvalue of W. For the seriation problem, $\hat{g}_E(\vec{x})$ is the sum of curvature weights associated with the path. Moreover, it results in an ordering in which the nodes of greatest weight (i.e. smallest curvature) are placed first in the path.

It is worth stressing that, if we visit the nodes of the graph in the order defined by the magnitudes of the co-efficients of the leading eigenvector of the matrix W, then the path may not be edge-connected. Hence, we need a means of placing the nodes in a serial order in which edge constraints are preserved using the elements of the leading eigenvector ϕ_*. To do this, we commence from the node associated with the largest component of ϕ_*. We then sort the elements of the leading eigenvector such that they are both in the decreasing magnitude order of the co-efficients of the eigenvector, and satisfy neighbourhood connectivity constraints on the pixel lattice. The procedure is a recursive one that proceeds as follows. At each iteration, we maintain a list of sites visited. At iteration k let the list of sites be denoted by $\overline{C}_k$. Initially, $\overline{C}_1 = j_1$ where $j_1 = \arg\max_j \phi_*(j)$, i.e. j_1 is the com-

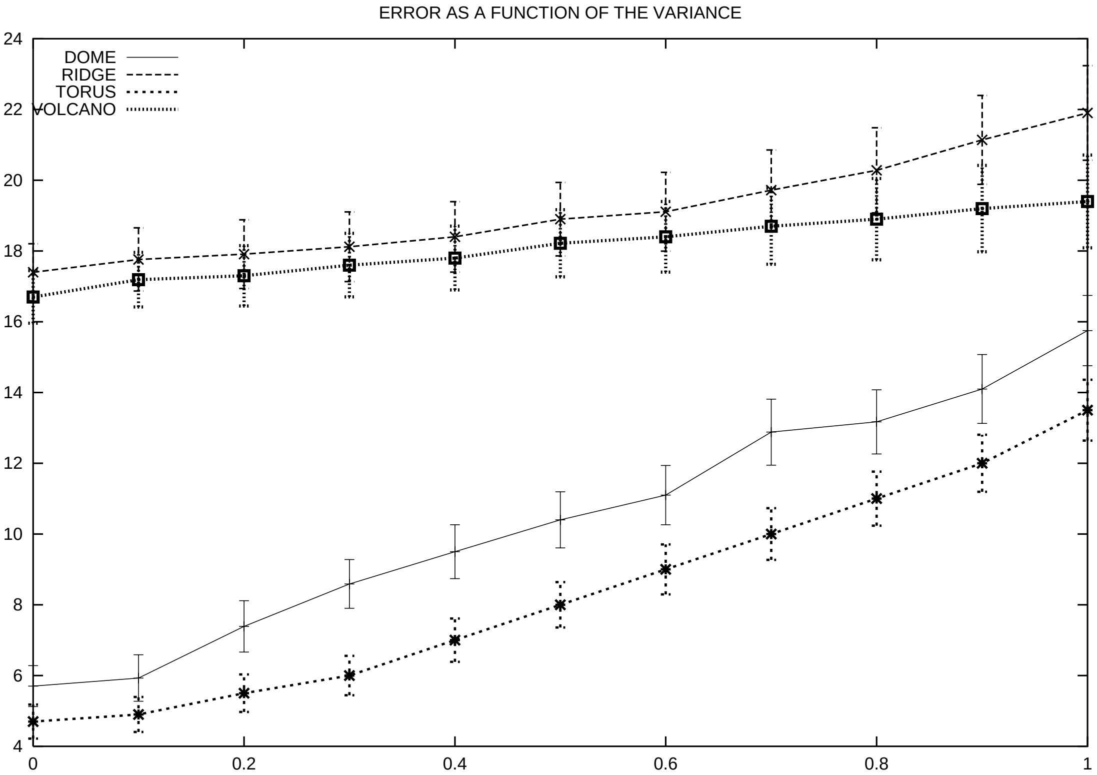


Figure 5.10: Plot of the error percentage as a function of the variance for the four synthetic basic shapes

ponent of ϕ_* with the largest magnitude. Next, we search through the set of 8-neighbours of j_1 to find the pixel associated with the largest remaining component of ϕ_*. If $\mathcal{N}_{j_1}$ is the set of 8-neighbours of j_1, the second element in the list is $j_1 = \arg\max_{l \in \mathcal{N}_{j_1}} \phi_*(l)$. The pixel index j_1 is appended to the list of sites visited and the result is $\overline{C}_2$. In the kth (general) step of the algorithm we are at the pixel site indexed j_k and the list of sites visited by the path so far is $\overline{C}_k$. We search through those 8-neighbours of j_k that have not already been traversed by the path. The set of pixel sites is $C_k = \{l | l \in \mathcal{N}_{j_k} \wedge l \notin \overline{C}_k\}$. The next site to be appended to the path list is therefore $j_{k+1} = \arg\max_{l \in C_k} \phi_*(l)$. This process is repeated until no further moves can be made. This occurs when $C_k = \emptyset$ and we denote the index of the termination of the path by T. The integration path $\Gamma_{\mathcal{S}}$ is given by the list of pixel sites $\overline{C}_T$.

To conclude this section, we return to the cost function underpinning the recovery of the integration path, to make the role of the optimisation process more explicit. When po-

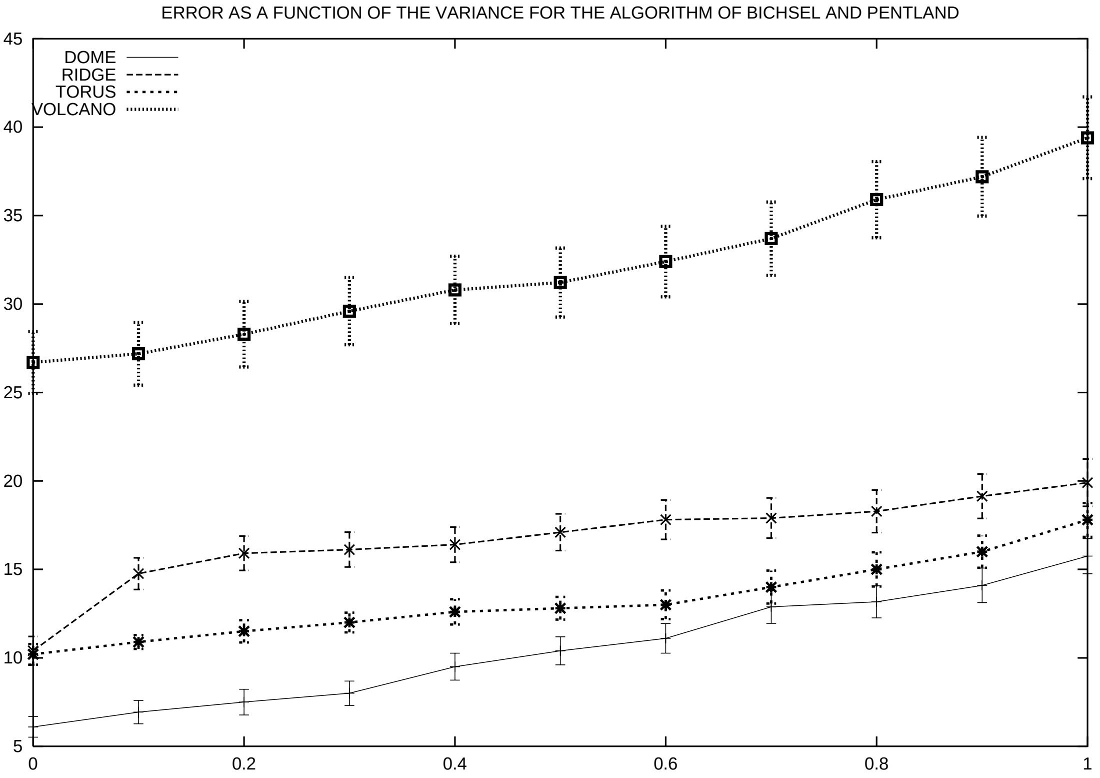


Figure 5.11: Plot of the error percentage as a function of the variance for the four synthetic basic shapes when processed using the algorithm of Bichsel and Pentland

sitioned at the site j_k, the next pixel visited satisfies the condition $j_{k+1} = \arg\max_{l \in C_k} \phi_*(l)$. Since ϕ_* is the leading eigenvector of W it satisfies the equation $W\phi_* = \lambda_* \phi_*$, As a result

$$j_{k+1} = \arg\max_{l \in C_k} \sum_{m \in \mathcal{N}_l} W(l,m)\phi_*(m)$$

Substituting for the definition of the weight in terms of the surface normals from using Equations 5.6 and 5.7,

$$j_{k+1} = \arg\max_{l \in C_k} \sum_{m \in \mathcal{N}_l} \left\{ \exp\left[-2\beta(1 - \vec{N}_m \cdot \vec{N}_l) \right] \right\} \phi_*(m)$$

When the angle between the surface normals is small, then we can write

$$j_{k+1} = \arg\max_{l \in C_k} \sum_{m \in \mathcal{N}_l} \left\{ 1 - 2\beta(1 - \vec{N}_m \cdot \vec{N}_l) \right\} \phi_*(m)$$

Hence, the path minimises the change in surface normal direction, and hence curvature, on the image plane Π.

5.4 Extracting Patches

In practice the surface under study may have a patch structure. The patches may be identified by finding the blocks of the weight matrix induced under a permutation of the nodes. We find the blocks by computing the leading eigenvector of the weight matrix. The algorithm proceeds in an iterative fashion. The leading eigenvector of the current weight matrix represents a patch. The nodes with non-zero components in the leading eigenvector belong to the patch. The nodes are identified, and are then removed from further consideration by nulling their associated elements in the weight matrix. This process is repeated until all the principal patches are identified. This is the case when only an insignificant number of unassigned and unconnected nodes remain.

We commence by constructing the thresholded weight matrix W whose elements are defined as follows

$$A(i,j) = \begin{cases} 0 & \text{if } W(i,j) << 1 \\ P(i,j) & \text{otherwise} \end{cases} \tag{5.11}$$

The matrix A is simply a thresholded version of the transition weight matrix W in which the vanishingly small elements are set to zero.

Our aim is identify groups of surface normals from a potentially noisy or ambiguous adjacency matrix A which correspond to surface patches. Stated formally, suppose that in an image with an adjacency matrix A there are m disjoint patches. Each such group should appear as a sub-block of the matrix A. However, as a consequence of noise or errors in the shape-from-shading method which delivers the field of surface normals, these distinct groups or patches may be merged together. In other words, their corresponding sub-blocks are no longer disjoint.

Suppose that there are m distinct surface patches, each associated with an adjacency matrix $B^{(i)}$ where i is the patch index. If C represents a noise matrix, then the relationship between the observed transition weight matrix A and the underlying block-structured transition weight matrix is $A = B + C$ where $B = \mathcal{P} B_D \mathcal{P}^T$, $\mathcal{P}$ is a permutation matrix and

$$B_D = \begin{pmatrix} B^{(1)} & 0 & \dots & 0 \\ 0 & B^{(2)} & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \dots & 0 & B^{(m)} \end{pmatrix} \tag{5.12}$$

is a block diagonal matrix in which $B^{(i)}$ is the sub-block corresponding to the patch indexed i.

To recover the matrix B, we turn to the eigenvector expansion of the matrix A and write

$$A = \phi_* \phi_*^T + \sum_{i=2}^{|V|} \lambda_i \vec{\phi}_i \phi_i^T \tag{5.13}$$

where the leading eigenvalue is unity i.e. $\lambda_1 = 1$, ϕ_* is the leading eigenvector and the eigenvectors are normalised to be of unit length, i.e. $|\phi_i| = 1$. To identify the patches, we use the following iterative procedure. We initialise the algorithm by letting $A^{(1)} = A$. Further suppose that $\phi_*^{(1)}$ is the leading eigenvector of $A^{(1)}$. The matrix $B^{(1)} = \phi_*^{(1)} \phi_*^{(1)T}$ represents the first block of A, i.e. the most significant surface patch. The nodes with non-zero entries belong to the patch. These nodes may be identified and removed from further consideration. To do this we compute the residual transition weight matrix $A^{(2)} = A^{(1)} - B^{(1)}$ in which the elements of the first patch are nulled. The leading eigenvector $\phi_*^{(2)}$ of the residual transition weight matrix $A^{(2)}$ is used to compute the second block $B^{(2)} = \phi_*^{(2)} \phi_*^{(2)T}$. The process is repeated iteratively to identify all of the principal blocks of A. At iteration p, $\phi_*^{(p)}$ is the leading eigenvector of the residual transition weight matrix $A^{(p)}$, and the p^{th} block is $B^{(p)} = \phi_*^{(p)} \phi_*^{(p)T}$. The patch indexed n is the set of nodes for which the components of the leading eigenvector $\phi_*^{(p)}$ are non-zero. Hence, the index-set for the p^{th} patch is $\mathcal{S}_p = \{i | \phi_*^{(p)}(i) \neq 0\}$. It is important to stress that the patches are non-overlapping, i.e. the inner product of the block eigenvectors for different patches is zero $\phi_*^{(p)} . \phi_*^{(l)} = 0$, where $p \neq l$.

The process of patch extraction is iterated until the associated regions are of insignificant size. In practice, this is the case when the number of pixels associated with a path is less than 5.

5.5 Height Recovery

Our surface height recovery algorithm proceeds along the sequence of pixel sites defined by the order of the co-efficients of the scaled leading eigenvector associated with the separate patches. For the p^{th} patch, the path is $\Gamma_p = (1^p, 2^p, 3^p, ...)$ where the order is established using the method outlined in Section 5.3. As we move from pixel-site to pixel-site defined by this path we increment the surface height-function. In this section, we describe the trigonometry of the height incrementation process.

At step n of the algorithm, we make a transition from the pixel with path-index j_{n-1}^{p} to the pixel with path-index j_n^p. The distance between the pixel-centres associated with this transition is

$$r_n^p = \sqrt{(x_{n^p} - x_{n-1^p})^2 + (y_{n^p} - y_{n-1^p})^2} \tag{5.14}$$

This distance, together with the surface normals

$$\vec{N}_{n^p} = [N_{n^p}(x), N_{n^p}(y), N_{n^p}(z)]^T$$

and

$$\vec{N}_{n-1^p} = [N_{n-1^p}(x), N_{n-1^p}(y), N_{n-1^p}(z)]^T$$

at the two pixel-sites may be used to compute the change in surface height associated with the transition. Assuming that the two pixel sites are connected by a plane, whose normal is $\vec{N}_{n^p}$, we can write $N_{n^p}(x)(x - x_{n^p}) + N_{n^p}(y)(y - y_{n^p}) + N_{n^p}(z)(z - z_{n^p}) = 0$. As a result, the height increment is given by

$$h_n^p = \frac{r_n^p}{2}\left((x_{n-1^p} - x_{n^p})\left(\frac{N_{n-1^p}(x)}{N_{n-1^p}(z)} + \frac{N_{n^p}(x)}{N_{n^p}(z)}\right) + (y_{n-1^p} - y_{n^p})\left(\frac{N_{n-1^p}(y)}{N_{n-1^p}(z)} + \frac{N_{n^p}(y)}{N_{n^p}(z)}\right)\right) \tag{5.15}$$

If the height-function is initialised by setting $z_{j_0^p} = 0$, then the centre-height for the pixel with path-index $n + 1^p$ is

$$z_{n+1^p} = z_{n^p} + h_n^p \tag{5.16}$$

Once the surface normals that belong to the individual patches have been integrated together, then we merge them together to form a global surface. Suppose that $\mathcal{S}_p$ is the integrated surface for the p^{th} patch. We compute the mean height for the pixels belonging to this boundary. We merge the patches together by ensuring that abutting patches have the same mean boundary height.

The geometry of this procedure is illustrated in Figure 5.1b.

5.6 Region Quadric Patch Fitting and Surface Normal Adjustment

Once the height values are available for each pixel site in a patch, then we perform smoothing. We do this by fitting a local bi-quadric function to the height data for the patch sites. To do this we employ a simple least-squares fitting method.

Suppose that the pixels belonging to the patch indexed p are denoted by the set $\mathcal{S}_p$. We aim to fit the bi-quadric

$$f_p(x_i, y_i) = a_{1,p} + a_{2,p}x + a_{3,k}y_i + a_{4,p}x_i^2 + a_{5,p}x_iy_i + a_{6,p}y_i^2 \tag{5.17}$$

to the height data for the sites in the patch. Let

$$M_i = (1, x_i, y_i, x_i^2, x_iy_i, y_i^2)^T$$

be a vector of co-ordinate moments and let $H_p = (a_{1,p}, ..., a_{6,p})^T$ be a vector of parameters. The least-squares parameter vector for the quadric patch satisfies the condition

$$H_p^* = \arg\min_{H_p} \sum_{i \in \mathcal{S}_p} \left[z_i - H_p^T M_i \right]^2 \tag{5.18}$$

In matrix form the solution vector is given by

$$H_p^* = (F_p^T F_p)^{-1} F_p^T \tag{5.19}$$

where

$$F_p = \sum_{i \in \mathcal{S}_p} z_i M_i M_i^T \tag{5.20}$$

The surface fitting process can be viewed as one of smoothing over the extent of the patches defined by the leading eigenvectors of the weight matrix. As well as smoothing the local height estimates, the process also implicitly smooths the surface normal directions, since these can be computed from the gradient of the fitted local quadric patch. The estimate of the unit surface normal at the point (x_i, y_i) on the patch p is

$$\vec{N}_i = \frac{1}{R_i} \left[\frac{\partial f_p(x_i, y_i)}{\partial x_i}, \frac{\partial f_p(x_i, y_i)}{\partial y_i}, 1 \right]^T = \frac{1}{R_i} \left[a_{2,p} + 2a_{4,p} x_i + a_{5,p} y_i, a_{3,p} + a_{5,p} x_i + 2a_{6,p} y_i, 1 \right]^T \tag{5.21}$$

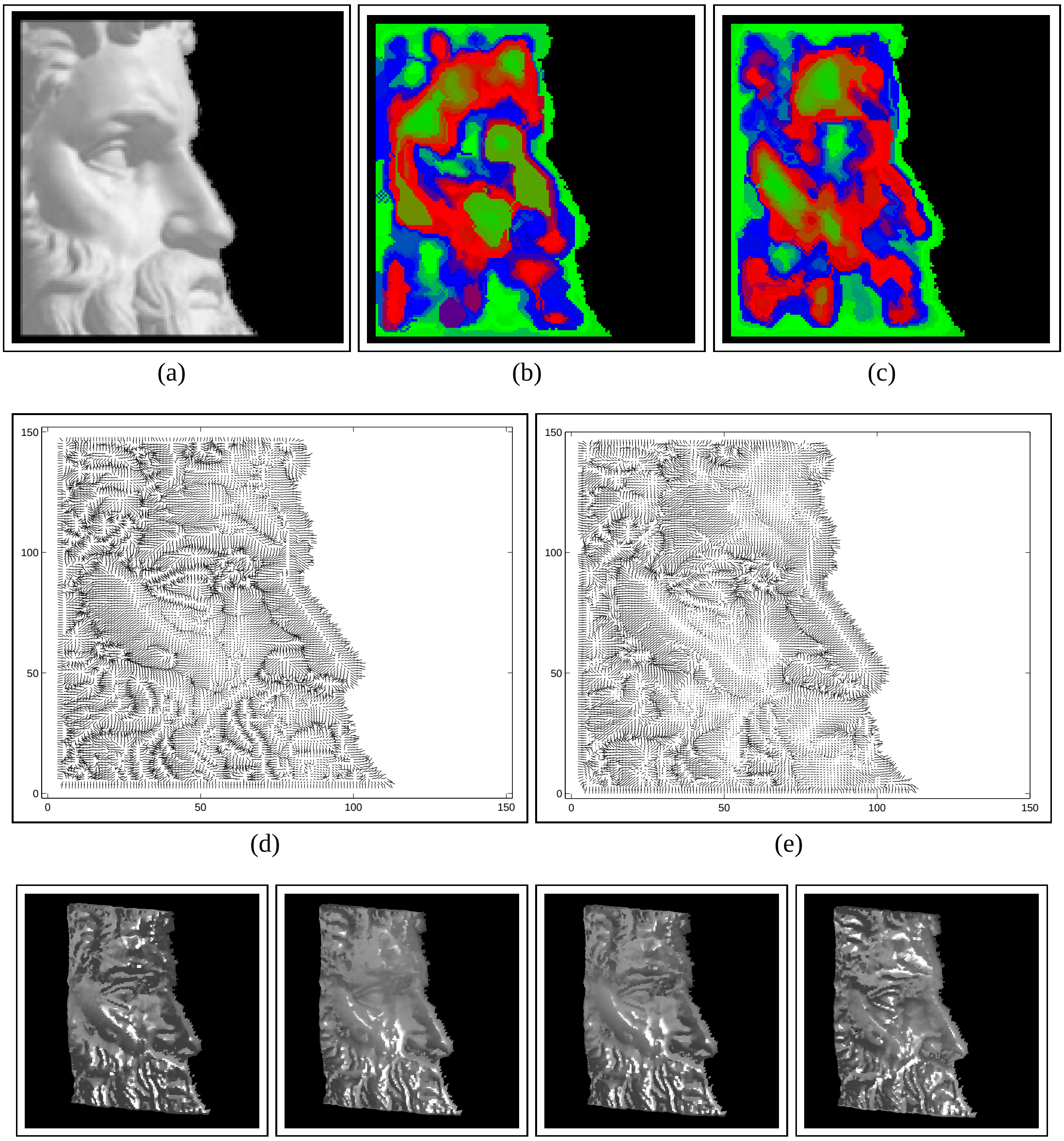
where $R_i = \sqrt{\left(1 + (\frac{\partial f_p(x_i,y_i)}{\partial x_i}\right)^2 + \left(\frac{\partial f_p(x_i,y_i)}{\partial y_i}\right)^2}$.

It is interesting to note that since

$$\frac{\partial^2 f_p(x_i, y_i)}{\partial x_i \partial y_i} = \frac{\partial^2 f_p(x_i, y_i)}{\partial y_i \partial x_i} = a_{5,p} \tag{5.22}$$

the field of smoothed surface normals satisfy the integrability constraint.

To update the surface normal direction so that they satisfy the image radiance equation, we rotate them using the matrix $\Phi^(k)$ as described in Section 5.1. The updated surface normals are no longer guaranteed to satisfy the integrability constraint, but fall on the closest location on the appropriate radiance cone.



(a) (b) (c)

(d) (e)

(f) (g) (h) (i)

Figure 5.12: Results of the algorithm on the image of Michelangelo's Moses

5.7 Algorithm Description

Having described the different steps of the surface reconstruction process, in this section we summarise how the steps are combined together in an algorithm. The sequence of processing steps is as follows:

- Step 0: The surface normals are placed in their initial positions on the irradiance cones. To do this we align them in the directions of the image gradient. The gradient is computed by first smoothing the grey-scale image by fitting local quadric patches to the raw image intensity. The smoothed image gradient is found from the derivatives of the fitted patch.

- Step 1: From the initial field of surface normals, we compute the sectional curvatures and hence the transition weight matrix. The blocks of the matrix are surface patches. The leading eigenvector of the transition matrix for each block is the patch integration path. Using the patch integration paths, we recover estimates of the surface height.

- Step 2: For the sites in each patch, we fit a quadric patch to the available height estimates. The fitted surface patches are used to compute an estimate of the surface gradient. At each location, the gradient estimate is used to adjust the position of the surface normals on their respective irradiance cones.

- Iteration: Steps 1 and 2 are iterated until a stable set of surface height estimates are located.

5.8 Algorithm Behaviour

In this section, we illustrate, making use of objects with known surface topology, the impact of changes in surface curvature on the minimum curvature path. We also provide evidence regarding the stability of the recovered surface.

5.8.1 Graph-spectral Surface Reconstruction

We commence by illustrating the effect of changes in the surface curvature on the surface reconstruction process. Recall that the idea underpinning the surface reconstruction process is that of recovering a minimum curvature path across the surface. As a result, the understanding of the impact of curvature change on the algorithm becomes of capital importance to comprehend the relation between the surface curvature and the trajectory of the recovered path.

In the left-hand column of Figure 5.2, we show the images of the surfaces under study. These are a sphere and a cylinder. In the second column, we show the plot of the field of surface normals for each of the two images. From these plots, is evident that, despite being clearly defined, the principal curvature directions are very different for each of the two surfaces under study. For instance, in the case of the cylinder, the minimum curvature directions are parallel to the image column directions. For the sphere, the minimum curvature directions are described by the set of circles on the image plane whose trace comprises those pixels with common image brightness. The minimum curvature paths for the cylinder and the sphere, as recovered by our algorithm, are shown in the third column. Here, we have marked with a black circle the starting point of the path. We have used an asterisk to denote the ending point. From the plots, we conclude that the trajectories described by the recovered path are in accordance with the minimum curvature directions. A sample view for each of the recovered surfaces is shown in the left-hand column of Figure 5.2.

5.8.2 Graph-spectral Shape-from-shading Algorithm

Next, we turn our attention to the stability of the recovered surface. To this end, we have generated a set of 6 synthetic images of a Lambertian teapot illuminated with a single light source positioned in the direction $\vec{s}_D = [\sin(\theta_s), 0, \cos(\theta_s)]^T$. For our experiments, we have varied the angular variable θ_s in increments of 6^o between -30^o and 0^o.

In the top row of Figure 5.3, we show the images in our dataset. Here, we have ordered the images, from left-to-right, in decreasing θ_s. In the remaining rows, we show two views of the recovered surfaces for each of the input images. The views corresponding to different illuminant directions are in good agreement with one another. This is an important observation since it indicates that the surface is stable to variations in the light source direction. Furthermore, despite some errors across the surface boundaries, the shape of the teapot is well recovered.

5.9 Performance Analysis

In this section we provide a sensitivity study on synthetic data. To this end we have generated synthetic surfaces. These surfaces are a torus, a volcano, a dome and a ridge. Our performance analysis is divided in two parts. The first of these is a sensitivity study aimed to evaluate the accuracy of the surface reconstruction process comprised by our shape-from-shading algorithm. The second part is a sensitivity study of the shape-from-shading algorithm as a whole. The aims in doing these two studies are twofold. Firstly Secondly, we aim to understand better the relation between the spectral reconstruction process and the shape-from-shading algorithm.

5.9.1 Graph-spectral Surface Reconstruction

We commence our performance analysis evaluating the accuracy of the surface reconstruction process comprised by our spectral shape-from-shading algorithm. From the closed form of the surfaces mentioned above, we have computed the field of surface normal directions. We have then applied the graph-spectral surface reconstruction method to the 2D field of surface normals to recover an estimate of the surface height.

In Figure 5.4 we show the results obtained for a series of different surfaces. From top-to-bottom the surfaces studied are a dome, a sharp ridge, a torus and a volcano. In the left-hand column we show the original synthetic surface. The second column shows the fields of surface normals. The third column shows the reconstructed surfaces, while the fourth column the absolute error between the ground-truth and reconstructed surface height. In all four cases the surface reconstructions are qualitatively good. For the dome the height errors are greater at the edges of the surface where the slope is largest. In the case of the ridge, there are errors at the crest. For the volcano, there are some problems with the recovery of the correct depth of the “caldera”, i.e. the depression in the centre. For the reconstructed surfaces, the mean-squared errors are 5.6% for the dome, 10.8% for the ridge, 7.8% for the torus and 4.7% for the volcano. Hence, the method seems to have greater difficulty for surfaces containing sharp creases.

We have repeated these experiments under conditions of controlled noise. To do this we have added random measurement errors to the surface normals by randomly sampling error vectors from a circularly symmetric 2D Gaussian distribution with zero mean and known variance. In the fifth column of Figure 5.4, we show the noise-corrupted field of surface normals. In the seventh column, we show the reconstructed height-function obtained from the noisy surface normals. In the case of all four surfaces, the gross structure is maintained, and the method appears robust to the added noise. Our graph-spectral method simply transfers errors in surface normal direction into errors in height, without producing structural noise artefacts. However, there are some large errors on the surface which may be attributed to poor patch alignment.

To investigate the effect of noise further, we plot the mean-squared error for the reconstructed surface height as a function of the standard deviation of the added Gaussian noise in Figure 5.5. From the plots for the different surfaces shown in Figure 5.4, it is clear that the mean-squared error grows slowly with increasing noise standard deviation. In the worst case, i.e. when the variance is equal to unity, the mean absolute difference between the ground truth surface normal and the noisy surface normal direction is 48.9 degrees.

5.9.2 Graph-spectral Shape-from-shading Algorithm

The aim here is to determine the accuracy of the surface recovered by the shape-from-shading algorithm when the image is used as input instead of the ideal field of surface normals. To this end we have selected a light source direction and the surfaces mentioned at the beginning of this section have been rendered using the Lambertian reflectance process outlined in Section 5.1. We have then applied the graph-spectral shape-from-shading method to the resulting synthetic images. We confront the resulting height estimates with the height data for the original surfaces. We also provide a comparative study using the shape-from-shading height recovery method of Bichsel and Pentland (Bichsel and Pentland, 1992) as an alternative to our graph-spectral shape-from-shading algorithm. We

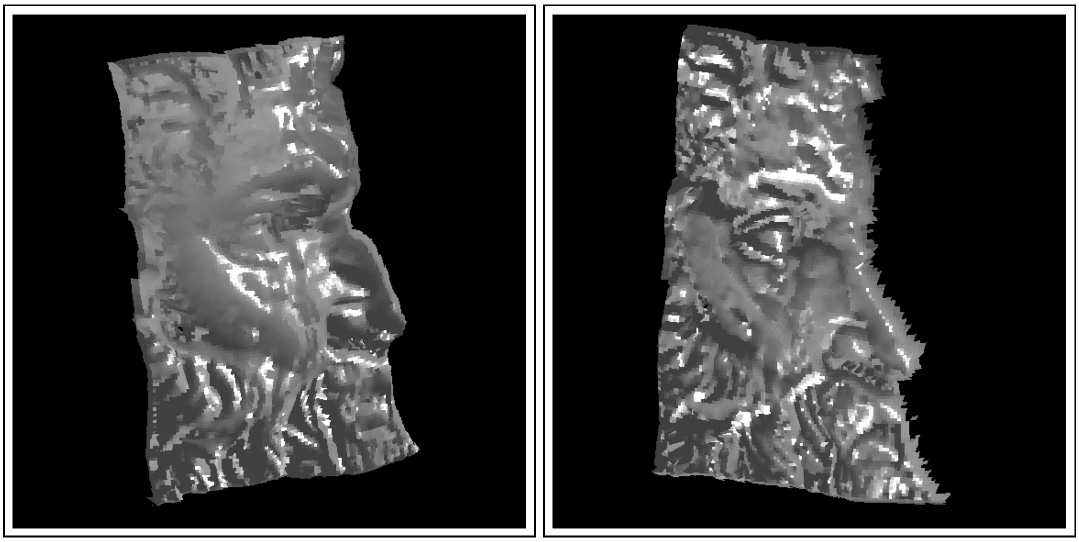


Figure 5.13: Views of the reconstructed surface of Michelangelo's Moses

have chosen the algorithm of Bichsel and Pentland since, from Zhang et al (Zhang et al., 1999), appears to deliver reasonable results on a wide variety of images. Moreover, this shape-from-shading algorithm is local a technique that implicitly recovers the surface. Therefore, comparison upon the basis of surface reconstruction is possible without introducing additional errors.

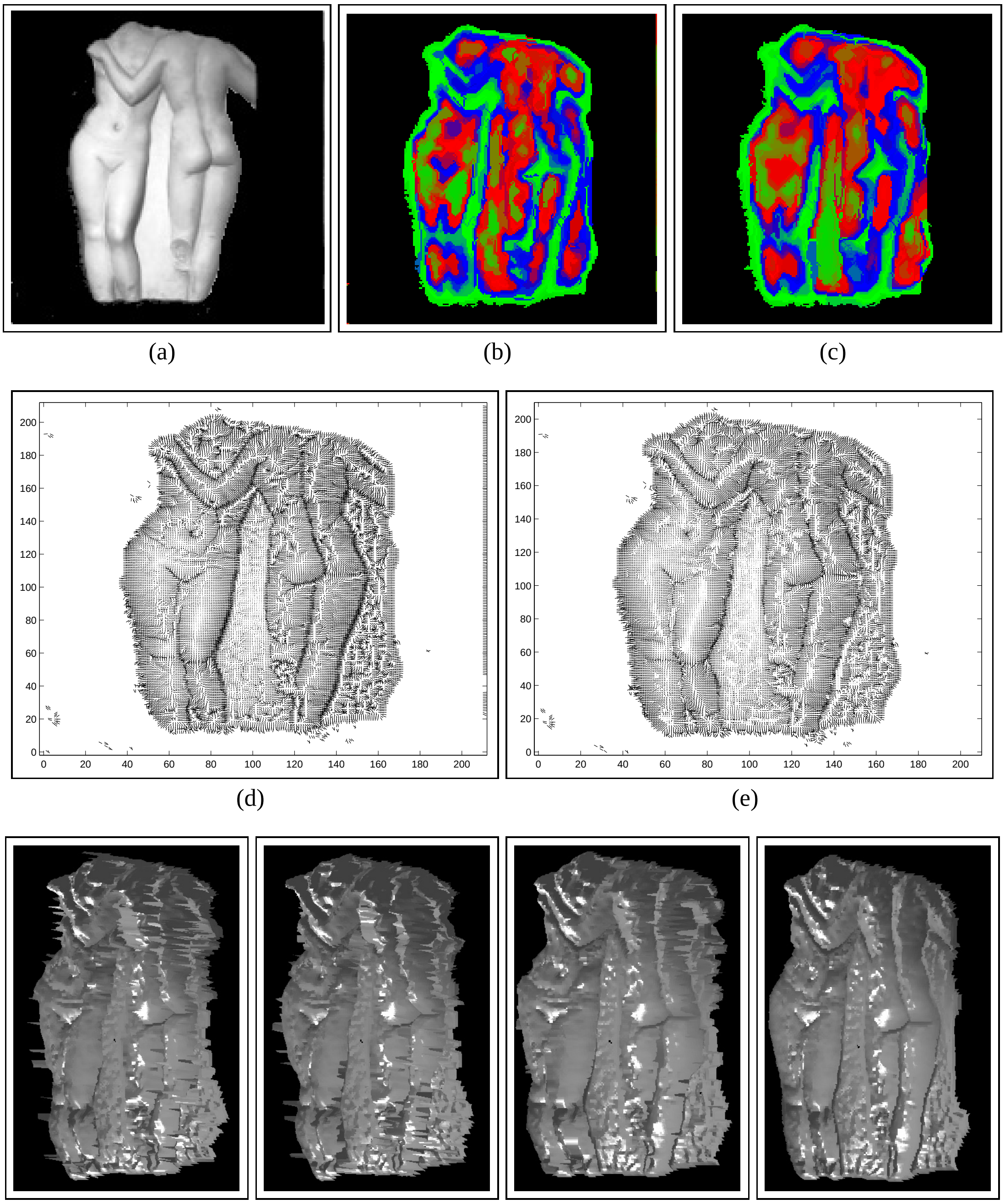
In Figure 5.6 we show the results obtained for a series of different surfaces. In the first, second and third rows we show the Lambertian shading, height data and surface normals for the synthetic surfaces. In the fourth row of the figure we show the surface reconstructed by applying our shape-form-shading method to the images in the top row. The bottom row of the figure shows the absolute error between the ground-truth and reconstructed surface height. From left-to-right the surfaces studied are a dome, a sharp ridge, a torus and a volcano. In all four cases the surface reconstructions are qualitatively good. For the dome the height errors are greater at the edges of the surface where the slope is largest. In the case of the ridge, there are errors at the crest. For the volcano, there are some problems with the recovery of the correct depth of the "caldera", i.e. the depression in the centre. For the reconstructed surfaces, the mean-squared errors are 5.6% for the dome, 10.8% for the ridge, 7.8% for the torus and 4.7% for the volcano. Hence, the method seems to have greater difficulty for surfaces containing sharp creases.

Next, we compare our results with those for the shape-from-shading height recovery method of Bichsel and Pentland (Bichsel and Pentland, 1992). In the top row of Figure 5.7, we show the four basic shapes under study. The remaining rows, from top-to-bottom, show the ground-truth height data, the recovered surface height and the absolute error between the ground-truth and the reconstructed surface height. We do not show the needlemaps due to the fact that the Bichsel and Pentland's algorithm does not deliver a field of surface normals at output. From the plots, it is clear that our algorithm outperforms the one of Bichsel and Pentland. Furthermore, in contrast with our results, the mean-squared error is 6.1% for the dome, 11.9% for the ridge, 10.2% for the torus and 26.8% for the volcano.

We have repeated these experiments under conditions of controlled noise. To do this we have added random measurement errors to the raw image brightness. The measurement errors have been sampled from a Gaussian distribution with zero mean and known variance. In Figure 5.8 we show the result of reconstructing the surfaces shown in Figure 5.6, when brightness errors have been added. The order of the rows is the same as Figure 5.6. The results for the shape-from-shading height recovery method of Bichsel and Pentland are shown in Figure 5.9. The row-order is the same as in Figure 5.7.

From Figure 5.8, it is clear that, for all four surfaces, our algorithm preserves the gross structure of the surfaces under study. However, the recovered height is clearly noisy. The height difference plots are relatively unstructured. These are important observations. They mean that our graph-spectral method simply transfers errors in surface normal direction into errors in height, without producing structural noise artefacts. In contrast, the surfaces recovered making use of the Bichsel and Pentland's algorithm are poorer for all basic shapes but the ridge. Furthermore, the algorithm of Bichsel and Pentland recovers broken, uneven surfaces due to convex-concavity surface inversion errors introduced by the noise added to the image brightness.

To investigate the effect of noise further, in Figures 5.10 and 5.11 we plot the mean-squared error for our graph-spectral shape-from-shading algorithm and the height recov-


(a) (b) (c)

(d) (e)

(f) (g) (h) (i)

Figure 5.14: Results of the algorithm on the image of “The Three Graces” Relief fragment

ery method of Bichsel and Pentland. Here, we show the mean-squared error for the reconstructed surface height as a function of the standard deviation of the added Gaussian noise. The different curves are for the different surfaces shown in Figures 5.6 and 5.7. From the plots for the different surfaces shown in Figure 5.10, it is clear that, for our algorithm, the mean-squared error grows slowly with increasing noise standard deviation.

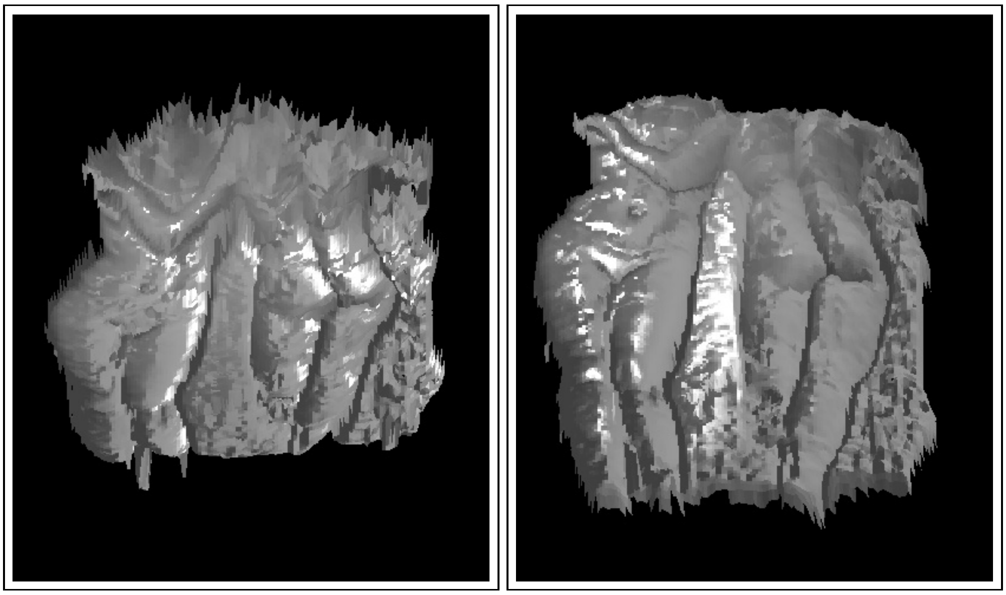
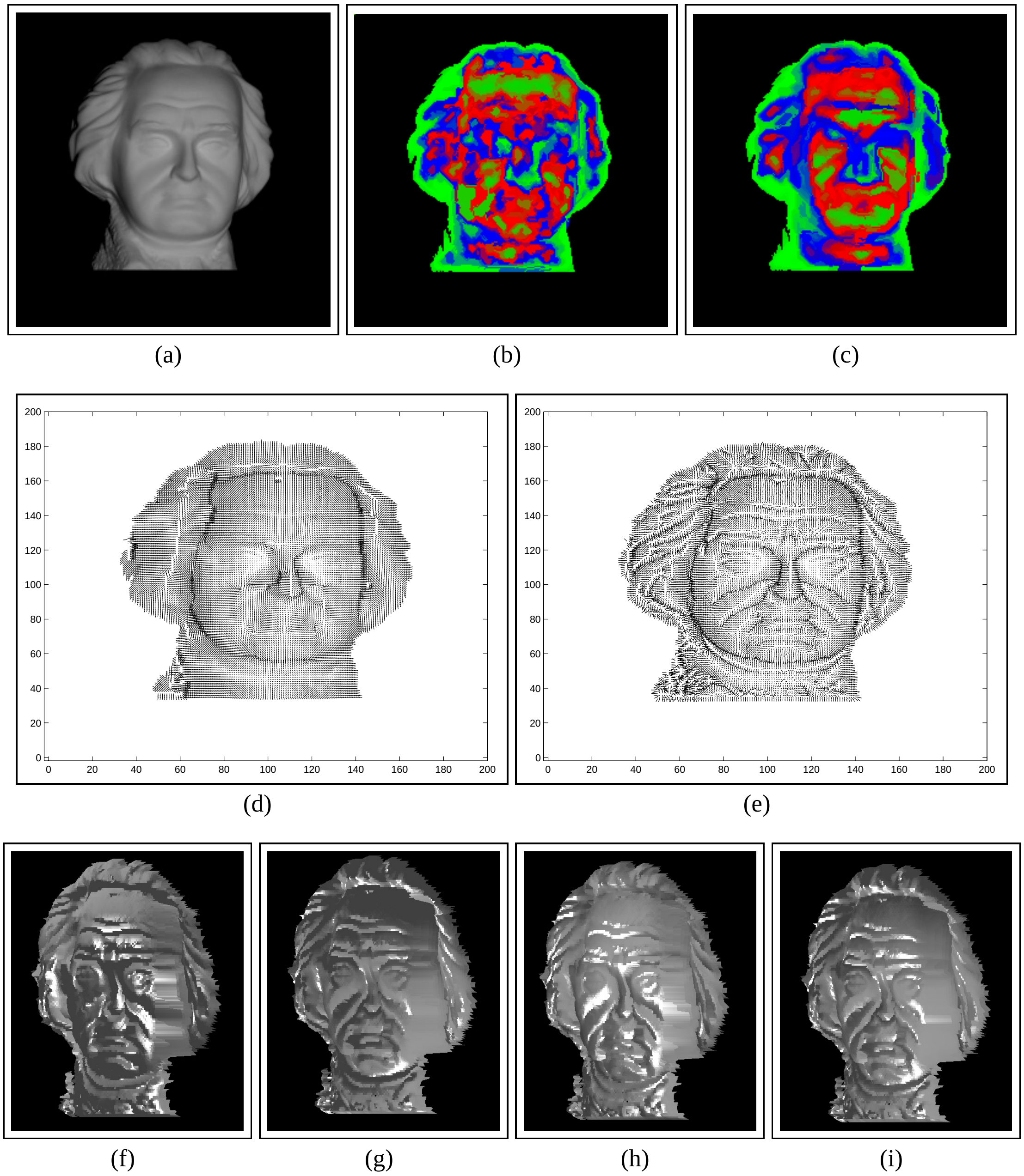


Figure 5.15: Views of the surface of “The Three Graces” Relief fragment

The torus and the volcano give the poorest errors, while the ridge and the dome give the smallest errors. This is a reflection of the fact that the torus and the volcano are the more structured surfaces. This characteristic of the torus and volcano becomes ever evident in Figure 5.11. Despite the error for the dome in Figure 5.11 is relatively small, the mean-squared error increases rapidly for the other three basic shapes, i.e. the ridge, the torus and the volcano.

5.10 Experiments

We have experimented with a variety of real world images, but we have concentrated our attention mainly on images of classical statues. These objects are predominantly Lambertian, although there are local specular highlights and local albedo variations. In principle we can overcome both of these problems. In a recent paper, we have described a probabilistic method for specularity removal which uses the Torrance and Sparrow model to perform Lambertian reflectance correction for shiny objects (Ragheb and Hancock, 2001). Local albedo changes can be accommodated using brightness normalisation or histogram equalisation. The objects studied are a detail of Michaelangelo’s Moses and a



(a) (b) (c)

(d) (e)

(f) (g) (h) (i)

Figure 5.16: Results of the algorithm on the image of the Beethoven bust

section of the relief “Three Graces”. We have also used an image of a bust of Beethoven from the University of Central Florida shape-from-shading data-base (Zhang et al., 1999).

In Figure 5.12 we show our first sequence of results. The panels of the figure are organised as follows. In Figure 5.12a we show the original image used as input to the shape-from-shading process. This is a side view of the head of the statue “Moses”. Figures 5.12b and c show the arrangements of quadric patches after one and four iterations

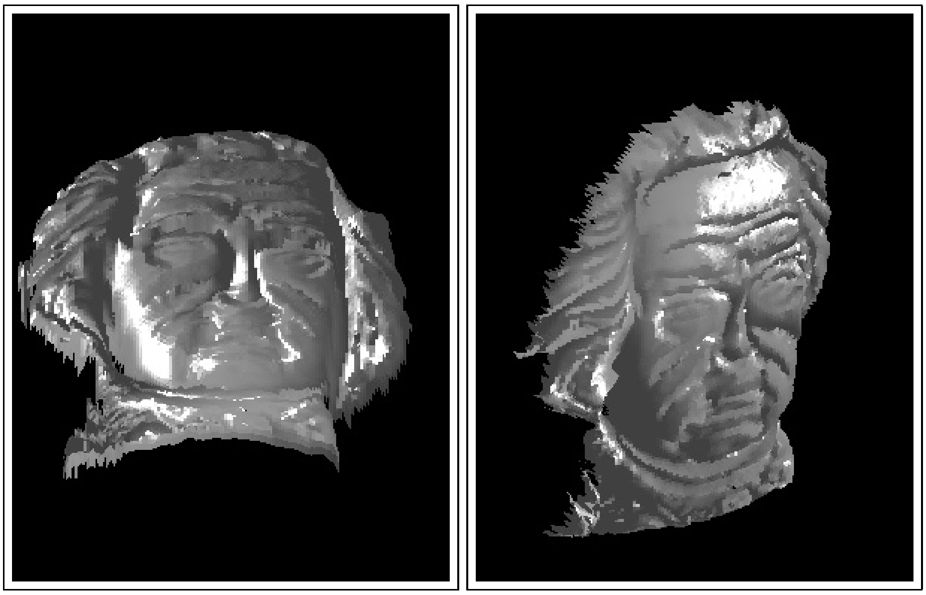


Figure 5.17: Views of the reconstructed surface of the Beethoven bust

of the algorithm. In these images, the different quadric patches are coded in different colours. Figures 5.12 d and e show the initial and final needle maps used for the purposes of surface integration. Finally, Figures 5.12 f to i show the reconstructed surface after successive iterations of the algorithm. Initially, the set of surface patches is fragmented, lack coherence and do not reproduce the surface detail well. However, after four iterations the reconstructed surface is more continuous and the fine detail of the object is well reproduced.

In Figure 5.13 we show the reconstructed surface viewed from two different directions. There are a number of features that are worth noting from the panels of the figure. First, the organisation of the surface normals and the arrangement of patches both improve as the algorithm iterates. Second, from the different surface views it is clear that the surface structure is well reconstructed. For instance the shape of the nose, particularly in the proximity of the nostrils, is well reproduced. Moreover, the fine structure of the beard, the detail in the eye-sockets and the shape of the cheek bones are all well reconstructed.

In Figure 5.14 we repeat the sequence of panels and in Figure 5.15 we show different views of the reconstructed surface for a section of the relief "The Three Graces". However, only two of the subjects of the original sculpture are visible in the section of image used. The same iterative improvement in the quality of the surface normals and the patch arrangement is clear. In this case the algorithm converged after 3 iterations. Morover, the different views of the surface reveal that the detail of the relief is well reproduced. The legs, buttocks and indentation in the back of the left-hand figure are all well reconstructed.

Figures 5.16 and 5.17 show analogous results for an image of a bust of Beethoven. Here the path structure is particularly clear and corresponds well to the topographic structure of the surface. For instance, the eye sockets correspond to distinct patches. Both the patch structure and the structure of the needle maps are improved as the algorithm iterates. Initially, little of the surface structure is evident. However, after the algorithm has converged the structure of the hair and the boundary of the cheeks have become well defined. These features are all clear in the different views of the reconstructed surface.

Finally, we have compared our algorithm with two alternatives. The first of these

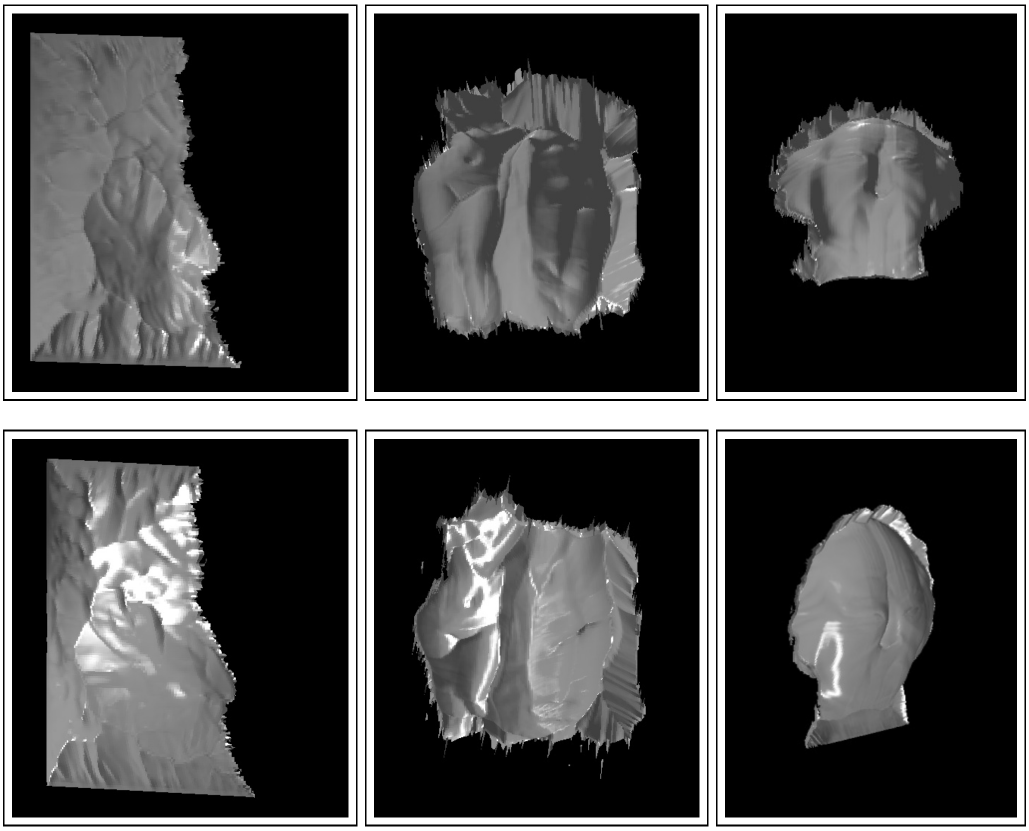


Figure 5.18: Results for the Bichsel and Pentland’s algorithm

is the shape-from-shading height recovery method of Bichsel and Pentland (Bichsel and Pentland, 1992). The second method used in our comparison is a purely geometric surface integration method, which like the technique reported in this paper takes the needle-maps delivered by the Worthington and Hancock algorithm as input. This method uses the trapezoid rule to increment the height from equal height reference contours. The method was developed in conjunction with a study of terrain reconstruction using radar shape-from-shading and a full description can be found in (Bors et al., 2003). The results for these two alternatives are shown in Figures 5.18 and 5.19.

For the Bichsel and Pentland's method, the overall quality of surface detail is poorer, and the method is unstable in shadows. For instance, in the case of the Moses only the chin and mouth appear to be well reconstructed. The height of the cheek is overestimated, and convex parts of the surface have become concave. There are similar problems with convexity-concavity inversion for the Three Graces. The results obtained by the goemetric surface integration method are better than those obtained by Bichsel and Pentland and do not suffer so badly from the problems of surface inversion. Although the detail is beter than that obtained using the Bichsel and Pentland method, the results delivered by the geometric integration method but do not contain the fine detail delivered by our graph spectral method. For instance, the hair and fabric details of Beethoven are not well reproduced and the structures of the beard of Moses is not well recovered.

5.11 Conclusions

In this chapter, we have described a graph-spectral algorithm for shape-from-shading. We constrain the surface normals at each image location to fall on an irradiance cone whose axis is the light source direction and whose apex angle is determined by the measured image brightness. The method uses the modal structure of a transition weight matrix to both locate surface patches and identify a curvature minimising path for surface integration, and hence height recovery. By fitting quadric surfaces to the height data for pixel sites contained within patches, we perform surface smoothing. We update the surface normal

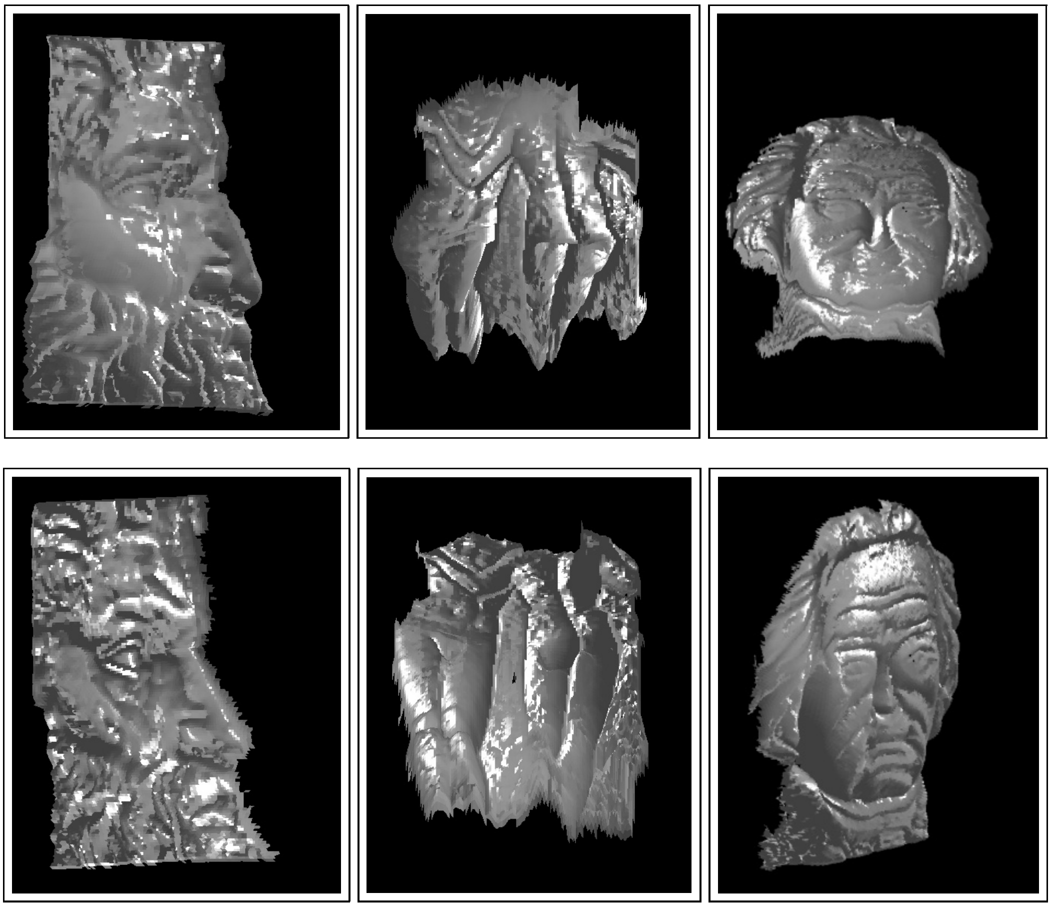


Figure 5.19: Results for the geometric surface integration algorithm

directions by rotating them so that they point in the direction of the fitted surface gradient. The surface integration and surface normal adjustment steps are iterated until stable height estimates are recovered. The method proves effective for reconstructing surfaces from single views of 3D objects, and gives subjectively better results than a number of alternative shape-from-shading methods.

Furthermore, the spectral analysis for generating paths is of generic utility. In the next chapter, we will explore how it can be used to convert graphs to strings for the purposes of graph-matching and graph edit distance computation.

Chapter 6

String Edit Distance and Graph Matching

In the previous chapter, we presented a spectral analysis for extracting paths from a graph such that strongly correlated nodes are placed close to one another. In this chapter, we aim to make use of this spectral seriation technique for purposes of graph-matching and graph edit distance computation.

The search for a robust means of inexact graph-matching has been the focus of sustained activity over the last two decades. Early work drew heavily on ideas from structural pattern recognition and revolved around extending the concept of string edit distance to graphs (Sanfeliu and Fu, 1983; Eshera and Fu, 1984a). One of the criticisms that can be aimed at this early work is that it lacks the formal rigour of the corresponding work on string edit distance. However, recently, Bunke and his co-workers have returned to the problem and have shown the relationship between graph edit distance and the size of the maximum common subgraph (Bunke, 1997). Another argument leveled against these methods draws from the fact they lack the elegance of the matrix representation first used by Ullman in his work on subgraph isomorphism (Ullman, 1976).

To overcome this drawbacks, we adopt an alternative approach to the problem. Here, we convert graphs to string sequences and use the existing theory of string edit distance to recover the matches. In doing this, we exploit the seriation method presented in the previous chapter to develop a spectral method for computing graph edit distance. The use of a path-based method for purposes of graph-matching has a number of advantages. First, it allows the inexact graph matching problem to be posed in a matrix setting. This has proved to be an elusive task, which is disappointing since a rich set of potential tools are available from spectral-graph theory. Secondly and most importantly, in this way, spectral properties may be used for posing the graph edit distance computation in a rigorous theoretical setting. It is worth stressing that although there been attempts to extend the string edit idea to trees and graphs (Wang et al., 1998; Oommen and Zhang, 1996; Sanfeliu and Fu, 1983; Shapiro and Haralick, 1982), there is considerable current effort aimed at putting the underlying methodology on a rigourous footing.

The outline of this chapter is as follows. In Section 6.1 we describe a graph-spectral method for performing graph seriation. This revolves around computing the leading eigenvector a modified adjacency matrix. Section 6 introduces the classical concept of string edit distance and poses the problem of computing graph-edit distance as maximum *a posteriori* probability (MAP) estimation of an edit path given the leading eigenvectors of the adjacency matrices of the graphs being matched. The model ingredients needed to apply the MAP estimation scheme are detailed in Section 6.3. Experiments on real world and synthetic data are presented in Section 6.4. Finally, Section 6.5 offers some conclusions.

6.1 Graph Seriation

Consider the graph $G = (V, E)$ with node index-set V and edge-set $E = \{(i,k)|(i,k) \in V \times V, i \neq k\}$. Associated with the graph is an adjacency matrix A whose elements are defined as follows

$$A(i,k) = \begin{cases} 1 & \text{if } (i,k) \in E \\ 0 & \text{otherwise} \end{cases} \tag{6.1}$$

By taking the leading eigenvector ϕ^* of the adjacency matrix, we convert the graph into a string using the procedure outlined in Section 5.3. In practice we will be interested in finding the edit distance for a pair of graphs $G_M = (V_M, E_M)$ and $G_D = (V_D, E_D)$. The leading eigenvectors of the corresponding adjacency matrices A_M and A_D are respectively ϕ_M^* and ϕ_D^*. The string representations of the graph G_M is denoted by X and that for the graph G_D by Y.

There are similarities between the use of the leading eigenvector for seriation and the use of spectral methods to find the steady state random walk on a graph. There are more detailed discussions of the problem of locating the steady state random walk on a graph in the reviews by Lovasz (Lovász, 1993) and Mohar (Mohar, 2000). An important result described in these papers, is that if we visit the nodes of the graph in the order defined by the magnitudes of the co-efficients of the leading eigenvector of the transition probability matrix, then the path is the steady state Markov chain. However, this path is not guaranteed to be edge connected. Hence, it can not be used to impose a string ordering on the nodes of a graph. The seriation approach adopted in this chapter does, on the other hand, impose edge connectivity constraints and can hence be used to convert graphs to strings in a manner which is suitable for computing edit distance.

6.2 Probabilistic Framework

We are interested in computing the edit distance between the graphs $G_M = (V_M, E_M)$ referred to as the model graph and the graph $G_D = (V_D, E_D)$ referred to as the data-graph. The seriations of the two graphs are denoted by $X = \{x_1, x_2, ..., x_{|V_M|}\}$ for the model graph and $Y = \{y_1, y_2, ..., y_{|V_D|}\}$ for the data graph. These two strings are used to index the rows and column of an edit lattice. The rows of the lattice are indexed using the data-graph string, while the columns are indexed using the model-graph string. To allow for differences in the sizes of the graphs we introduce a null symbol ϵ which can be used to pad the strings. We pose the problem of computing the edit distance as that of finding a path $\Gamma =< \gamma_1, \gamma_2, ...\gamma_k, ..., \gamma_L >$ through the lattice. Each element $\gamma_k \in (V_D \cup \epsilon) \times (V_M \cup \epsilon)$ of the edit path is a Cartesian pair. We constrain the path to be connected on the edit lattice. In particular, the transition on the edit lattice from the state γ_k to the state γ_{k+1} is constrained to move in a direction that is increasing and connected in the horizontal, vertical or diagonal direction on the lattice. The diagonal transition corresponds to the match of an edge of the data graph to an edge of the model graph. A horizontal transition means that the data-graph index is not incremented, and this corresponds to the case where the traversed nodes of the model graph are null-matched. Similarly when a vertical transition is made, then the traversed nodes of the data-graph are null-matched.

Suppose that $\gamma_k = (a, b)$ and $\gamma_{k+1} = (c, d)$ represent adjacent states in the edit path between the seriations X and Y. According to the classical approach, the cost of the edit path is given by

$$d(X, Y) = \mathcal{E}(\Gamma) = \sum_{\gamma_k \in \Gamma} \eta(\gamma_k \rightarrow \gamma_{k+1}) \tag{6.2}$$

where $\eta(\gamma_k \rightarrow \gamma_{k+1})$ is the cost of the transition between the states $\gamma_k = (a, b)$ and $\gamma_{k+1} = (c, d)$. The optimal edit path is the one that minimises the edit distance between string, and satisfies the condition

$$\Gamma^* = \arg \min_{\Gamma} \mathcal{E}(\Gamma) \tag{6.3}$$

and hence the edit distance is $d(X, Y) = \mathcal{E}(\Gamma^*)$. Classically, the optimal edit sequence may be found using Dijkstra's algorithm or by using the quadratic programming method of Wagner and Fisher (Wagner and Fisher, 1974).

However, in this chapter we adopt a different approach. We aim to find the edit path that has maximum probability given the available leading eigenvectors of the data-graph and model-graph adjacency matrices. Hence, the optimal path is the one that satisfies the condition

$$\Gamma^* = \arg \max_{\Gamma} P(\Gamma | \phi_X^*, \phi_Y^*) \tag{6.4}$$

To develop this decision criterion into a practical edit distance computation scheme, we need to develop the *a posteriori* probability appearing above. We commence by using the definition of conditional probability to re-write the *a posteriori* path probability in terms of the joint probability density $P(\phi_X^*, \phi_Y^*)$ for the leading eigenvectors and the joint density function $P(\phi_X^*, \phi_Y^*, \Gamma)$ for the leading eigenvectors and the edit path. The result is

$$P(\Gamma|\phi_X^*, \phi_Y^*) = \frac{P(\phi_X^*, \phi_Y^*, \Gamma)}{P(\phi_X^*, \phi_Y^*)} \tag{6.5}$$

We can rewrite the joint density appearing in the numerator to emphasise the role of the components of the adjacency matrix leading eigenvectors and the component edit transitions explicit

$$P(\Gamma|\phi_M^*, \phi_D^*) = \frac{P(\phi_X^*(1), \phi_X^*(2), ..., \phi_Y^*(1), \phi_Y^*(2)..., \gamma_1, \gamma_2, ...)}{P(\phi_M^*, \phi_D^*)} \tag{6.6}$$

To simplify the numerator, we make a conditional independence assumption. Specifically, we assume that the components of the leading eigenvector of the adjacency matrices, depend only on the edit transition γ_k associated with their node-indices. Hence, we can perform the factorisation

$$P(\Gamma|\phi_M^*, \phi_D^*) = \frac{\left\{\prod_{k=1}^{L} P(\phi_X^*(a), \phi_Y^*(b)|\gamma_k)\right\} P(\gamma_1, \gamma_2, \ldots, \gamma_L)}{P(\phi_X^*, \phi_Y^*)} \tag{6.7}$$

where $P(\gamma_1, \gamma_2, ..., \gamma_L)$ is the joint prior for the sequence of edit transitions and (a, b) is the coordinate pair on the edit lattice associated with the state γ_k. To simplify the joint prior, we assume that transitions between sites that are not adjacent on the edit lattice are conditionally independent. As a result

$$P(\gamma_1, \gamma_2, \ldots, \gamma_L) = P(\gamma_L) \prod_{k=1}^{L-1} P(\gamma_k|\gamma_{k+1}) \tag{6.8}$$

This takes the form of a factorisation of conditional probabilities for transitions between sites on the edit lattice $P(\gamma_k|\gamma_{k+1})$, except for the term $P(\gamma_L)$ which results from the final site visited on the lattice. To arrive at a more homogeneous expression, we use the definition of conditional probability to re-express the joint conditional measurement density for the adjacency matrix leading eigenvectors in the following form

$$P(\phi_X^*(a), \phi_Y^*(b)|\gamma_k) = \frac{P(\gamma_k|\phi_X^*(a), \phi_Y^*(b))P(\phi_X^*(a), \phi_Y^*(b))}{P(\gamma_k)} \tag{6.9}$$

Substituting Equations (6.8) and (6.9) into Equation (6.7) we find

$$\begin{aligned} P(\Gamma|\phi_X^*, \phi_Y^*) &= \left\{ \prod_{k=1}^{L} P(\gamma_k|\phi_X^*(a), \phi_Y^*(b)) \frac{P(\gamma_k, \gamma_{k+1})}{P(\gamma_k)P(\gamma_{k+1})} \right\} \\ &\times \frac{\prod_{k=1}^{L} P(\phi_X^*(a), \phi_Y^*(b))}{P(\phi_X^*, \phi_Y^*)} \end{aligned} \tag{6.10}$$

Since, the joint measurement density $P(\phi_X^*, \phi_Y^*)$ does not depend on the edit path, it does not influence the decision process and we remove it from further consideration. Hence, the optimal path across the edit lattice is

$$\Gamma^* = \arg \max_{\gamma_1, \gamma_2 \cdots, \gamma_L} \left\{ \prod_{k=1}^{L} P(\gamma_k|\phi_X^*(a), \phi_Y^*(b)) \frac{P(\gamma_k, \gamma_{k+1})}{P(\gamma_k)P(\gamma_{k+1})} \right\} \tag{6.11}$$

The information concerning the structure of the edit path on the lattice is captured by the quantity

$$R_{k,k+1} = \frac{P(\gamma_k, \gamma_{k+1})}{P(\gamma_k)P(\gamma_{k+1})} \tag{6.12}$$

we refer to this quantity as the edge-compatibility co-efficient.

To establish a link with the classical edit distance picture presented in Section 3, we can re-write the optimal edit path as a minimisation problem involving the negative logarithm of the *a posteriori* path probability. The optimal path is the one that satisfies the condition

$$\Gamma^* = \arg \min_{\gamma_1, \gamma_2 \cdots, \gamma_L} \left\{ \sum_{k=1}^{L} \left[-P(\gamma_k|\phi_X^*(a), \phi_Y^*(b)) - \ln R_{k,k+1} \right] \right\} \tag{6.13}$$

As a result the elementary edit cost $\eta(\gamma_k \to \gamma_{k+1})$ from the site (a, b) to the site (c, d) is

$$\begin{aligned}\eta(\gamma_k \to \gamma_{k+1}) &= \eta((a, b) \to (c, d)) \\ &= -\big(\ln P(\gamma_k|\phi_X^*(a), \phi_Y^*(b)) + \ln P(\gamma_{k+1}|\phi_*^X(c), \phi_*^Y(d)) + \ln R_{k,k+1}\big)\end{aligned} \tag{6.14}$$

The expression hence contains separate terms associated with the cost of visiting individual sites on the lattice, and for making transitions between sites.

6.3 Model Ingredients

To apply the decision rule and compute the edit costs, we require models of the *a posteriori* probabilities of visiting the sites of the edit lattice, i.e. $P(\gamma_k|\phi_X^*(a), \phi_Y^*(b))$ and of $R_{k,k+1}$.

6.3.1 Lattice Transition Probabilities

We have recently reported a methodology which lends itself to the modelling of the edge compatibility co-efficient $R_{k,k+1}$ (Myers et al., 2000). It is based on the idea of constraint corruption through the action of a label-error process. The basis for this constraint softening process is a binomial distribution of probability. Uncorrupted edges occur with total probability mass $(1-p)^2$. This probability mass is distributed between the $|E|$ edges of the graph. Edges with one node matched and one-node null-matched have total probability mass $2p(1-p)$. This probability mass is distributed amongst the $2|V|$ configurations involving a node and a null-symbol ϵ. Edges in which both nodes are null-matched take the remaining probability mass, i.e. p^2. The is one such configuration i.e (ϵ, ϵ) to which this mass of probability may be assigned. Matches involving non-null label pairs outside the edge-set V_M are completely forbidden and therefore account for zero total probability mass.

In each of the three cases listed above the available mass of probability is distributed uniformly among the label configurations falling into the relevant constraint class. We assume that joint errors in the two graphs under consideration are independent. As a result the resulting distribution of joint probability is specified by the following rule

$$P(\gamma_k, \gamma_{k+1}) = \begin{cases} \frac{(1-p_D)^2}{|E_D|}\frac{(1-p_M)^2}{|E_M|} & \text{if } (a,c) \in E_D \text{ and } (b,d) \in E_M \\ \frac{(1-p_D)^2}{|E_D|}\frac{p_M(1-p_M)}{|V_M|} & \text{if } (a,c) \in E_D \text{ and } (b,d) \in (V_M \times \epsilon) \cup (\epsilon \times V_M) \\ \frac{(1-p_D)^2}{|E_D|}p_M^2 & \text{if } (a,c) \in E_D \text{ and } (b,d) = (\epsilon, \epsilon) \\ \frac{p_D(1-p_D)}{|V_D|}\frac{(1-p_M)^2}{|E_M|} & \text{if } (a,c) \in (V_D \times \epsilon) \cup (\epsilon \times V_D) \text{ and } (b,d) \in E_M \\ \frac{p_D(1-p_D)}{|V_D|}\frac{p_M(1-p_M)}{|V_M|} & \text{if } (a,c) \in (V_D \times \epsilon) \cup (\epsilon \times V_D) \text{ and} \\ & (b,d) \in (V_M \times \epsilon) \cup (\epsilon \times V_D) \\ \frac{p_D(1-p_D)}{|V_D|}p_M^2 & \text{if } (a,c) \in (V_D \times \epsilon) \cup (\epsilon \times V_D) \text{ and } (b,d) = (\epsilon, \epsilon) \\ p_D^2\frac{(1-p_M)^2}{|V_M|} & \text{if } (a,c) = (\epsilon, \epsilon) \text{ and } (b,d) \in E_M \\ p_D^2\frac{p_M(1-p_M)}{|V_M|} & \text{if } (a,c) = (\epsilon, \epsilon) \text{ and } (b,d) \in (V_M \times \epsilon) \cup (\epsilon \times V_M) \\ p_D^2 p_M^2 & \text{if } (a,c) = (\epsilon, \epsilon) \text{ and } (b,d) = (\epsilon, \epsilon) \\ 0 & \text{otherwise} \end{cases} \tag{6.15}$$

As a result of this distribution rule, it can be shown that the quantity $R_{k,k+1}$ depends only of the edge densities $\rho_D = \frac{E_D}{|V_D|^2}$ and $\rho_M = \frac{E_M}{|V_M|^2}$ for the two graphs under study.

Specifically,

$$R_{k,k+1} = \begin{cases} \rho_D \rho_M & \text{if } (a,c) \in E_D \text{ and } (b,d) \in E_M \\ \rho_D & \text{if } (a,c) \in E_D \text{ and } (b,d) \in (V_M \times \epsilon) \cup (\epsilon \times V_M) \\ \rho_D & \text{if } (a,c) \in E_D \text{ and } (b,d) = (\epsilon, \epsilon) \\ \rho_M & \text{if } (a,c) \in (V_D \times \epsilon) \cup (\epsilon \times V_D) \text{ and } (b,d) \in E_M \\ 1 & \text{if } (a,c) \in (V_D \times \epsilon) \cup (\epsilon \times V_D) \text{ and } (b,d) \in (V_M \times \epsilon) \cup (\epsilon \times V_D) \\ 1 & \text{if } (a,c) \in (V_D \times \epsilon) \cup (\epsilon \times V_D) \text{ and } (b,d) = (\epsilon, \epsilon) \\ 1 & \text{if } (a,c) = (\epsilon, \epsilon) \text{ and } (b,d) \in E_M) \\ 1 & \text{if } (a,c) = (\epsilon, \epsilon) \text{ and } (b,d) \in (V_M \times \epsilon) \cup (\epsilon \times V_M) \\ 1 & \text{if } (a,c) = (\epsilon, \epsilon) \text{ and } (b,d) = (\epsilon, \epsilon) \\ 0 & \text{otherwise} \end{cases} \tag{6.16}$$

The contingency table given above can be re-expressed in terms of the transitions on the edit lattice. The move contingency table is

$$R_{k,k+1} = \begin{cases} \rho_M \rho_D & \text{if } \gamma_k \to \gamma_{k+1} \text{ is a diagonal transition on the edit lattice,} \\ & \text{i.e. } (a,c) \in E_D \text{ and } (b,d) \in E_M \\ \rho_M & \text{if } \gamma_k \to \gamma_{k+1} \text{ is a horizontal transition on the edit lattice} \\ & \text{i.e. } (a,c) \in E_D \text{ and } b = \epsilon \text{ or } d = \epsilon \\ \rho_D & \text{if } \gamma_k \to \gamma_{k+1} \text{ is a vertical transition on the edit lattice,} \\ & \text{i.e. } a = \epsilon \text{ or } c = \epsilon \text{ and } (b,d) \in E_M \\ 1 & \text{if } a = \epsilon \text{ or } c = \epsilon \text{ and } b = \epsilon \text{ or } d = \epsilon \end{cases} \tag{6.17}$$

6.3.2 A Posteriori Correspondence Probabilities

The second model ingredient is the *a posteriori* probability of visiting a site on the lattice. Here we assume that the differences in the components of the adjacency matrix leading eigenvectors are drawn from a Guassian distribution. Hence,

$$P(\gamma_k|\phi_X^*(a),\phi_Y^*(b)) = \begin{cases} \frac{1}{\sqrt{2\pi}\sigma}\exp\left\{-\frac{1}{2\sigma^2}(\phi_X^*(a)-\phi_Y^*(b))^2\right\} & \text{if } a \neq \epsilon \text{ and } b \neq \epsilon \\ \alpha & \text{if } a = \epsilon \text{ or } b = \epsilon \end{cases} \tag{6.18}$$

6.3.3 Minimum Cost Path

Once the edit costs are computed, we proceed to find the path that yields the minimum edit distance. Our adopted algorithm makes use of the fact that the minimum cost path along the edit lattice is composed of sub-paths that are also always of minimum cost. Hence, following Levenshtein (Levenshtein, 1966) we compute a $|V_D| \times |V_M|$ transition-cost matrix Ψ. The elements of the matrix are computed recursively using the formula

$$\psi_{i,j} = \begin{cases} \eta(\gamma_i \rightarrow \gamma_j) & \text{if } j = 1 \text{ and } i = 1 \\ \eta(\gamma_i \rightarrow \gamma_j) + \psi_{i,j-1} & \text{if } i = 1 \text{ and } j \geq 2 \\ \eta(\gamma_i \rightarrow \gamma_j) + \psi_{i-1,j} & \text{if } j = 1 \text{ and } i \geq 2 \\ \eta(\gamma_i \rightarrow \gamma_j) + \min(\psi_{i-1,j},\psi_{i,j-1},\psi_{i-1,j-1}) & \text{if } i \geq 2 \text{ and } j \geq 2 \end{cases} \tag{6.19}$$

The matrix Ψ is a matrix representation of the accumulated minimal costs of the path along the edit lattice constrained to horizontal, vertical and diagonal transitions between adjacent coordinates. The minimum cost path can be proven to be that of the path closest to the diagonal of the matrix (Levenshtein, 1966). As a result, the edit distance is given by the bottom rightmost element of the transition-cost matrix. Hence, $d(X,Y) = \Psi_{|V_D|,|V_M|}$.

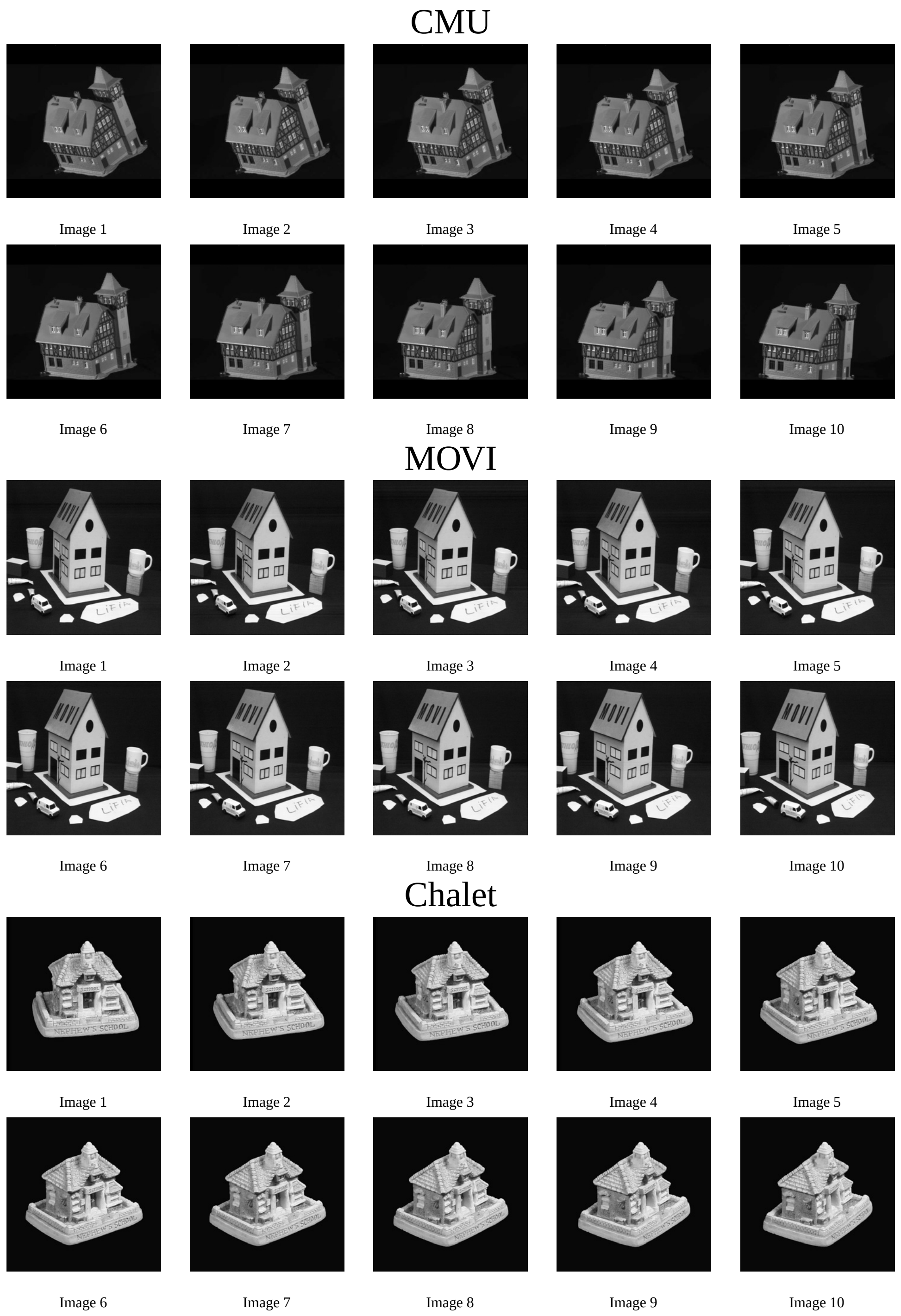


Figure 6.1: Image sequences used for the experiments

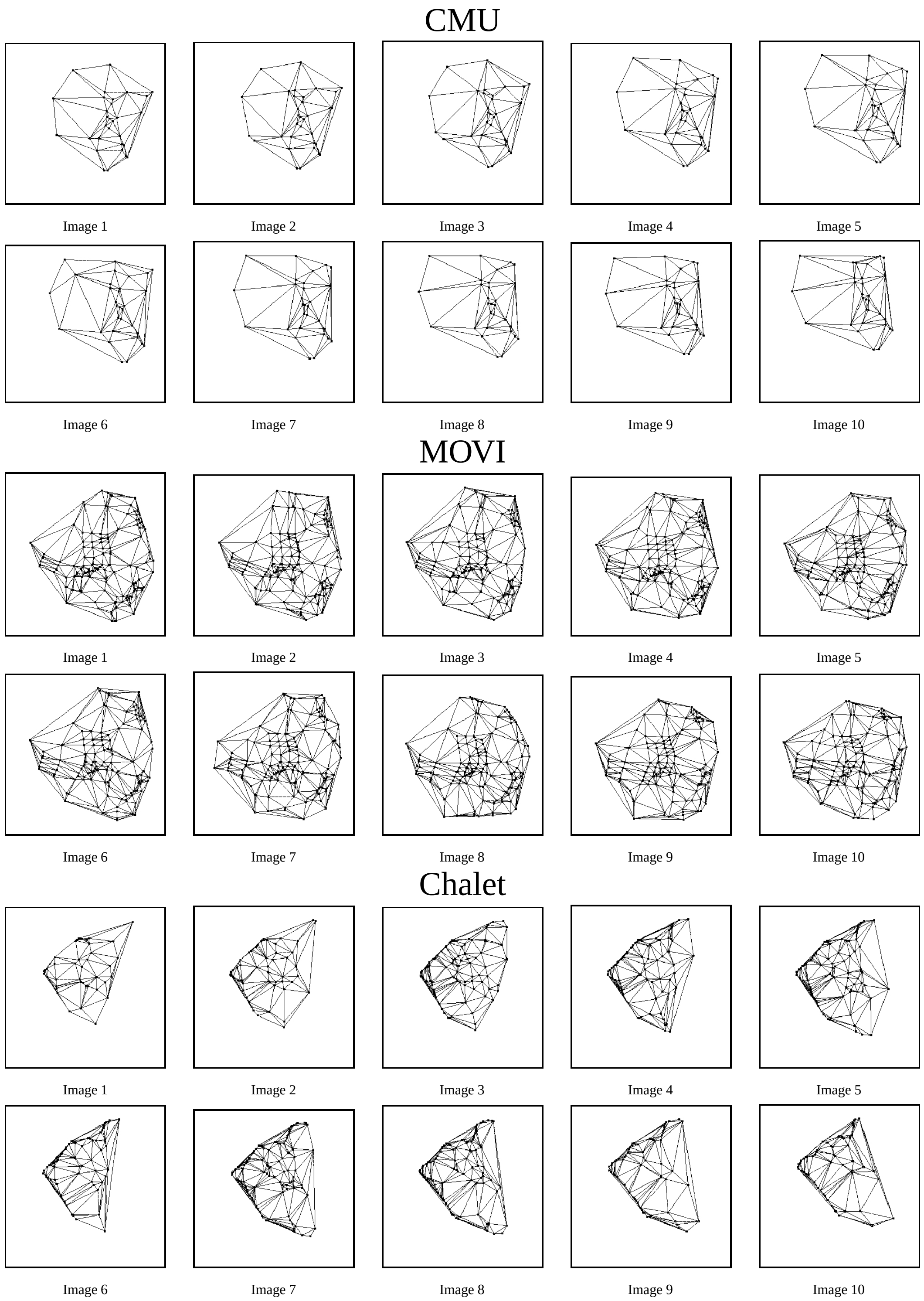


Figure 6.2: Delaunay triangulations corresponding to the image sequences

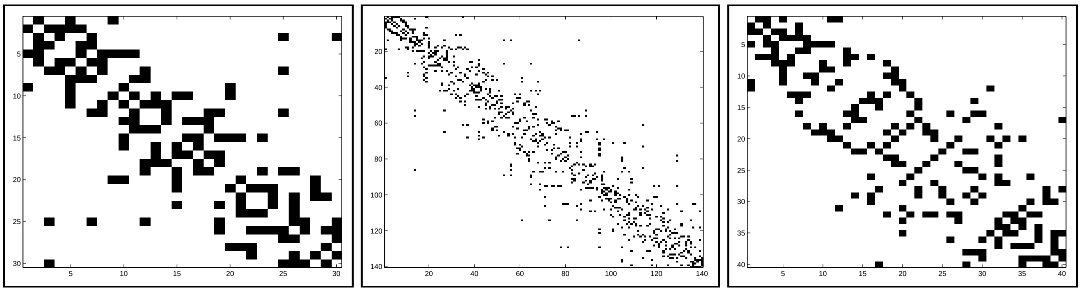
6.4 Experiments

Our experimental study is concerned with demonstrating the utility of the distances computed using our new method for the problem of graph-clustering. We explore two problems. The first of these involves a data-base containing different views of a number of 3D objects. The second problem involves finding sets of similar shapes in a large data-base of trademarks and logos. We conclude the experimental section with some experiments on synthetic data aimed at measuring the sensitivity of the method to structural differences in the graphs.

6.4.1 View-based Object Recognition

We have experimented with our new matching method on an application involving a database containing different perspective views of a number of 3D objects. The objects used in our study are model houses. The different views are obtained as the camera circumscribes the object. The three object sequences used in our experiments are the CMU-VASC sequence, the INRIA MOVI sequence and a sequence of views of a model Swiss chalet, that we collected in-house. In our experiments we use ten images from each of the sequences. The images used in out experiments are shown in Figure 6.1. To construct graphs for the purposes of matching, we have first extracted corners from the images using the corner detector of Luo, Cross and Hancock (Luo and Hancock, 1999). The graphs used in our experiments are the Delaunay triangulations of these corner points. The graphs extracted from the images are shown in Figure 6.2.

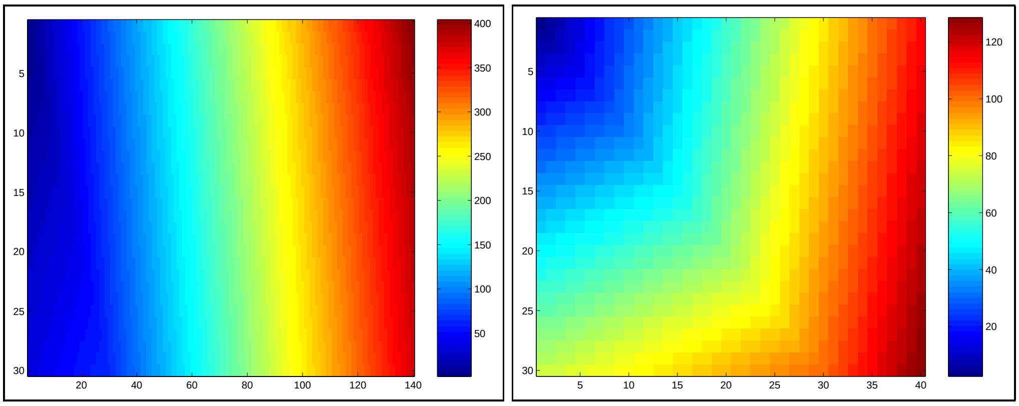
For the 30 graphs contained in the database, we have computed the complete set of 30×29 =870 distances between each of the distinct pairs of graphs. In Figure 6.3 we show the adjacency matrix for the first image of each of the three sequences used in our experiments. Finally, in Figure 6.4 we show the two Levenshtein matrices used to compute the edit distances between the first graph in the CMU sequence and the first graphs in the MOVI and Chalet sequences. The three graphs used in these examples contain 31 (CMU), 140 (MOVI) and 40 (Chalet) nodes.



(a) (b) (c)

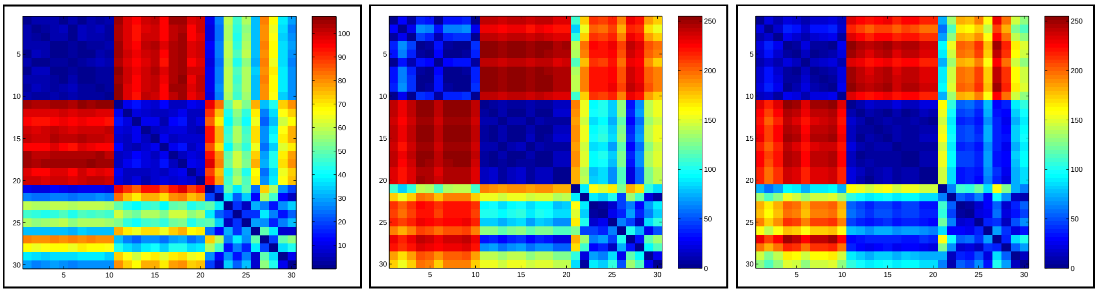
Figure 6.3: Adjacency matrices corresponding to the first image of each sequence

Next, we compare our results with those obtained when using the distances computed using two alternative methods. The first of these is the negative log-likelihood function computed using the EM algorithm reported by Luo and Hancock (Luo and Hancock, 2001). This similarity measure uses purely structural information. The method shares with our edit distance framework the use of a statistical method to compare modal structure of two adjacency matrics. However, unlike our method which uses only the leading eigenvector of the adjacency matrix, this method uses the full pattern of singular vectors of a weighted correlation of the two adjacency matrices. In addition, the method is an iterative one which alternates between computing singular vectors in the M or maximisation step and re-computing the correspondence weight matrix in the E or expectation step. The second of the distance measures is computed using a spectral emdedding of the



(a) (b)

Figure 6.4: Example Levenshtein matrices

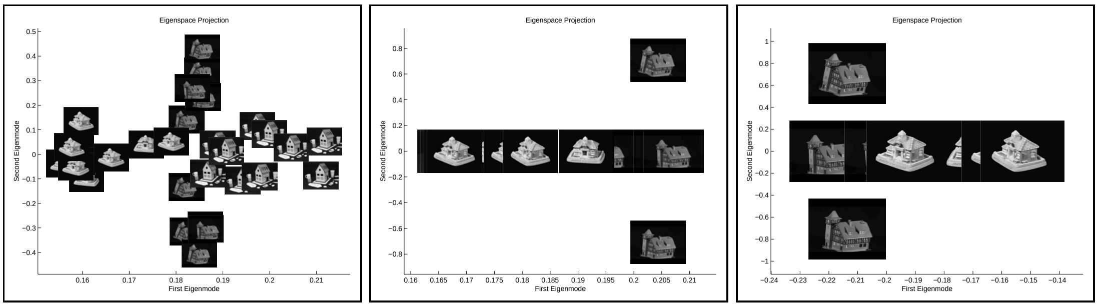


(a) (b) (c)

Figure 6.5: Edit distance matrices

graphs (Luo et al., 2002a). The method involves embedding the graphs in an pattern space spanned by the leading eigenvalues of the adjacency matrix. According to this method, the distance between graphs is simply the L2 norm between points in this pattern space (Luo et al., 2002b). This pairwise graph similarity measure again shares with our new method the feature of using spectral properties of the adjacency matrix.

In Figure 6.5a we show the distance matrix computed using our algorithm. The distance matrix obtained using the negative log-likelihood measure of Luo and Hancock (Luo et al., 2002a) is shown in Figure 6.5b, while the distance matrix computed from the spectral embedding is shown in Figure 6.5c. In each distance-matrix the element with row column indices i, j corresponds to the pairwise similarity between the graph indexed i and the graph indexed j in the data-base. In each matrix the graphs are arranged so that the row and column index increase monotonically with viewing angle. The blocks of



(a) (b) (c)

Figure 6.6: Eigenspace projections for each of the distance matrices used for purposes of view-based object recognition process

views for the diffeent objects follow one another. From the matrices in Figure 6.5, it is clear that as the difference in viewing direction increases, then the graphs corresponding to different views of the same object are more similar than the graphs for different objects. The different objects appear as distinct blocks in the matrices. Within each block, there is substructure (sub-blocks) which correspond to different characteristic views of the object.

For visualisation purposes, we have performed multidimensional scaling (MDS) on the pairwise distance matrices to embed the graphs in an eigenspace. Broadly speaking, this is a method for visualising objects characterised by pairwise distance rather than by ordinal values. It hinges around computing the eigenvectors of a similarity matrix, The leading components of the eigenvectors are the co-ordinates associated with the graphs. The method can be viewed as emdedding the graphs in a pattern space using a measure of their pairwise similarity to one another. Details of the procedure can be found in Appendix B. We plot the positions of the graphs on the plane corresponding to the two leading dimensions of the resulting eigenspace. We display two different sets of embeddings. The first consists of the embedding constructed using the entire similarity matrix for the three different objects. We do this in order to determine whether the eigenspace captures the structural differences between the sets of graphs for the distinct objects in our experimental data-set. In Figure 6.6, from left to right, we show the embeddings corresponding to the distance matrices computed using our algorithm, the negative log-likelihood computed using the algorihtm of Luo and Hancock (Luo et al., 2002a) and the spectral feature-vectors extracted from the adjacency matrix (Luo et al., 2002b). Of the three distance measures, the clusters resulting from the use of the edit distance described in this chapter produce the clearest cluster structure.

The second set of embeddings are for the different views of the same object only. Our aim here is to determine whether the eigenspace captures in a systematic way the structural differences encountered as the object undergoes changes of viewpoint. In the top row of Figure 6.7 we show the embeddings of the three objects that are obtained when we use the edit distance reported in this chapter. The embeddings corresponding to the eigenspaces computed using the negative log-likelhood (Luo et al., 2002a) and the

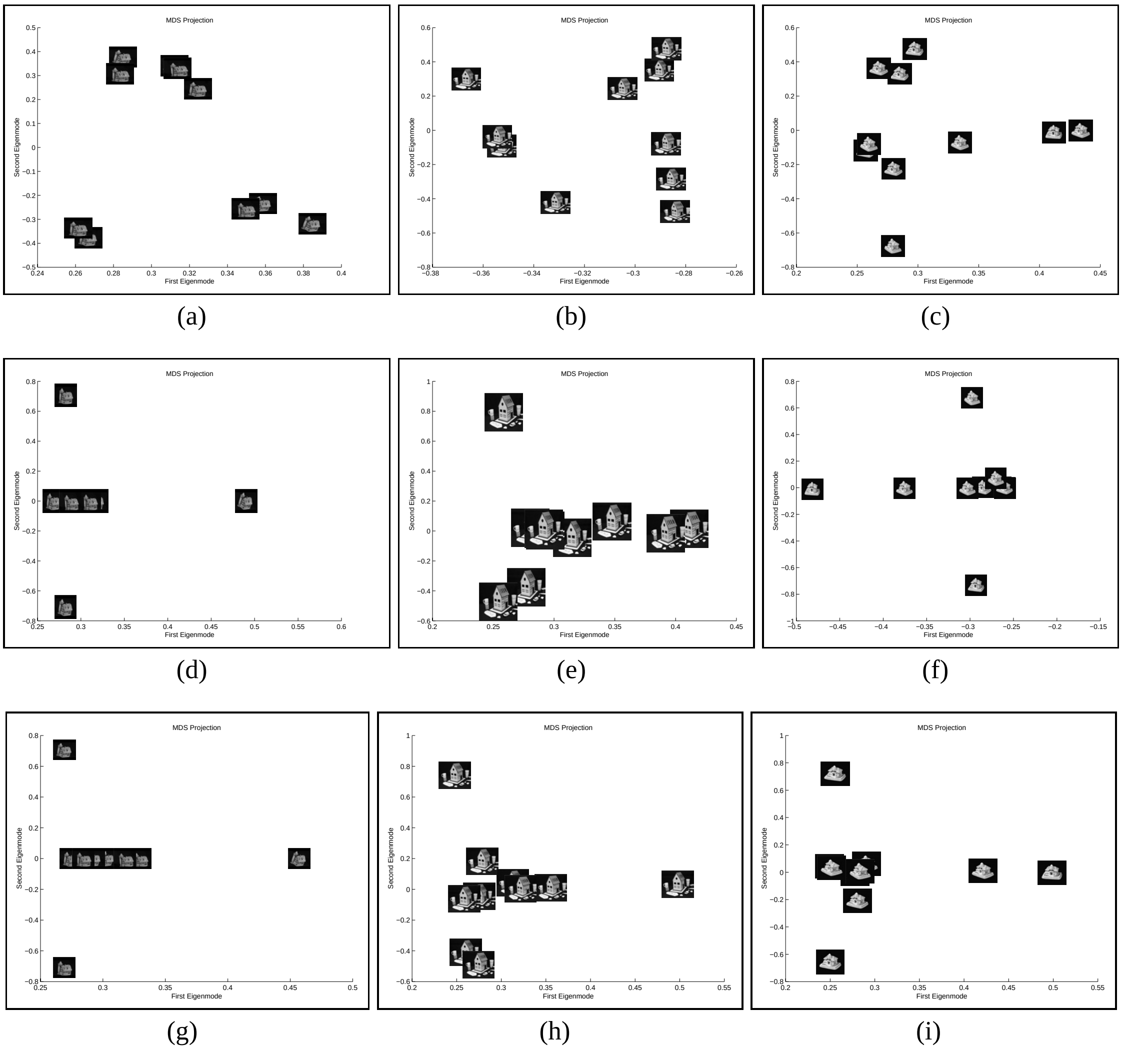


Figure 6.7: Eigenspace projections for each of the three different clusters

spectral feature vectors (Luo et al., 2002b) are shown in the middle and bottom rows. The main conclusion that can be drawn from these plots is that embeddings produced using the edit distance computation described in this chapter are better distributed than those resulting from the alternative two distance measures. This is particularly evident in the plots for the first cluster, i.e. those in left-hand column of Figure 6.7.

Hence our new distance measure appears to be effective in both distinguishing between different classes of object, and for measuring fine changes in the differences in graph structure for objects of the same class.

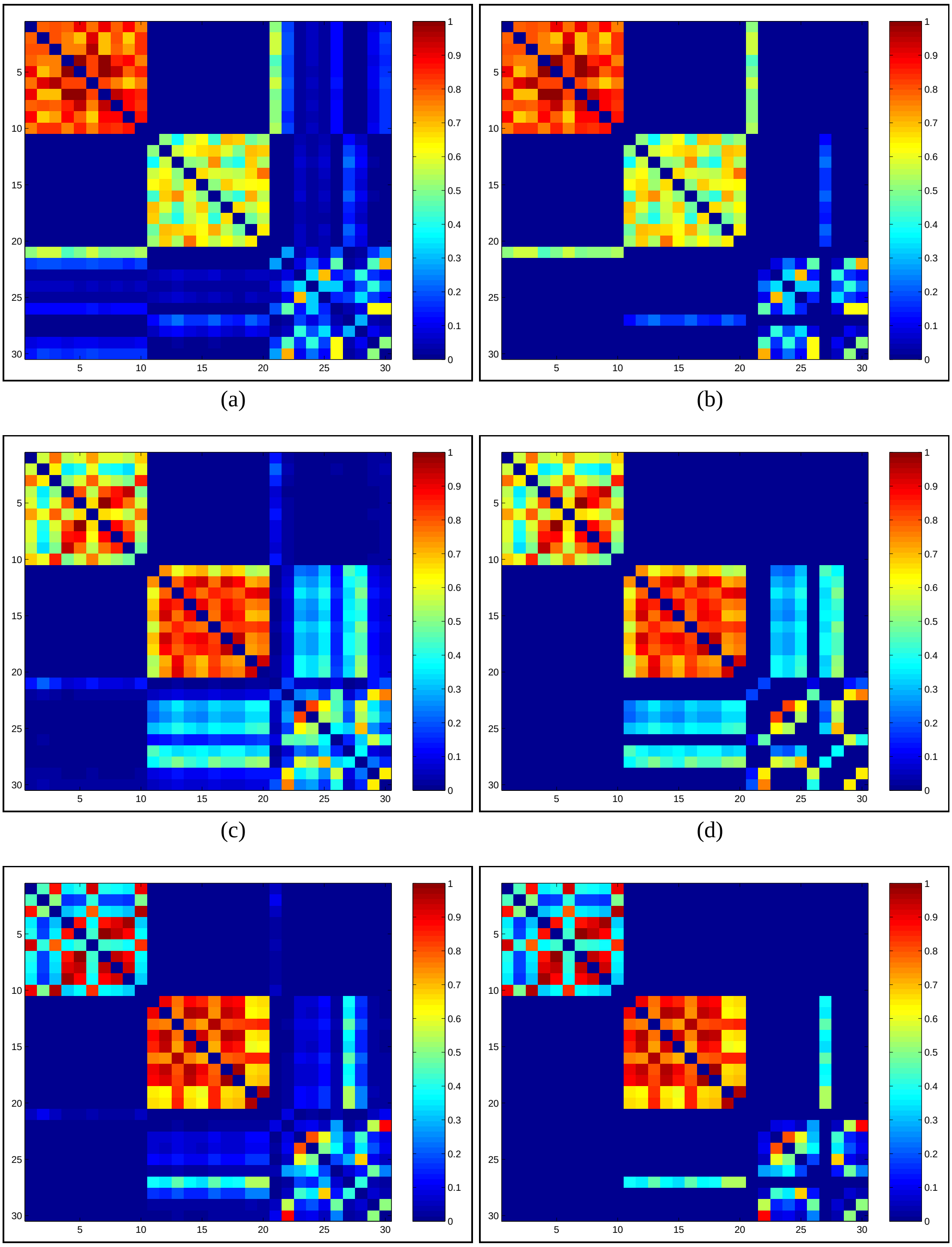
We have also applied the pairwise clustering algorithm described in Chapter 4 to the similarity data for the complete data-base of graphs. Our aim is to investigate which distance measure results in the best set of graph-clusters. Here we aim to determine which distance measure results in a cluster assignment which best corresponds to the three image sequences. The pairwise clustering algorithm requires distances to be represented by a matrix of pairwise affinity weights. Ideally. the smaller the distance, the stronger the weight, and hence the mutual affinity to a cluster. The affinity weights are required to be in the interval $[0, 1]$. Hence, for the pair of graphs indexed $i1$ and $i2$ the affinity weight is taken to be

$$W_{i1,i2} = \exp\big(-\delta \frac{d(i1, i2)}{\max(D)}\big) \tag{6.20}$$

where δ is a constant and $d_{i1,i2}$ is the element indexed $i1, i2$ of the matrix D of distances between distinct pairs of graphs (i.e the edit distance between the graph indexed $i1$ and the graph indexed $i2$). The clustering algorithm is iterative in nature and maintains two sets of variables. The first of these is a set of cluster membership indicators $s_{i\omega}^{(n)}$ which measures the affinity of the graph indexed i to the cluster indexed ω at iteration n of the algorithm. The second, is an estimate of the affinity matrix based on the current cluster-membership indicators $W^{(n)}$. These two sets of variables are estimated using interleaved update steps, which are formulated to maximise a likelihood function for the pairwise cluster configuration.

The initial and final similarity matrices generated by the clustering process for each of the distance measures studied are shown in Figure 6.8. In the top panel we show the initial and final similarity matrices obtained using our edit distance to compute the affinity matrix elements $W_{i,j}$. Similarly, the middle panel shows the affinity matrices obtained using the distances yielded by the negative log-likelihood function for the structural graph matching algorithm (SGM). Finally, the affinity matrices for the spectral feature vectors are displayed in the bottom panel of the figure.

We summarise the clustering results for the three similarity measures in Table 6.1. In contrast with the results obtained when using the structural graph matching algorithm, only two images of the Chalet sequence have been misassigned when the graph edit dis-



(a) (b)

(c) (d)

(e) (f)

Figure 6.8: Initial and final similarity matrices associated to the view-based clustering process

CMU Sequence										
Image Number	1	2	3	4	5	6	7	8	9	10
Number of Nodes	30	32	32	30	30	32	30	30	30	31
Cluster (SGM)	1	1	1	1	1	1	1	1	1	1
Cluster (MDS)	1	1	1	1	1	1	1	1	1	1
Cluster (ED)	1	1	1	1	1	1	1	1	1	1
MOVI Sequence										
Image Number	1	2	3	4	5	6	7	8	9	10
Number of Nodes	140	134	130	136	137	131	139	141	133	136
Cluster (SGM)	2	2	2	2	2	2	2	2	2	2
Cluster (MDS)	2	2	2	2	2	2	2	2	2	2
Cluster (ED)	2	2	2	2	2	2	2	2	2	2
Chalet Sequence										
Image Number	1	2	3	4	5	6	7	8	9	10
Number of Nodes	40	57	92	78	90	64	113	100	67	59
Cluster (SGM)	3	3	2	2	2	3	2	2	3	3
Cluster (MDS)	4	3	3	3	3	3	2	3	3	3
Cluster (ED)	1	3	3	3	3	3	2	3	3	3

Table 6.1: Clustering results statistics

tance is used. This is an improvement on the results obtained with the log-likelihood function of Luo and Hancock, and comparable to the results obtained with the spectral feature vectors. The errors are due to the fact that the graphs in question are morphologically more similar to the graphs in the CMU and MOVI sequence than to those in the Chalet sequence. Nevertheless a success rate of 93.3% is obtained when edit distance is used for the affinity matrix computation. Our method compares favourably when computational cost is taken into account. This is because our method only requires the computation of the leading eigenvector while the spectral feature vectors require the complete eigenstructure to be computed. In consequence, the edit distance takes on average 2.6 times less time to compute than the spectral feature vectors.

6.4.2 Large Image Database

The second application involves using the clustering methods outlined in the previous section for image database indexing and retrieval. We have performed experiments on a database containing 245 binary images of trademark-logos [1] used previously in the study

[1] All trademarks and logotypes remain the property of their respective owners. All trademarks and registered trademarks are used strictly for educational and academic purposes and without

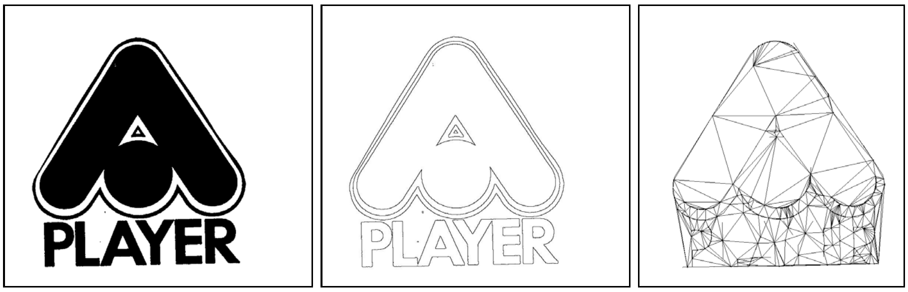


Figure 6.9: Example image, polygonalisation results and Delaunay triangulation

of Huet and Hancock (Huet and Hancock, 2002). Here, the graphs are constructed as follows. First, we apply the Canny edge detector (Canny, 1986) to the images to extract connected edge-chains. A polygonalisation procedure (Peng-Yeng, 1998) is applied to the edge-chains to locate straight line segments. For each line-segment, we compute the centre-point. The graphs used in our experiments are the Delaunay triangulations of the line-centres. In the top left-hand-panel in Figure 6.9, we show an example of the images used in our experiments. The middle panel shows the results of the line-segment extraction step. The corresponding Delaunay graph is displayed in the right-hand-panel. For each pair of Delaunay graphs we compute the edit-distance. We have again applied both multidimensional scaling and pairwise clustering to the distance matrix.

Turning our attention first to the results of multidimensional scaling Figure 6.10 shows the distribution of graphs in the space spanned by the leading two eigenvectors of the similarity matrix. The graphs are distributed along an anulus.

Next, we turn our attention to the results of applying pairwise clustering to the set of edit distances for the logos. We obtained 34 clusters, with an average of 7 images per cluster. To display the results, we find the modal graph for each cluster. For the cluster indexed ω the modal graph has the largest cluster-membership indicator $s_{i\omega}^{(\infty)}$ at convergence of the clustering process. In other words it is the graph with index $i_\omega^* = \arg\max_i s_{i\omega}^{(\infty)}$. We then rank the remaining graphs in the order of their increasing distance from the modal graph for the cluster ω. The significance of the cluster indexed ω is

intent to infringe on the mark owners.

gauged by the total mass of membership indicator $m_\omega = \sum_{i\in} s_{i\omega}^{(\infty)}$. In Figure 6.11, the different rows show the images belonging to the eight most significant clusters. The left-most image in each row is the image corresponding to the modal graph for the cluster. The subsequent images in each row correspond to the most similar graphs in order of increasing edit distance It is interesting to note that similar logos appear in the same cluster. For instance, the two "Crush" logos (one with a palm-tree and one with a fruit-segment) appear in the top row, the "Hotel Days", "Alberge Daystop" and "Les Suites Days" appear in the second row, and, the two "Incognito" logos in the third row. However, the clusters do not seem to correspond to obvious shape catgories. Hence, it would appear that we need to exploit the distances in a more sophisticated shape indexation procedure.

To this end, we note that once the database has been clustered, then we can use the

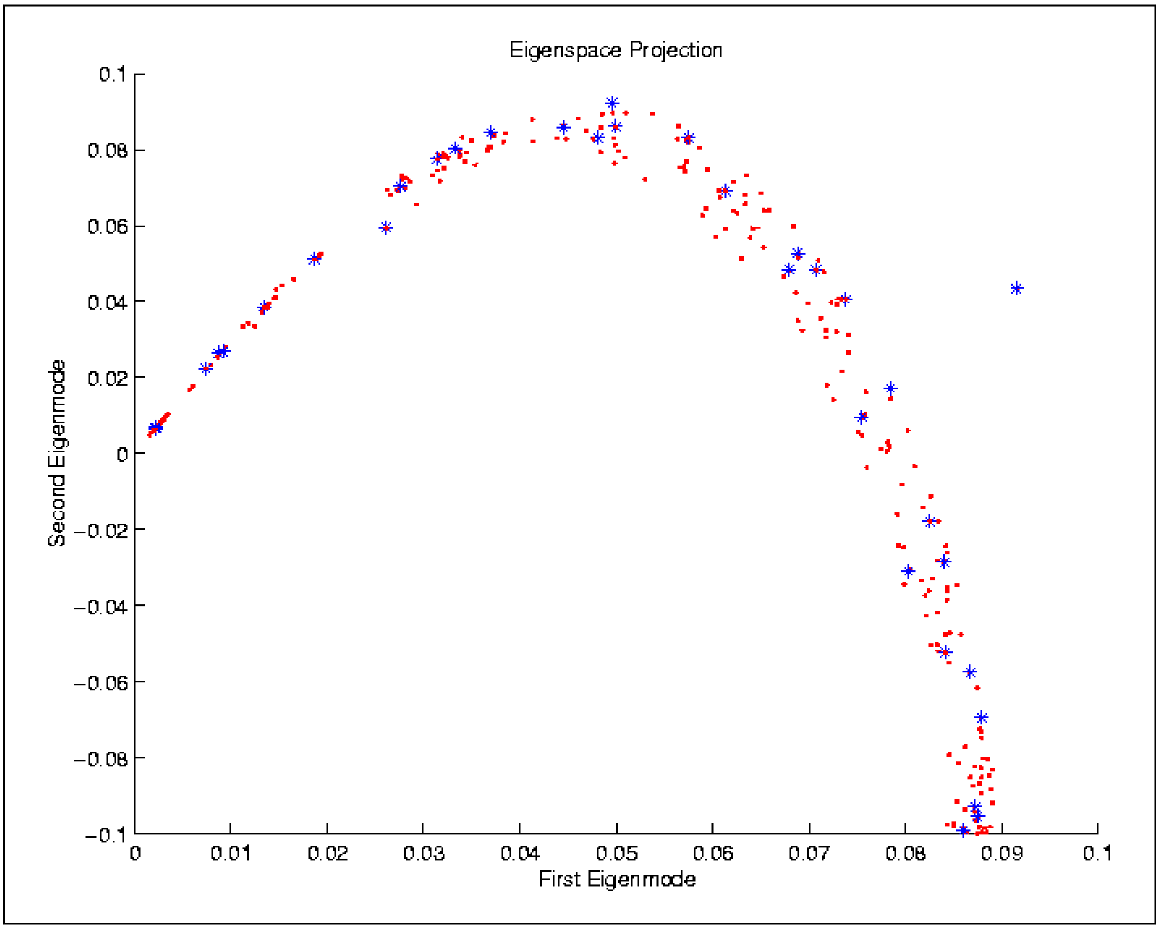


Figure 6.10: Multidimensional scaling applied to the matrix of edit distances for the logos

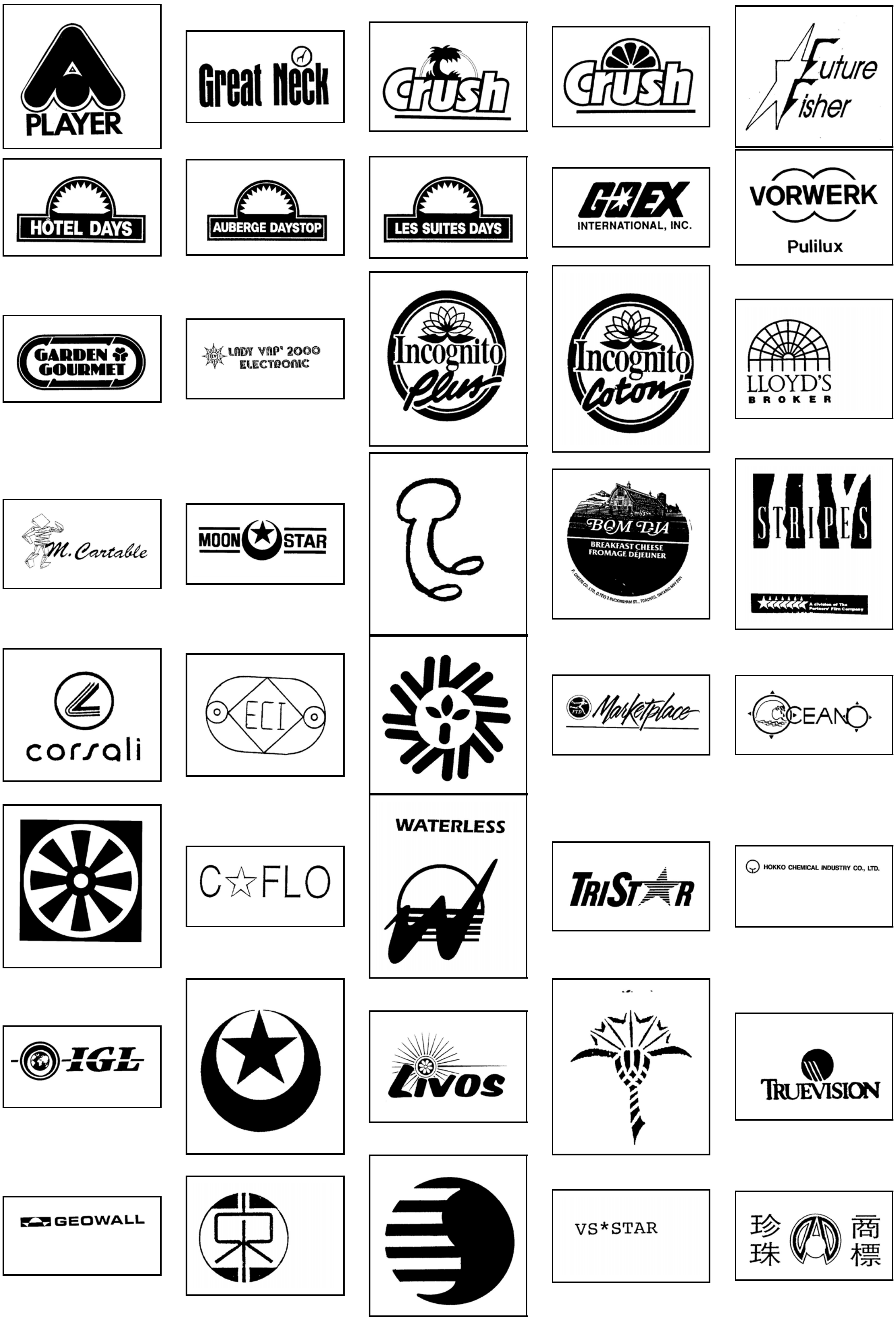


Figure 6.11: Left-hand column: Cluster-centers; Right-hand columns: Members of the cluster arranged according to their rank.

cluster structure to improve the efficiency of retrieval, i.e. the search for the most similar object. The idea is to find the cluster whose modal graph is most similar to the query. The graph within the cluster that is most similar to the query is the one that is retreived. The search process is as follows. Suppose the query graph is denoted by G_q. First, we compute the set of graph edit distances between the graph G_q and the modal graphs for each cluster. The distance between the graph G_q and the modal graph for the cluster ω is denoted by $d^*_{q\omega}$. The cluster with the most similar modal graph is $\omega_q = \arg\min_\omega d^*_{q\omega}$. We search the graphs in this cluster to find the one that is most similar to the query graph. The set of graphs belonging to the cluster ω_q is $C_q = \{i|s^{(\infty)}_{i\omega_q} = \arg\max_\omega s^{(\infty)}_{i\omega}\}$. The

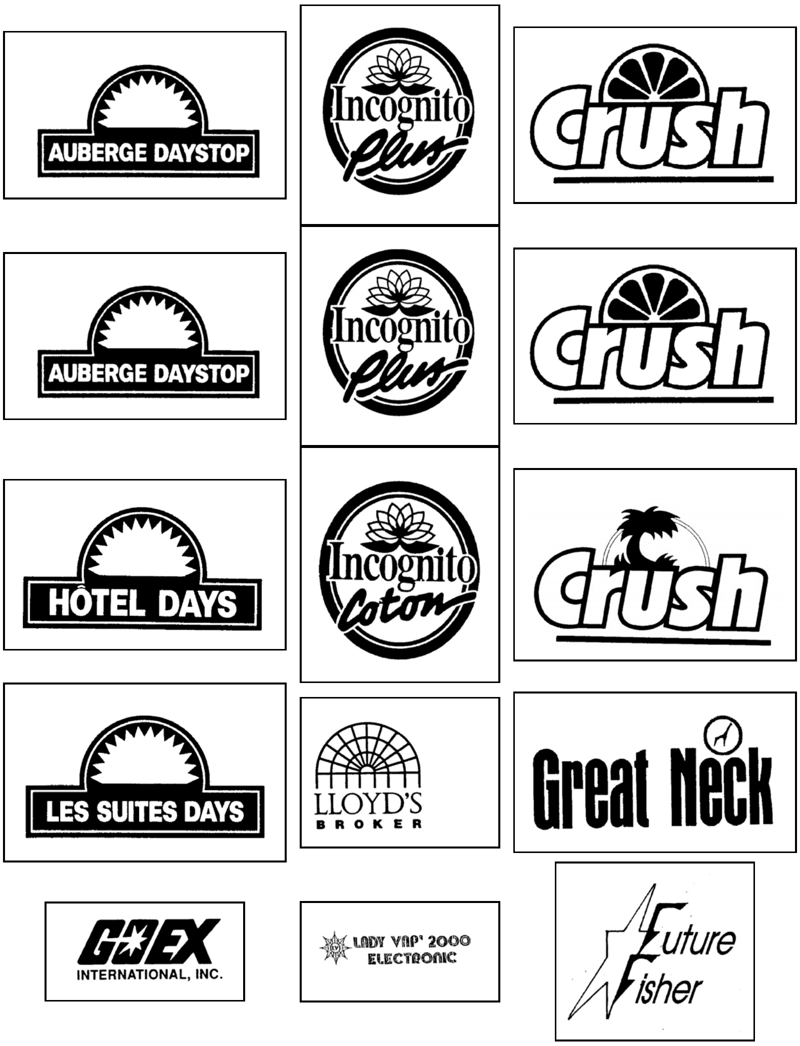


Figure 6.12: Top row: input query images; Bottom rows: search results.

retrieved graph is the member of the set C_q which has the minimum edit distance to the query graph G_q, i.e. the one for which $i_q = \arg\min_{i \in C_q} d_{qi}$.

In Figure 6.12, we show the results of two query operations. The top image in each column shows the query image and the subsequent images correspond to the retrieved images in order of increasing graph edit distance. In the left-hand column we show the result of querying with the "Auberge Daystop" logo. The two most similar images have the same shape, but carry the legends "Hotel Days" and "Les Suites Days". The middle row shows the result of querying with the "Incognito-plus" logo, which returns the "Incognito-coton" logo as the most similar graph. In the right-hand column we show the result of querying with the "Crush" logo. Here the most similar graph again contains the "Crush" legend, but the orange segment is replaced by a palm-tree. It is worth noting that the database contains only two logos of the "Days Inn" type and a single "Crush" and "Incognito" type image. Hence, from our results, we can conclude that the algorithm is able to cope with structural variations and differences in graph-size.

6.4.3 Synthetic Data

Finally, we have conducted some experiments with synthetic data and compared our algorithm with a number of alternatives. To do this, we have generated random point-sets and have constructed their Delaunay graphs. We have simulated the effects of structural errors by randomly deleting nodes and re-triangulating the remaining point-set. For all the results reported in this section, we have averaged the edit distance over 10 different sets of random node deletions.

We commence by illustrating the effects of random structural errors. To this end, in Figure 6.13, we show the edit distance as a function of the number of deleted nodes. The different curves in the plot are for each of the Delaunay triangulations corresponding to the first frame of the image sequences used in Section 6.4.1. In each case, the edit distance varies almost linearly with the number of deleted nodes. The deviations from the linear dependence occur when large fractions of the graph are deleted.

Turning our attention to the second aspect of our study, we illustrate the performance of our algorithm compared with a number of alternatives. The alternative algorithms used in our study are the Wilson and Hancock discrete relaxation scheme (Wilson and Hancock, 1997), the Gold and Rangarajan (Gold and Rangarajan, 1996) quadartic assignment method and the Finch, Wilson and Hancock (Finch et al., 1998) non-quadratic assignment method. In Figure 6.14, we show the fraction of correct correspondences as a function of the fraction of nodes deleted for the graph corresponding to the first frame of the "CMU-VASC" image sequence. The performance curve for our new method (marked as "Evidence combining" on the plot) is shown as the lightest of the curves. Also shown on the plot are performance curves for the Wilson and Hancock discrete relaxation scheme, (Wilson and Hancock, 1997), the Gold and Rangarajan (Gold and Rangarajan, 1996) quadratic assignment method and the Finch, Wilson and Hancock (Finch et al., 1998) non-quadratic assignment method. In the case of random node deletion, our method gives performance

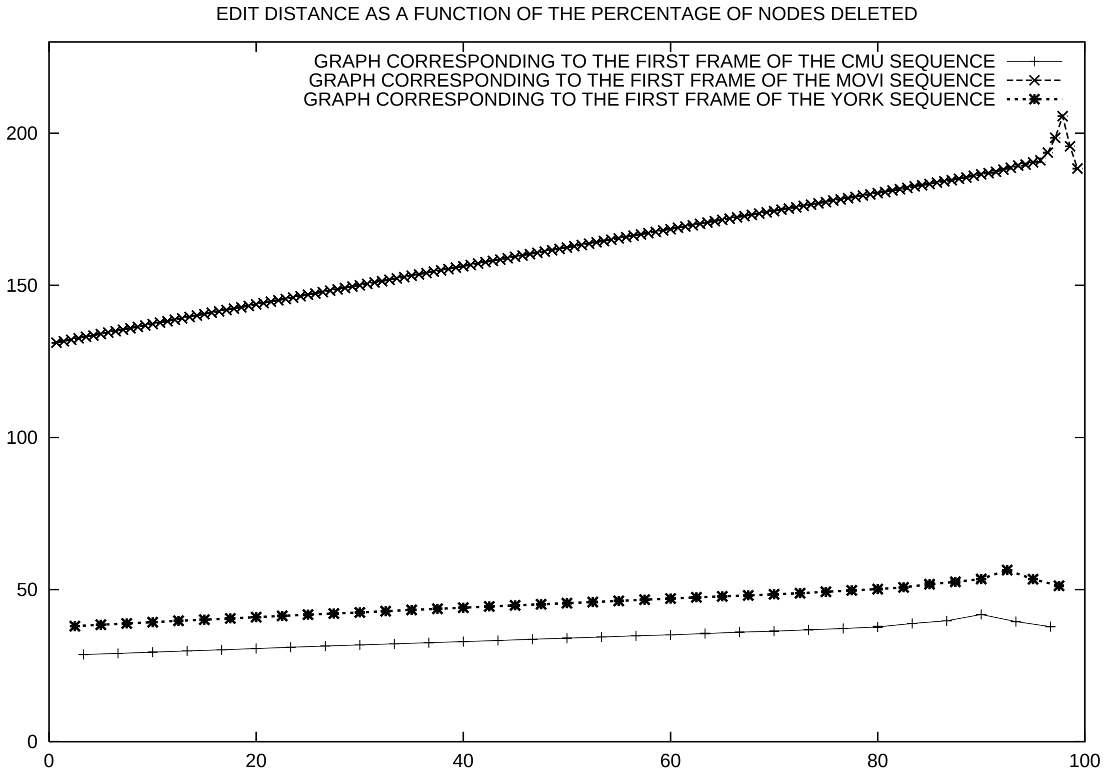


Figure 6.13: Graph edit distance as a function of the percentage of nodes deleted for the first frame of the image sequences used in Section 6.4.1.

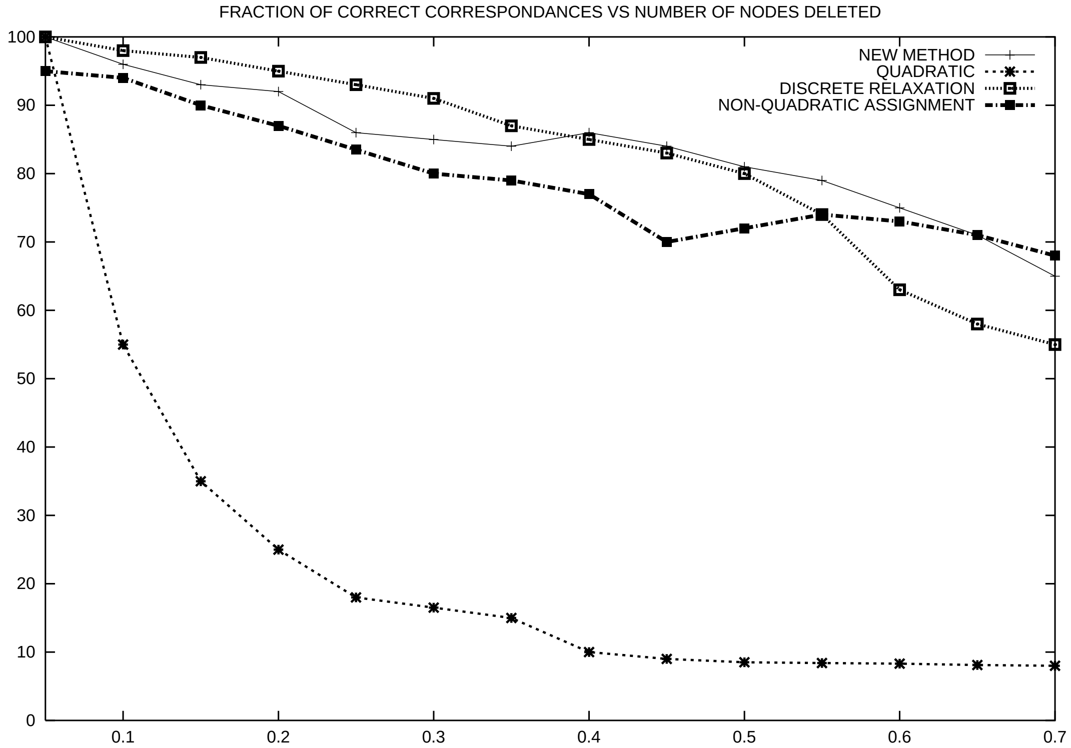


Figure 6.14: Plot of the Error as a function of the fraction of nodes deleted for our algorithm and a number of alternatives.

that is significantly better than the Gold and Rangarajan method, and intermediate in performance between the discrete relaxation and non-quadratic assignment methods.

6.5 Conclusions

The work reported in this chapter provides a synthesis of ideas from spectral graph-theory and structural pattern recognition. We use a graph spectral seriation method based on the leading eigenvector of the adjacency matrix to convert graphs to strings. We match the resulting string representations by minimising edit distance. The edit costs needed are computed using a simple probabilistic model of the edit transitions which is designed to preserve the edge order on the correspondences. The minimum cost edit sequence may be used to locate correspondences between nodes in the graphs under study.

We have demonstrated that both the edit sequence and the associated distance are of practical use. The edit sequence delivers correspondences that are robust to structural error, and compares favourably with those delivered by a number of alternative graph-matching algorithms. The edit distances can be used to cluster graphs into meaningful classes.

Chapter 7

Contributions and Future Work

The aims in this chapter are twofold. Firstly, we aim to summarise the main contributions of this thesis. These include the two novel algorithms for performing segmentation and grouping, the shape-from-shading algorithm and the graph edit distance approach. Secondly, we shall discuss several possible extensions and further development of the research presented in this thesis.

7.1 Contributions

The contributions of the thesis are both theoretical and practical. The theoretical contributions have been to

a) Cast eigenvector clustering methods into a probabilistic setting by using the EM algorithm in Chapter 3.

b) Show how eigenvector analysis of the log-likelihood function can be used to reduce noise through a process of "modal sharpening" (Chapter 4).

c) Use the Perron-Frobenius theorem to show how maximally edge connected paths can be located using the leading eigenvector of the transition matrix (Chapter 5).

d) Estimate graph edit costs using a probabilistic model of transitions on the edit lattice (Chapter 6).

The practical contributions have been to use spectral-graph theory to

a) Develop new methods for line-segment grouping, gray-scale image segmentation and motion analysis in Chapters 3 and 4.

b) Develop a new framework for shape-from-shading (Chapter 5).

c) Develop new methods for computing the graph edit distance and performing inexact graph-matching in Chapter 6.

Specific details concerning the methods presented in the thesis are described in the following subsections.

7.1.1 Statistical Methods for Spectral Clustering

In Chapter 3, our first step was to develop a graph-spectral algorithm for segmentation and grouping based upon the EM algorithm. We have cast the segmentation process in a pairwise clustering setting, where the maximum likelihood of the elements of the link-weight matrix was recovered using the EM algorithm. This is, by itself, a unique feature of our grouping algorithm. Another important feature, that sets apart our algorithm from those in the literature, is the probabilistic characterisation of the grouping graph using a link-weight matrix and a set of cluster-membership variables. Given these ingredients, we obtained a joint likelihood function that could be used for updating both, the cluster-membership variables and the link-weight matrix. This dual-step estimation scheme is one of the unique features of our algorithm when compared to those proposed by Hofmann and Buhmann (Hofmann and Buhmann, 1997) and Shi and Malik (Shi and Malik, 1997). To overcome the initialisation problems common to a wide variety of EM approaches, we initialised the cluster-membership variables using the leading eigenvector of the adjacency matrix. We did this in order to exploit the relation between the cluster cohesiveness and the leading eigenvector of the link-weight matrix proposed by Sarkar and Boyer (Sarkar and Boyer, 1998). Since the initialisation is an optimal one, the algorithm exhibited good convergence properties.

Another contribution of Chapter 3 is the development of a probabilistic grouping field following ideas by Heitger and von der Heydt (Heitger and von der Heydt, 1993). Although grouping fields are proposed elsewhere in the literature (Williams and Jacobs, 1997b; Williams and Jacobs, 1997a), the distinctive characteristic of our approach is the use of an elongated polar function for modeling the linking process. As a result, the cost of computing the linking probability was low compared to competing approaches. With the initial line-grouping field to hand, we proceeded to illustrate the utility of our algorithm for line-grouping and provided a performance analysis of the results. We also performed experiments on gray-scale image segmentation.

In Chapter 4, we took the statistical formulation of the segmentation and grouping problem a step further and explored the relation between the log-likelihood function and the eigenvalues and eigenvectors of the adjacency matrix. This treatment provided interesting results concerning the modal structure of the log-likelihood function and the convergence of the estimation process. One of the main results arising from the modal analysis of log-likelihood function was that the leading eigenvector is the maximiser of the joint likelihood function. Further, using monotonicity principles, we also proved that the leading eigenvalue can be used to determine whether convergence has been reached. To conclude the chapter, we applied our algorithm to line-segment grouping and developed a hierarchical motion segmentation system. A comparison with competing algorithms showed that our method provides a good margin of improvement over the normalised cut algorithm (Shi and Malik, 1997).

7.1.2 Shape-from-shading

In Chapter 5, we turned our attention to shape-from-shading. The method developed differs from those elsewhere in the literature in several aspects. For instance, in contrast with other shape-from-shading algorithms, where the problem is posed as one of regularisation, our approach makes use of spectral-graph theory to recover the surface height and to segment surfaces into piecewise-smooth patches. To do this, we characterised the field of surface normals using a transition matrix whose elements are computed by employing the sectional curvature between different image locations. We posed the problem of recovering a surface integration path as that of maximising the sum of curvature weights. We demonstrated that the leading eigenvector of the adjacency matrix is the solution to sequencing problems. As a result, a minimum curvature surface integration path can be located using the rank-order of the coefficients of the leading eigenvector of the transition matrix.

With the minimum curvature path to hand, we performed surface integration by incrementing the height function while traversing the path. We did this by applying the trapezium rule to the slant and tilt angles of the surface normals. To perform the surface normal adjustment and smoothing steps, we fitted quadric patches to the height data. The gradient of the fitted surface was used to update the direction of the surface normals on their irradiance cones. The processes of height recovery and surface normal adjustment were interleaved and iterated until a stable surface was obtained.

To investigate the robustness of our shape-from-shading algorithm to brightness errors, we performed a sensitivity study. The study showed that the algorithm yields good results even when a considerable amount of noise is present. We also illustrated the utility of our shape-from-shading algorithm on a variety of synthetic and real-world imagery.

The advantages of our shape-from-shading algorithm over existing regularisation and level-set approaches are twofold. First, since the algorithm performs surface-height recovery in a modular manner, this is no longer an integral part of the shape-from-shading algorithm and can be used for integrating surfaces from their Gauss maps. Second, as a result of the use of quadric patches for updating the surface normals, the smoothing step does not over-smooth the patch boundaries.

7.1.3 Graph-matching and Graph Edit Distance

Finally, in Chapter 6, we showed how the eigenstructure of the adjacency matrix can be used for the purposes of robust graph-matching. Making use of the spectral approach to graph seriation developed in Chapter 5, we mapped pairs of graphs onto strings. We did this using the rank-order of the leading eigenvector of the adjacency matrix.

Once the graphs were mapped onto strings, the edit distance was computed by finding the sequence of string edit operations that minimise the cost of the path traversing an edit lattice. The edit costs were computed using a probabilistic model of transitions on the edit lattice. The approach taken to finding the minimum cost path on the edit lattice is closely related to the solution of the shortest path problem developed by Dijkstra (Dijkstra, 1959). We concluded the chapter by showing results on graph clustering. A comparison of the clustering results with those obtained using alternative distance measures revealed that the graph edit distance gave results that compare favourably with more computationally demanding methods.

At this point, it is worth pausing to note that the path-based analysis of graphs presented in Chapters 5 and 6 has a number of interesting theoretical features. First, it provides an insight into closely related combinatorial problems known to pose exponential complexity. Problems of this sort arise when solving the “traveling salesman” or DNA sequencing problems. Second, it provides proof of concept spectral methods operating on one or more symmetric, positive, semidefinite matrices may be employed to solve problems that involve complex sorting operations.

7.2 Future Work

There are a number of ways in which the work presented in this thesis can be extended and further developed. For instance, in Chapter 3 and 4, the segmentation and grouping algorithms are reminiscent of relaxation approaches. In our experiments, the initial link-weight matrix was computed directly from image features and no attempt was made to impose data closeness in the clustering process. In many practical applications this is not desirable. Hence, the inclusion of a data-closeness term is important in order to guarantee the interaction between the data and the clustering process.

The treatment of the joint likelihood function developed in Chapter 4 can be extended in a number of ways. First, the convergence analysis can be applied to alternative probability distributions (rather than that arising from the assumption of independent Bernoulli trials). Further refinement of the maximum likelihood framework may be achieved by using variational inference techniques (Jordan, 1998). This is an attractive alternative which facilitates the computation of posterior probability distributions while overcoming the lack of data closeness with the raw link-weight matrix.

In Chapter 5, maybe the most unrealistic of the assumptions made while developing the shape-from-shading algorithm was that of a Lambertian reflectance. Hence, there is considerable scope for improvement by extending the Lambertian reflectance model to accommodate effects such as local specularity, coherent limb brightening and rough surface scattering. A second future direction could be to incorporate information concerning the local minimum curvature directions. In doing this, it will be possible to facilitate coupling between the minimum curvature path and the minimum curvature direction.

There are a number of ways in which the work on graph edit distance may be improved. First, although we have concentrated on unweighted graphs, the method can be easily extended and applied to weighted graphs. The method has a variety of possible applications that are worth exploring. These include, but are not limited to, pose estimation, and, the structuring and retrieval of large image databases. In order to achieve the robustness and accuracy required for these tasks, it is also important to improve the computation of the minimum cost path. Despite proving effective, the current approach cannot identify multiple solutions to ambiguous labelling problems. Hence, we intend to use population based methods such as genetic algorithms to compute the minimum cost path on the edit lattice.

Another topic that merits further investigation is the possibility of incorporating learning into the graph matching task. The aim would be to learn the cost of the edit operations (i.e. transitions on the edit lattice) from a training set. Finally, since the main limitation of the experimental study was the relatively small amount of data used, it would be interesting to extend the study to a database comprising thousands of graphs.

Appendix A

A Fast Leading Eigenvector Approximation for Segmentation and Grouping

One of the criticisms that can be leveled at the spectral-graph methods for segmentation and grouping developed in Chapters 3 and 4 is that they are computationally demanding because they rely on the numerical determination of eigenvectors of large matrices. The problem of computing the eigenvalues and eigenvectors of a matrix is one of classical linear algebra which arises in many practical problems in science and engineering. However, it is frequently the determination of the leading eigenvector which turns out to be of pivotal importance. The reason for this is that it is intimately related to the Raleigh Quotient (Sarkar and Boyer, 1998; Weiss, 1999).

When only the first few eigenvalues and eigenvectors of a matrix are required, then the most commonly used methods are those based on Krylov information (Wilkinson, 1965). Among these, the most well known are the power method and, for symmetric matrices, the Lanczos method. These methods are iterative, and hence need to be initialised with a starting vector. Moreover, it can be proven that the rate of convergence of the Lanczos method changes in accordance with the error criterion used (Del Corso, 2000). Hence, the efficiency of the method is dependant on how distant the extreme eigenpair is from the others.

Other approximation approaches include the Nyström method (Nyström, 1928) and matrix perturbation theory (Stewart and guang Sun, 1990). The Nyström method has been applied to a variety of problems, including grouping (Fowlkes et al., 2001) and solving integral equations with singular kernels (Press et al., 1992). The perturbation theory of eigenvalues and eigenvectors has been researched thoroughly, and the bounds of a single eigenvalue have been set extremely accurately using Gerschgorin's Theorem.

In this appendix, we present a leading eigenvector approximation that aims to render spectral-graph methods for segmentation and grouping more computationally efficient. To do this, we make use of matrix perturbation theory and assume the matrix upon which the algorithm operates is large, dense, real, non-negative and symmetric.

A.1 Approximation via Matrix Perturbation

Perturbation methods attempt to solve a complex problem approximating it by a simpler problem that is reasonably well known. We will use the perturbation approach to approximate the first eigenvector of a large, real, non-negative symmetric matrix A to a linear combination of perturbation matrices whose spectral properties are well known. In segmentation and grouping this prerequisites are not prohibitive since the matrix A (i.e. the adjacency matrix) is often used to capture the pairwise similarity between two image tokens. Hence, it is symmetric in nature and its elements are real and non-negative .

For computational purposes, we will constrain the elements of the matrix A to be between 0 and 1. This is a scaling step that will not affect the generality of the method.

We commence expressing the matrix A of order m as a linear combination of perturbation matrices $\hat{A}_i$ as follows

$$A = \frac{1}{m} \sum_{i=1}^{m} \hat{A}_i + H \tag{A.1}$$

where matrix H is an error matrix.

For our analysis, we require the perturbation matrices $\hat{A}_i$ to be real, non-negative symmetric matrices proportional to $\mathcal{X}_i \mathcal{X}_i^T$, where $\mathcal{X}_i = [\chi_{i,1}, \ldots, \chi_{i,m}]^T$ is a real, non-negative vector of order m normalized to satisfy the condition

$$\sum_{k=1}^{m} {\chi_{i,k}}^2 = 1 \tag{A.2}$$

Providing that these requirements are satisfied, it can be shown that

$$\hat{A}_i = \eta_i \mathcal{X}_i \mathcal{X}_i^T \tag{A.3}$$

where $\hat{\eta}_i$ is the perturbation parameter for the matrix $\hat{A}_i$. It is relatively easy to prove from the theory of non-negative symmetric matrices (Varga, 2000) that the vector $\mathcal{X}_i$ is equal to the first eigenvector x* of A if $A = \hat{A}_1$. Moreover, it can be shown that, in such cases, x* can be obtained in a closed form at a cost of $\mathcal{O}(2\mathrm{m})$; $\mathcal{O} = 1$.

In order to obtain the vector $\mathcal{X}_i$ we re-write Equation (A.3) using the expansion

$$\hat{A}_i = \begin{pmatrix} (A_{1,1})_i & \cdots & (A_{1,m})_i \\ \vdots & \ddots & \vdots \\ (A_{m,1})_i & \cdots & (A_{m,m})_i \end{pmatrix} = \eta_i \begin{pmatrix} (\chi_{i,1})^2 & \cdots & \chi_{i,1}\chi_{i,m} \\ \vdots & \ddots & \vdots \\ \chi_{i,m}\chi_{i,1} & \cdots & (\chi_{i,m})^2 \end{pmatrix} \tag{A.4}$$

After some algebra we find that

$$\eta_i \chi_{i,j} = \sqrt{\sum_{k=1}^{m} {(A_{j,k})_i}^2} \tag{A.5}$$

So far, we have limited ourselves to the properties of $\hat{A}_i$. We now turn our attention to the task of approximating the leading eigenvalue and eigenvector (i.e. the leading eigenpair) of the matrix A using the vectors $\mathcal{X}_i$ and the perturbation parameters η_i. Substituting Equation (A.3) into Equation (A.1) yields

$$A = \frac{1}{m} \sum_{i=1}^{m} \eta_i \mathcal{X}_i \mathcal{X}_i^T + H \tag{A.6}$$

The above expression is very similar to the spectral decomposition of the matrix A. The vectors $\mathcal{X}_i$ are closely related to the first eigenvector of A. Even though the leading eigenvector of matrix A and the vector $\mathcal{X}_i$ are not equal, the contribution of the former to the latter is proportional to η_i.

When the off-diagonal elements A are all non-zero then H will be a diagonal matrix. As a result, it is easy to show that when the matrix A is dense, then H will be approximately diagonal in structure. For large dense matrices, we can treat H as a diagonal matrix with little loss of accuracy.

Using the spectral expansion of A together with the properties of symmetric non-negative matrices (Varga, 2000; Berman and Plemmons, 1994), when A is not diagonally dominant then it follows that

$$A \approx \lambda^* \mathrm{x}^* \mathrm{x}^{*T} \tag{A.7}$$

where λ^* and $\mathrm{x}^* = [\mathrm{x}^*(1), \ldots, \mathrm{x}^*(m)]^T$ are the leading eigenvalue and eigenvector of A.

In practice, the requirement that A is not diagonally dominant is not restrictive. The reason for this is that

$$[A + I]\mathrm{x}^* = (\lambda^* + 1)\mathrm{x}^* \tag{A.8}$$

This property also opens-up the possibility of processing other canonical forms, such as the normalized Laplacian.

From Equations (A.4) and (A.5), and using the approximation in Equation (A.7), we get

$$\chi_{i,j} = \frac{A_{i,j}}{\sqrt{\sum_{k=1}^{m} {A_{k,j}}^2}}, i \neq j \tag{A.9}$$

where $A_{i,j}$ is the element indexed i, j of matrix A.

If there is no dominant perturbation parameter η_i and if the matrix A is large, then from Equations (A.7) and (A.9) we have

$$\mathrm{x}^*(i) \approx \frac{1}{m-1} \sum_{j=1;j\neq i}^{m} \chi_{i,j} \tag{A.10}$$

Expanding the right-hand side of Equation (A.10), including the elements $H_{i,j}$ of

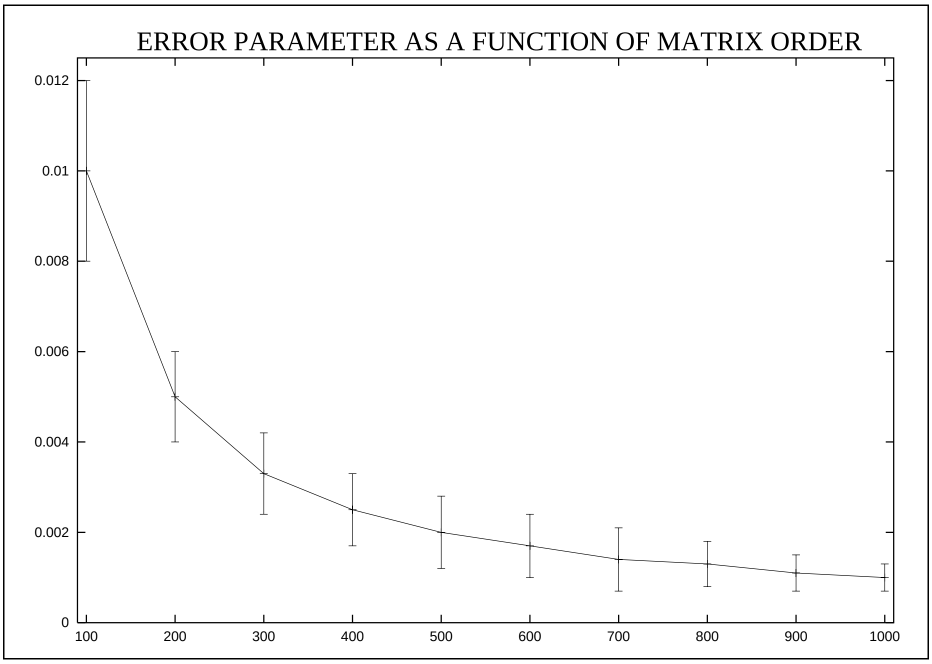


Figure A.1: Effect of m on δ

matrix H, and assuming H to be diagonal, we find that

$$\begin{aligned}\frac{1}{m-1}\sum_{j=1;j\neq i}^{m}\chi_{i,j} = \frac{1}{m-1}\Bigg\{&\frac{A_{i,1}}{\sqrt{\sum_{k=1}^{m} {A_{k,1}}^2 + (A_{1,1}-H_{1,1})^2}} \\ &+\ldots+\frac{1}{m-1}\frac{A_{i,m}}{\sqrt{\sum_{k=1}^{m} {A_{k,m}}^2 + (A_{m,m}-H_{m,m})^2}}\Bigg\}\end{aligned} \tag{A.11}$$

Hence, the accuracy of the approximation will be closely related to the diagonal elements of A. More precisely, it is governed by terms of the form $(A_{i,i} - H_{i,i})^2$.

A less computationally expensive estimate of the approximation accuracy is given by the norm condition of the eigenvector. Hence, we define the accuracy variable δ to be

$$\delta = \mid 1 - \sum_{i=1}^{m} \underline{\mathrm{x}}^*(i)^2 \mid \tag{A.12}$$

where x_i is the ith element of the approximated first eigenvector.

A.2 Pseudocode

Having presented the theoretical prerequisites, we can develop a practical algorithm for computing an approximation to the first eigenvector of the matrix A. The pseudo-code is as follows

```
For $i = 1 \ldots m$
    $\hat{x}_i = 0$;
    For $j = 1 \ldots m$
        If $(i \neq j)$ $\mathrm{x}^*(i) = {A_{i,j}}^2 + \mathrm{x}*(i)$;
For $i = 1 \ldots m$
    $\mathrm{x}^*(i) = \sqrt{\mathrm{x}^*(i)}$;
For $i = 1 \ldots m$
    For $j = 1 \ldots m$
        If $(j \neq i)$ $A_{i,j} = \frac{A_{i,j}}{\mathrm{X}^*(i)}$;
    If $(j = i)$ $A_{i,j} = \mathrm{x}^*(i)$;
```

continued on next column

continued from previous column

```
For $i = 1 \ldots m$
    $\sigma = 0$;
    For $j = 1 \ldots m$
        If $(i \neq j)$ $\delta = \delta + A_{i,j}$;
    $x_i = \delta$;
For $i = 1 \ldots m$
    $\mathrm{x}^*(i) = \frac{\mathrm{X}^*(i)}{m-1}$;
$\sigma = 0$;
For $i = 1 \ldots m$
    $\delta = \delta + \mathrm{x}^*(i)^2$;
```

There are two important practical considerations underpinning this algorithm. First, in order to reuse memory, the matrix A is overwritten at output and δ has been used more than once as a variable. Second, the sequence of operations has been slightly altered in order to reduce the number of floating point operations. The complexity of the algorithm is $4m^2$. The memory requirement, assuming matrix A has already been allocated, is $m+3$ memory locations.

A.3 Error and Performance Analysis

Our aim in this section is to explore the effects of varying the size m and sparseness of the matrix A on our approximate eigenvector calculation.

For our sensitivity study we have generated matrices whose elements are normally distributed random variables with mean 0.5 and variance 0.1667. The variance has been chosen in order to ensure $A_{i,j} \in [0, 1]$. It can be proven that, as a consequence of the normal distribution of the matrix elements, a modification of the variance would not alter the parameter δ.

First, we investigate the effect of varying the matrix order m. To do this we have generated a set of 5 matrices for each value m. None of the matrices generated has null off-diagonal elements. In figure A.1 we show the plot of the error variable δ, averaged over the 5 matrices, as a function of the matrix order m. The error variable falls rapidly with increasing matrix size, and becomes insignificant when $m > 500$.

Next we consider the effect of increasing the sparseness of the matrices. To do this we have generated random element matrices of size $m = 200, 500, 1000$. We have controlled the degree of sparseness of these matrices by setting to zero a specified fraction of randomly selected off-diagonal elements. For each fraction of zero elements, we have generated 5 different matrices and computed the average value of δ. Figure A.2 shows the result of this study. Here the different curves are for different values of m.

A.4 Experiments

To evaluate the practical utility of our eigenvector approximation method, we have used it in conjunction with the grouping method described in Chapter 4. We have compared the

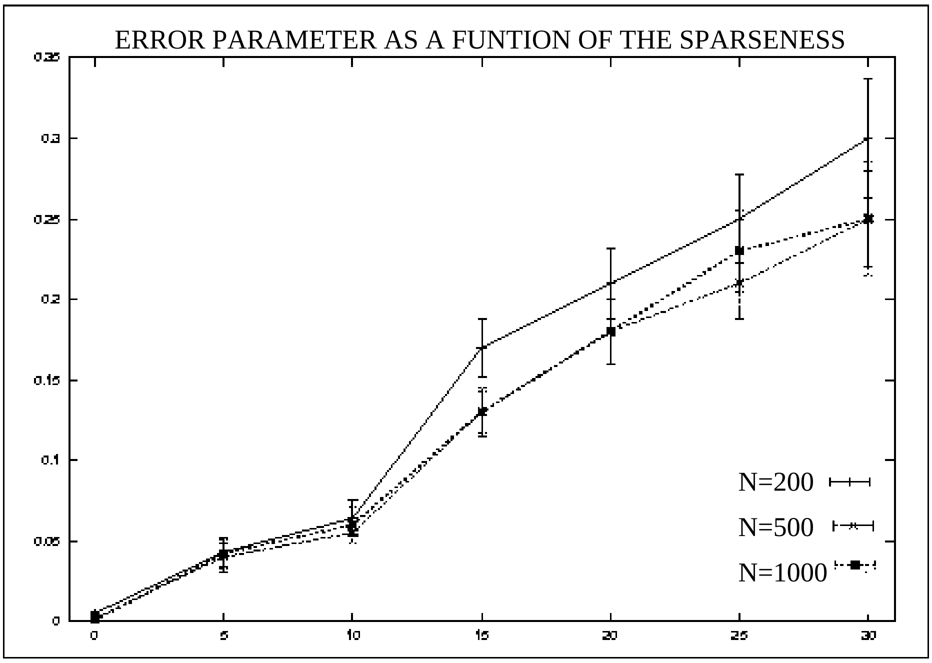


Figure A.2: Effect of the sparseness on δ

results obtained when the leading eigenvector is computed using the approximation described here and the well-known power method. Since the order of the matrix A decreases with each iteration of the grouping process, provisions have been taken when using the approximation method in order to avoid processing matrices with $m < 100$.

We commence by showing results obtained using simulated images. For this study, we use scenes composed of three rectangular regions, as illustrated in Figure A.3. To the synthetic images we have added Gaussian noise of increasing standard deviation. In the figure, from left to right the different columns show the original images, the segmentaion result obtained using the power method and the result obtained using our eigenvector approximation method. The different rows are obtained when the standard deviation of the Gaussian noise is 35% and 50% of the grey-scale difference between the regions. There are no significant differences between the results delivered by the two methods. The result of a more systematic quantitative evaluation is shown in Figure A.4. Here we plot the percentage of mis-assigned pixels as a function of the noise standard deviation quoted as a fraction of the grey-scale difference between regions. Again, there is little to distinguish the performance of the two methods.

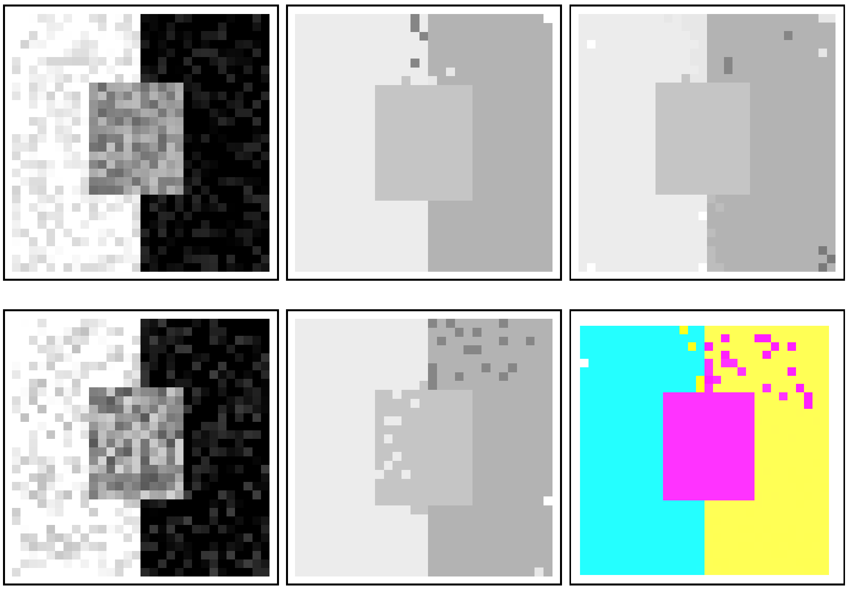


Figure A.3: Results obtained on synthetic images.

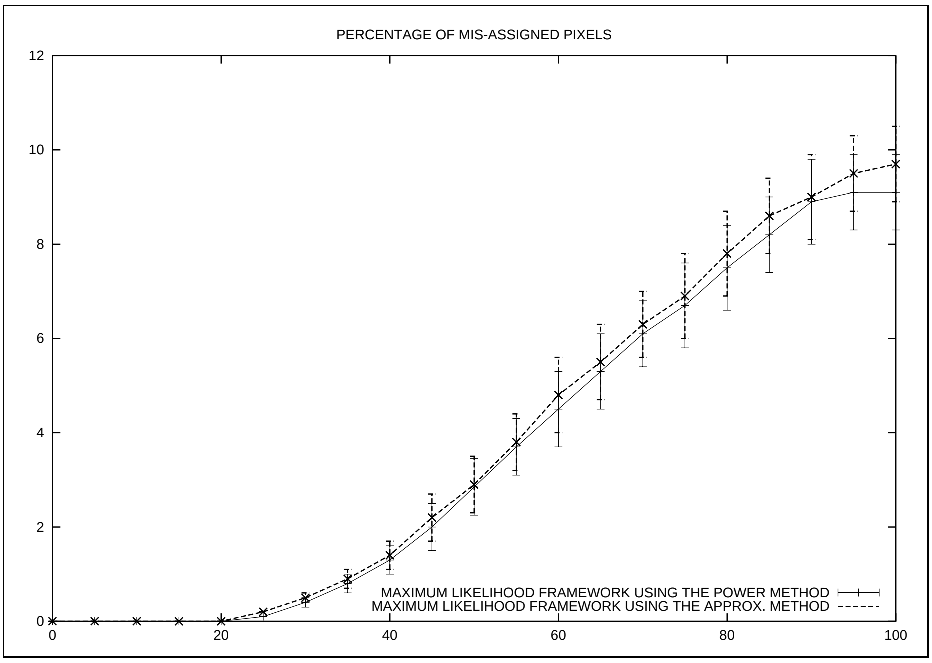


Figure A.4: Percentage of mis-assigned pixels

Finally, we study some real-world imagery. In the left hand column of Figure A.5 we show the original images, the middle column shows the segmentations obtained when the leading eigenvector is computed using the power method and the right hand column shows

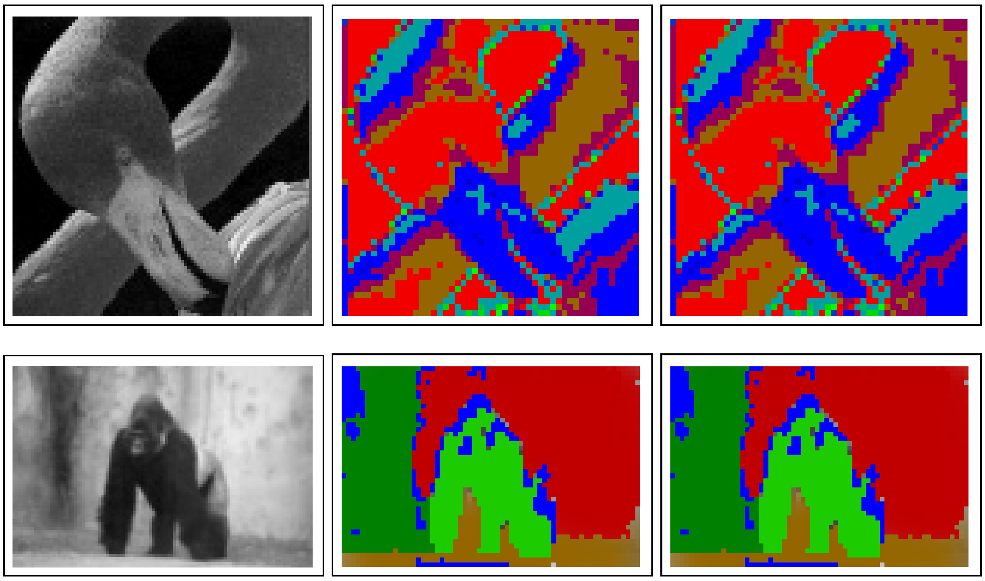


Figure A.5: Segmentations of grey-scale images.

the result obtained when we use our approximation method. There is no apparent segmentation difference between the two methods. However, the new approximation method is on average 2 times faster when applied to the synthetic images and 2.5 times faster when processing the real images. In our experiments the parameter δ never exceeded 0.005.

A.5 Conclusions

We have presented a non-iterative method for approximating the leading eigenvector of a large, dense, non-negative symmetric matrix with the aim of developing more efficient graph-spectral image segmentation algorithms. The method decomposes the original matrix using a linear combination of perturbation matrices whose spectral properties are well known. This results in a significant improvement in computational efficiency. Moreover, in a practical setting, the method reduces the required computation time by a factor of at least 2.

Appendix B

Pairwise Graph Similarity Measures and Multidimensional Scaling

In Chapter 6, we made use of a multidimensional scaling algorithm for displaying data using its projection onto a Euclidean space defined in $\Re^2$. We also used two pairwise graph similarity measures developed by Luo *et al.* (Luo et al., 2002a). In this appendix, we describe how these distance metrics were computed and provide a brief description of the multidimensional scaling algorithm.

B.1 Pairwise Graph Distance Measures

When comparing the graph clustering results in Chapter 6, we made use of two graph distance metrics alternative to our graph edit distance. The first of these similarity measures employs a normalised histogram representation to capture the morphological affinity of the graphs. The aim in doing this is to use the information in the adjacency matrix and, at the same time, avoid computing the covariance matrix when the correspondance information is not available. For a binary graph $G_i = (V_i, E_i)$, the elements of the adjacency matrix $\{A_{j,k} \in [0,1] \mid j \in\mid V_i \mid \wedge k \in\mid V_i \mid\}$ can be classified into m non-overlapping intervals $\delta_l, l = 1, 2, \ldots, m$. Hence, we increment the contents of the histogram bin $\mathcal{H}_i(l)$

corresponding to the interval indexed l using the following association rule

$$\mathcal{H}_i(l) = \begin{cases} \mathcal{H}_i(l) + A_{j,k} & \text{if } A_{j,k} \in \delta_l \\ \\ \mathcal{H}_i(l) & \text{otherwise} \end{cases} \tag{B.1}$$

From the bin-contents, we compute a normalised histogram whose bins are given by

$$\hat{\mathcal{H}}_i(l) = \frac{\mathcal{H}_i(l)}{\sum_{l=1}^{m} \mathcal{H}_i(l)} \tag{B.2}$$

With these normalised bins, we construct a vector of the form $\hat{\mathcal{H}}_i = (\hat{\mathcal{H}}_i(l), \hat{\mathcal{H}}_i(2), \ldots, \hat{\mathcal{H}}_i(m))$. This vector notation suggests a way of computing the pairwise similarities between each pair of graphs. Thus, the pairwise similarity between the graphs indexed i and j is given by the Euclidean distance between their normalised bin-contents. In this manner, the elements of the pairwise similarity matrix $\mathcal{D}$ are

$$\mathcal{D}_{i,j} = \sum_{l=1}^{K} \left[\hat{\mathcal{H}}_i(l) - \mathcal{H}_j(l) \right]^2 \tag{B.3}$$

The second similarity measure makes use of a simple matrix method for capturing the graph structure. In this case, we assume the correspondance information is available in the form of the matrix of correspondance indicators C. For the graphs G_i and G_j, the similarity measure is given by

$$\mathcal{D}_{i,j} = Tr[A_i^T(\mathbf{1} - C)A_j C^T] \tag{B.4}$$

where $\mathbf{1}$ is a matrix whose elements are all unity and A_i is the adjacency matrix corresponding to the graph indexed i.

B.2 Multidimensional Scaling

Multidimensional scaling (MDS), is the term used to describe the family of procedures which allow a set of pairwise distances to be embedded in a small number of dimensions, usually three or four (Chatfield and Colins, 1980). The classical multidimensional scaling method was proposed by Torgenson (Torgerson, 1952) and Gower (Gower, 1966). These are essentially algebraic approaches that assume the pairwise similarities to be Euclidean distances. The main drawback of these approaches is the inappropriateness of the Euclidean metric to reflect the significance of the rank-order of the dissimilarities. To overcome this problem, Shepard and Kruskal (Kruskal, 1964) developed a different scaling technique which makes use of the ordinal properties of the dissimilarities.

The multidimensional scaling method employed in Chapter 6 is based upon that of Torgenson (Torgerson, 1952). We used this method since it is particularly appropriate when the dissimilarities are, approximately or exactly, Euclidean distances. To develop his MDS method, Torgerson made use of the fact that it is straightforward to compute the Euclidean distances between each member of a set of points with known co-ordinates in a high-dimensional Euclidean space. Here, our goal of computation is to solve the inverse problem. That is, we aim to find the co-ordinates that satisfy in an optimum manner a set of known pairwise distances. We solve the problem in two stages. First, we find the matrix $H = JJ^T$, which is the matrix of pairwise sums of squares and scalar products. Second, we factorise it to find J.

With this in mind, we construct an $m \times m$ similarity matrix B, whose elements are defined as

$$B_{i,j} = \begin{cases} \mathcal{D}_{i,j} & \text{if } i \neq j \\ 0 & \text{if } i = j \end{cases} \tag{B.5}$$

Next, we proceed to calculate a matrix H, whose element indexed i, j is given by

$$H_{i,j} = -\frac{1}{2}[B_{i,j}^2 - \hat{B}_{i.}^2 - \hat{B}_{.j}^2 + \hat{B}_{..}^2], \tag{B.6}$$

where

$$\begin{aligned}
\hat{B}_{i.} &= \frac{1}{m}\sum_{j=1}^{m} B_{i,j} \\
\hat{B}_{.j} &= \frac{1}{m}\sum_{i=1}^{m} B_{i,j} \\
\hat{B}_{..} &= \frac{1}{m^2}\sum_{i=1}^{m}\sum_{j=1}^{m} B_{i,j} \qquad \text{(B.7)}
\end{aligned}$$

From the above equations, $\hat{B}_{i.}$ can be interpreted as the average dissimilarity value over the ith row. Similarly, $\hat{B}_{.j}$ is the average dissimilarity value over the jth column and $\hat{b}_{..}$ is the average distance value over all rows and columns of the similarity matrix H.

In order to obtain a matrix of embedding co-ordinates J from matrix H, we perform an eigenvector analysis on H. Since H is real, positive, definite, it has $k \leq m$ non-null eigenvalues. We arrange the k non-zero eigenvalues λ_i of matrix H according to their magnitude order. In other words, the eigenvalues are arranged so that they satisfy the condition $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_k > 0$. We denote the eigenvector corresponding to the eigenvalue indexed i as x_{λ_i}.

The fact that a semidefinite, positive matrix can be factorised into the form JJ^T is based upon the Young-Householder factorisation theorem (Young and Householder, 1938). The Young-Householder theorem states that if J is of the form

$$J = [\vec{f}_1, \vec{f}_2, \ldots, \vec{f}_k], \qquad \text{(B.8)}$$

then, for all $i = 1, 2, \ldots, k$, we have

$$(JJ^T)\mathrm{x}_{\lambda_i} = \sqrt{\lambda_i}\vec{f}_i \qquad \text{(B.9)}$$

Let $\vec{f}_i$ be the ith eigenvector scaled so its sum of squares is equal to λ_i (i.e. $\vec{f}_i = \sqrt{\lambda_i}\mathrm{x}_{\lambda_i}$). Since $H\mathrm{x}_{\lambda_i} = \lambda_i\mathrm{x}_{\lambda_i}$ and $(JJ^T)\mathrm{x}_{\lambda_i} = H\mathrm{x}_{\lambda_i}$, it follows that $H = JJ^T$. As a result, the embedded vector $\vec{p}_i$ in a two-dimensional space for the graph indexed i is given by

$$\vec{p_i} = [\mathrm{x}_{\lambda_1}(i), \mathrm{x}_{\lambda_2}(i)]^T \tag{B.10}$$

where $\mathrm{x}_{\lambda_1}(i), \mathrm{x}_{\lambda_2}(i)$ are the first two elements in the ith row of matrix J.

Appendix C

Pairwise Clustering

In Chapter 6, we illustrated the utility of the graph edit distance for purposes of graph clustering. In this appendix, we provide a brief description of the algorithm used for clustering the graphs. This is a variant of the pairwise clustering method developed in Chapter 3. Recall that the process of pairwise clustering is somewhat different to the more familiar one of central clustering. Whereas central clustering aims to characterise cluster-membership using the cluster mean and variance, in pairwise clustering it is the relational similarity of pairs of objects which are used to establish cluster membership. In Chapter 6, the relational similarity was computed using the edit distance between graphs in the data-set.

To commence, we require some formalism. The objects to be clustered are characterised using a matrix of pairwise similarity weights. The elements of this weight matrix are given by

$$W^{(0)}_{i1,i2} = \begin{cases} \exp[-kr^2_{i1,i2}] & \text{if } i1 \neq i2 \\ 0 & \text{otherwise} \end{cases} \tag{C.1}$$

where k is a constant and $r^2_{i1,i2}$ is the pairwise distance between the pair of graphs indexed $(i1, i2) \in \Phi$. The aim in pairwise clustering is to locate the updated set of similarity weights which partition the graphs into clusters. To be more formal, let V denote the index-set of the graphs and suppose that Ω is the set of pairwise-clusters to which the graphs are assigned. The initial set of clusters are defined by the eigenmodes of the link-weight matrix $W^{(0)}$. Here we follow Sarkar and Boyer (Sarkar and Boyer, 1998). They have shown how the positive eigenvectors of the matrix of link-weights can be used to assign nodes to perceptual clusters. Using the Rayleigh-Ritz theorem, they observe that the scalar quantity $\underline{x}^T W^{(0)} \underline{x}$ is maximised when $\underline{x}$ is the leading eigenvector of $W^{(0)}$. Moreover, each of the subdominant eigenvectors corresponds to a disjoint perceptual cluster. They confine their attention to the same-sign positive eigenvectors (i.e. those whose corresponding eigenvalues are real and positive, and whose components are either all positive or are all negative in sign). If a component of a positive eigenvector is non-zero, then the corresponding node belongs to the perceptual cluster associated with the associated eigenmodes of the weighted adjacency matrix. The eigenvalues $\lambda_1, \lambda_2....$ of $W^{(0)}$ are the solutions of the equation $|W^{(0)} - \lambda I| = 0$ where I is the $|V| \times |V|$ identity matrix. The corresponding eigenvectors $\underline{x}_{\lambda_1}, \underline{x}_{\lambda_2}, ...$ are found by solving the equation $W^{(0)} \underline{x}_{\lambda_i} = \lambda_i \underline{x}_{\lambda_i}$. Let the set of positive same-sign eigenvectors be represented by $\Omega = \{\omega | \lambda_\omega > 0 \wedge [(\underline{x}^*_\omega(i) > 0 \forall i) \vee \underline{x}^*_\omega(i) < 0 \forall i])\}$.

Suppose that $s^{(n)}_{i\omega}$ is the probability that the graph indexed i belongs to the cluster indexed ω at iteration n of the algorithm. We are interested in posing the clustering problem in a maximum likelihood setting. Following Chapters 3 and 4, we assume that pairs of nodes associate to clusters as the outcome of a Bernoulli trial with the cluster-memberships and the link-weights as parameters. For the affinity weight matrix W, the likelihood-function originated from this model is given by

$$P(W) = \prod_{\omega \in \Omega} \prod_{(i,j) \in V \times V} W_{i,j}^{s_{i\omega} s_{j\omega}} (1 - W_{i,j})^{1 - s_{i\omega} s_{j\omega}} \tag{C.2}$$

For the likelihood function above, the corresponding expected log-likelihood function is

$$Q(A^{(n+1)} | A^{(n)}) = \sum_{\omega \in \Omega} \sum_{(i,j) \in V \times V} \zeta^{(n)}_{i,j,\omega} \Big\{ s_{i\omega} s_{j\omega} \ln W_{i,j} + (1 - s_{i\omega} s_{j\omega}) \ln(1 - W_{i,j}) \Big\} \tag{C.3}$$

where $\zeta^{(n)}_{i,j,\omega}$ is the *a posteriori* cluster membership probability (i.e. $P(s_{i\omega} s_{j\omega} | A^{(n)}_{i,j})$).

In Chapters 3 and 4, we have shown how this log-likeihood function can be iteratively optmised using an EM-like process. In the E-step, the cluster membership probabilities and the *a posteriori* probabilities are updated according to the formulas

$$\begin{aligned} s_{i\omega}^{(n+1)} &= \frac{\prod_{j\in V}\left\{\frac{W_{i,j}^{(n)}}{1-W_{i,j}^{(n)}}\right\}^{\zeta_{i,j,\omega}^{(n)} s_{j\omega}^{(n)}}}{\sum_{\omega\in\Omega}\prod_{j\in V}\left\{\frac{W_{i,j}^{(n)}}{1-W_{i,j}^{(n)}}\right\}^{\zeta_{i,j,\omega}^{(n)} s_{j\omega}^{(n)}}} \\ \zeta_{i,j,\omega}^{(n+1)} &= \frac{A_{i,j}^{(n)\,s_{i\omega}^{(n)} s_{j\omega}^{(n)}}(1-A_{i,j}^{(n)})^{1-s_{i\omega}^{(n)} s_{j\omega}^{(n)}}}{\sum_{(i,j)\in\Phi} A_{i,j}^{(n)\,s_{i\omega}^{(n)} s_{j\omega}^{(n)}}(1-A_{i,j}^{(n)})^{1-s_{i\omega}^{(n)} s_{j\omega}^{(n)}}} \end{aligned} \tag{C.4}$$

Once the revised cluster membership variables and the updated *a posteriori* probabilities are to hand, we apply the M-step of the algorithm to update the similarity-weight matrix. The updated similarity-weights are given by

$$W_{i,j}^{(n+1)} = \frac{\zeta_{i,j,\omega}^{(n)}\sum_{\omega\in\Omega} s_{i\omega}^{(n)} s_{j\omega}^{(n)}}{\sum_{\omega\in\Omega}\zeta_{i,j,\omega}^{(n)}} \tag{C.5}$$

These two steps are interleaved and iterated to convergence.

Appendix D

List of Publications

The following is a list of publications produced during the course of my PhD. Each of the following sections corresponds to a chapter comprising material contained in the referenced papers.

Chapter 3

- A. Robles-Kelly, E. R. Hancock, An Expectation-Maximisation Framework for Segmentation and Grouping. In *Image and Vision Computing* Vol.20, pages 725-738. 2002

- E. Ribeiro, A. Robles-Kelly, E. R. Hancock, Detecting multiple Texture Planes Using Local Spectral Distortion. In *Image and Vision Computing* Vol.20, pages 739-750. 2002

- E. Ribeiro, A. Robles-Kelly, E. R. Hancock, An Expectation-Maximization Framework for Perceptual Grouping. In *Proceedings of the International Workshop of Visual Form 4*, Lecture Notes on Computer Science Vol. 2059, pages 594-605. 2001.

- A. Robles-Kelly, E. R. Hancock, An EM-like Algorithm for Motion Segmentation via Eigendecomposition. In *Proceedings of the British Machine Vision Conference*, pages 123-132. 2001.

Chapter 4

- A. Robles-Kelly, E. R. Hancock, Pairwise Clustering with Matrix Factorisation and the EM Algorithm. In *Proceedings of European Conference on Computer Vision*, Lecture Notes in Computer Science Vol. 2351, pages 63-67. 2002.

- A. Robles-Kelly, E. R. Hancock, A Maximum Likelihood Framework for Iterative Eigendecomposition. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 654-661. 2001.

- A. Robles-Kelly, A. G. Bors, E. R. Hancock, Hierarchical Iterative Eigendecomposition for Motion Segmentation. In *Proceedings of the IEEE International Conference on Image Processing*, pages 363-366. 2001.

- A. Robles-Kelly, E. R. Hancock, A Maximum Likelihood Framework for Grouping and Segmentation. In *Energy Minimization Methods in Computer Vision and Pattern Recognition*, Lecture Notes on Computer Science Vol. 2134, pages 251-266. 2001.

- A. Robles-Kelly, E. R. Hancock, Maximum Likelihood Motion Segmentation using Eigendecomposition. In *Proceedings of the 11th International Conference on Image Analysis and Processing*, pages 63-68. 2001.

- A. Robles-Kelly, E. R. Hancock, Grouping Line-Segments Using Eigenclustering. In *Proceedings of the British Machine Vision Conference*, pages 123-132. 2000.

Chapter 5

- A. Robles-Kelly and E. R. Hancock, A Graph-spectral Method for Surface Height Recovery. In *Proceedings of Mathematics of Surfaces X, To appear*. 2003.

- A. Robles-Kelly, A. Bors and E. R. Hancock, Surface Acquisition from Single Gray-scale Images. In *Proceedings of the IEEE International Conference on Image Processing, To appear*. 2003.

- A. Robles-Kelly, E. R. Hancock, A Graph Spectral Approach to Shape-from-Shading. In *Proceedings of the IEEE International Conference on Image Processing*, pages II:569-572. 2002.

- A. Robles-Kelly, E. R. Hancock, A Mumford-Shah Diffusion Process for Shape-from-Shading. In *Proceedings of the British Machine Vision Conference*, pages 708-717. 2002.

- A. Robles-Kelly, E. R. Hancock, A Graph-Spectral Approach to Surface Segmentation. In *Proceedings of the International Conference on Pattern Recognition*, pages III:509-512. 2002.

- A. Robles-Kelly, E. R. Hancock, A Graph-Spectral Method for Surface Height Recovery from Needle-maps. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages I:141-148. 2001.

Chapter 6

- A. Robles-Kelly, E. R. Hancock, String Edit Distance, Random Walks and Graph Matching. In *International Journal of Pattern Recognition and Artificial Intelligence, To appear*. 2004.

- A. Robles-Kelly, E. R. Hancock, Edit Distance From Graph Spectra. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 234-241. 2003.

- A. Robles-Kelly, E. R. Hancock, Graph Matching Using Spectral Seriation. In *Energy Minimisation Methods in Computer Vision and Pattern Recognition*, Lecture Notes in Computer Science Vol. 2683, pages 517-532. 2003.

- A. Robles-Kelly, E. R. Hancock, A Graph-spectral Approach to Correspondence Matching. In *Proceedings of the International Conference on Pattern Recognition*, pages IV:176-179. 2002.

- B. Luo, A. Robles-Kelly, A. Torsello, R. C. Wilson, E. R. Hancock, A Probabilistic Framework for Graph Clustering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages I:912-919. 2001.

- B. Luo, A. Robles-Kelly, A. Torsello, R. C. Wilson, E. R. Hancock, Learning Shape Categories by Clustering Shock Trees. In *Proceedings of the IEEE International Conference on Image Processing*, pages 672-675. 2001.

- A. Robles-Kelly, E. R. Hancock, Graph Matching using Adjacency Matrix Markov Chains. In *Proceedings of the British Machine Vision Conference*, pages 384-390. 2001.

- B. Luo, A. Robles-Kelly, A. Torsello, R. C. Wilson, E. R. Hancock, Discovering Shape Categories by Clustering Shock Trees. In *Proceedings of the International Conference on Computer Analysis of Images and Patterns*, Lecture Notes on Computer Science Vol. 2124, pages 152-160. 2001.

Appendix A

- A. Robles-Kelly, S. Sarkar, E. R. Hancock, A Fast Leading Eigenvector Approximation for Segmentation and Grouping. In *Proceedings of the International Conference on Pattern Recognition*, pages I:639-642. 2002.

Bibliography

Akaike, H. (1969). Fitting autoregressive models for prediction. *Annals of the Insttitute of Statistical Mathematics*, (21):243–247.

Akaike, H. (1978). A bayesian analysis of the minimum AIC proceduce. *Annals of the Institute of Mathematical Statistics*, (30):7–14.

Amir, A. and Lindenbaum, M. (1998). A generic grouping algorithm and its quantitative analysis. *Trans. on Pattern Analysis and Machine Intelligence*, 20(2):168–185.

Atkins, J. E., Bowman, E. G., and Hendrickson, B. (1998). A spectral algorithm for seriation and the consecutive ones problem. *SIAM Journal on Computing*, 28(1):297–310.

August, J., Siddiqi, K., and Zucker, S. (1999). Contour fragment grouping and shared, simple occluders. *Computer Vision and Image Understanding*, 76(2):146–162.

Azar, Y., Fiat, A., Karlin, A. R., McSherry, F., and Saia, J. (2000). Spectral analysis of data. In *ACM Symposium on Theory of Computing*, pages 619–626.

Barrow, H. G. and Poppplestone, R. J. (1971). Relational descriptions in picture processing. *Machine Intelligence*, 6:377–396.

Beck, J., Rosenfeld, A., and Ivry, R. (1989). Line segmentation. *Spatial Vision*, 4:75–101.

Berman, A. and Plemmons, R. J. (1994). *Nonnegative Matrices in the Mathematical Sciences*, volume 9 of *Classics in Applied Mathematics*. SIAM.

Bichsel, M. and Pentland, A. P. (1992). A simple algorithm for shape from shading. In *Proc. of the IEEE Conf. on Comp. Vision and Pattern Recognition*, pages 459–465.

Biggs, N. L. (1993). *Algebraic Graph Theory*. Cambridge University Press.

Borra, S. and Sarkar, S. (1997). A framework for performance characterization of intermediate-level grouping modules. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, (19):1306–1312.

Bors, A. G., Hancock, E. R., and Wilson, R. C. (2003). Terrain analysis using radar shape-from-shading. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(8):974–992.

Boyer, K. L. and Sarkar, S. (1999). Guest editors' introduction: Perceptual organization in computer vision: Status, challenges, and potential. 79(1):1–5.

Bremaud, P. (2001). *Markov Chains, Gibbs Fields, Monte Carlo Simulation and Queues*. Springer.

Bridle, J. S. (1990). Training stochastic model recognition algorithms can lead to maximum mutual information estimation of parameters. In *NIPS 2*, pages 211–217.

Brooks, M. J. and Horn, B. K. P. (1985). Shape and source from shading. *IJCAI*, pages 932–936.

Bülthoff, H. H. and Mallot, H. A. (1988). Integration of depth modules, integration and shading. *Journal of the Optical Society of America*, A5:1749–1758.

Bunke, H. (1997). On a relation between graph edit distance and maximum common subgraph. *Pattern Recognition Letters*, 18(8):689–694.

Canny, J. (1986). A computational approach to edge detection. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 8(6):679–698.

Castaño, R. and Hutchinson, S. (1996). A probabilistic approach to perceptual grouping. *Computer Vision and Image Understanding*, 64(3):339–419.

Chatfield, C. and Colins, A. (1980). *Introduction to multivariate analysis*. Chapman & Hall/CRC.

Christmas, W. J., Kittler, J., and Petrou, M. (1995). Structural matching in computer vision using probabilistic relaxation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(8):749–764.

Chung, F. R. K. (1997). *Spectral Graph Theory*. American Mathematical Society.

Cohen, L. D. (2001). Multiple contour finding and perceptual grouping using minimal paths. *Journal of Mathematical Imaging and Vision*, 14(3):225–236.

Cohen, L. D. and Kimmel, R. (1997). Global minimum for active contour models. *International Journal on Computer Vision*, 24(1):57–78.

Cox, I. J., Rehg, J. M., and Hingorani, S. (1993). A bayesian multiple-hypothesis approach to edge grouping and contour segmentation. *International Journal of Computing and Vision*, 11(1):5–24.

Crevier, D. (1999). A probabilistic method for extracting chains of collinear segments. *Image and Vision Computing*, 76(1):36–53.

Cross, A. D. J. and Hancock, E. R. (1998). Graph matching with a dual step em algorithm. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 20(11):1236–1253.

Cross, A. D. J., Wilson, R. C., and Hancock, E. R. (1997). Inexact graph matching using genetic search. *Pattern Recognition*, 30(6):953–970.

Cvetković, D., Doob, M., and Sachs, H. (1980). *Spectra of Graphs:Theory and Application*. Academic Press.

Del Corso, G. M. (2000). Randomized error estimation for eigenvalue approximation. *Calcolo*, 37:21–46.

Dempster, A., Laird, N., and Rubin, D. (1977). Maximum-likehood from incomplete data via the EM algorithm. *J. Royal Statistical Soc. Ser. B (methodological)*, 39:1–38.

Dickson, W. (1991). Feature grouping in a hierarchical probabilistic network. *Image and Vision Computing*, 9(1):51–57.

Dieci, L. (1996). Considerations on computing real logarithms of matrices,hamiltonian logarithms and skew-symmetric logarithms. *Linear Algebra and its Applications*, 244:35–54.

Dijkstra, E. W. (1959). A note on two problems in connection with graphs. *Numerische Math*, 1:269–271.

Do Carmo, M. P. (1976). *Differential Geometry of Curves and Surfaces*. Prentice Hall.

Dupuis, P. and Oliensis, J. (1992). Direct method for reconstructing shape from shading. In *Proc. of the IEEE Conf. on Comp. Vision and Pattern Recognition*, pages 453–458.

Eshera, M. A. and Fu, K. S. (1984a). A graph distance measure for image analysis. *IEEE Transactions on Systems, Man and Cybernetics*, 14:398–407.

Eshera, M. A. and Fu, K. S. (1984b). A graph distance measure for image analysis. *SMC*, 14(3):398–408.

Faloutsos, M., Faloutsos, P., and Faloutsos, C. (1999). On power-law relationships of the internet topology. In *SIGCOMM*, pages 251–262.

Ferrie, F. P. and Legarde, J. (1990). Curvature consistency improves local shading analysis. In *Proc. of the Int. Conf. on Pattern Recognition*, volume 1, pages 70–76.

Fiedler, M. (1975a). Algebraic connectiviity of graphs. *Czech Math. Journal*, (23):298–305.

Fiedler, M. (1975b). A property of eigenvectors of nonnegative symmetric matrices and its application to graph theory. *Czech Math. Journal*, (25):619–633.

Field, D. J. (1993). Contour integration by the human visual system: Evidence for a local association field. *Vision Res.*, 33(2):173–193.

Figueiredo, M., Leitao, J. M., and Jain, A. K. (1999). On fitting mixture models. In *Proceedings of the Second Int. Workshop on Energy Minimization Methods in Comp. Vis. and Pattern Recognition*, number 1654 in Lecture Notes in Comp. Science, pages 54–67.

Finch, A. M., Wilson, R. C., and Hancock, E. R. (1998). An energy function and continuous edit process for graph matching. *Neural Computation*, 10(7):1873–1894.

Fischler, M. A. and Eischlager, R. A. (1977). The representation and matching of pictorial structures. In *CMetImAly77*, pages 31–56.

Forsyth, D. and Zisserman, A. (1989). Mutual illumination. In *Proc. of the IEEE Conf. on Comp. Vision and Pat. Recognition*, pages 466–473.

Fowlkes, C., Belongie, S., and Malik, J. (2001). Efficient spatiotemporal grouping using the Nyström method. In *Proc. of the IEEE Int. Conf. on Computer Vision and Pattern Recognition*, pages I:231–238.

Frankot, R. T. and Chellappa, R. (1988). A method of enforcing integrability in shape from shading algorithms. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 4(10):439–451.

Frankot, R. T. and Chellappa, R. (1990). Estimation of surface topography from sar imagery using shape from shading techniques. *AI*, 43(3):271–310.

Gantmacher, F. R. (1971). *Matrix Theory*, volume II. Chelsea Publishing Co.

Gdalyahu, Y., Weinshall, D., and Werman, M. (1999). A randomized algorithm for pairwise clustering. In *Advances in Neural Information Processing Systems 11*, pages 424–430. MIT Press.

Gkantsidis, C., Mihail, M., and Zegura, E. (2003). Spectral analysis of internet topologies. To appear in Infocom.

Gold, S. and Rangarajan, A. (1996). A graduated assignment algorithm for graph matching. *PAMI*, 18(4):377–388.

Goldstein, E. B. (1989). *Sensation and Perception*. Wadsworth Publishing Company.

Gower, J. C. (1966). Some distance properties of latent root and vector methods used in multivariate analysis. *Biometrika*, 53:325–328.

Guy, G. and Medioni, G. (1996). Inferring global perceptual contours from local features. *International Journal of Computer Vision*, 20(1/2):113–133.

Heitger, F. and von der Heydt, R. (1993). A computational model of neural contour processing. In *IEEE Conf. on Comp. Vis. and Patt. Recog.*, pages 32–40.

Hofmann, T. and Buhmann, M. (1997). Pairwise data clustering by deterministic annealing. *IEEE Tansactions on Pattern Analysis and Machine Intelligence*, 19(1):1–14.

Horaud, R. and Skordas, T. (1989). Stereo correspondence through feature grouping and maximal cliques. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11(11):1168–1180.

Horaud, R. and Sossa, H. (1995). Polyhedral object recognition by indexing. *Pattern Recognition*, 28(12):1855–1870.

Horn, B. K. P. (1975). *The Psycology of Computer Vision*, chapter *Obtaining Shape from Shading Information*, pages 115–155. McGraw Hill.

Horn, B. K. P. and Brooks, M. J. (1986). The variational approach to shape from shading. *CVGIP*, 33(2):174–208.

Horn, R. A. and Johnson, C. R. (1991). *Topics in Matrix Analysis*. Cambridge University Press.

Hsieh, C. H., Lu, P. C., Shyn, J. S., and Lu, E. H. (1990). Motion estimation algorithm using inter-block correlation. *IEE Electron. Lett.*, 26(5):276–277.

Huet, B. and Hancock, E. R. (2002). Relational object recognition from large structural libraries. *Pattern Recognition*, 35(9):1895–1915.

Ikeuchi, K. (1984). Reconstructing a depth map from intensity maps. In *Int. Conf. on Pattern Recognition*, pages 736–738.

Jordan, M. I. (1998). *Learning in Graphical Models*. MIT Press.

Kalocsai, P. (1997). Improving the shape recognition performance of a model with gabor filter representation. In *Computer Analysis of Images and Patterns*, pages 369–375.

Kannan, R., Vampala, S., and Vetta, A. (2000). On clusterings: Good, bad and spectral. In *Proceedings of the 41st Symposium on the Foundations of Computer Science*, pages 367–77.

Kimmel, R. and Bruckstein, A. M. (1995). Tracking level sets by level sets: a method for solving the shape from shading problem. *Computer vision and Image Understanding*, 62(2):47–48.

Kimmel, R., Siddiqqi, K., Kimia, B. B., and Bruckstein, A. M. (1995). Shape from shading: Level set propagation and viscosity solutions. *International Journal of Computer Vision*, (16):107–133.

Koffka, K. (1922). Perception: An introduction to the gestalt-theorie. *Psychological Bulletin*, 19:531–585.

Kruskal, J. B. (1964). Nonmetric multidimensional scaling: A numerical method. *Psychometrika*, 29:115–129.

Leclerc, Y. G. and Bobick, A. F. (1991). The direct computation of height from shading. In *Proceedings of Computer Vision and Pattern Recognition*, pages 552–558.

Leite, J. A. F. and Hancock, E. R. (1997). Iterative curve organisation with the em algorithm. *Pattern Recognition Letters*, 18:143–155.

Levenshtein, V. I. (1966). Binary codes capable of correcting deletions, insertions and reversals. *Sov. Phys. Dokl.*, 6:707–710.

Levi, G. (1972). A note on the derivation of maximal common subgraphs of two directed or undirected graphs. *Calcols*, 9:341–354.

Levinson, R. (1992). Pattern associativity and the retrieval of semantic networks. *Comput. Math. Appl.*, 23:573–600.

Lovász, L. (1993). Random walks on graphs: a survey. *Bolyai Society Mathematical Studies*, 2(2):1–46.

Lüdtke, N. (2002). *A Population Coding Approach to Edge Detection and Perceptual Grouping*. PhD thesis, The University of York.

Luo, B. and Hancock, E. R. (1999). Procrustes alignment with the EM Algorithm. In *8th International Conference on Computer Analysis of Images and Image Patterns*, pages 623–631.

Luo, B. and Hancock, E. R. (2001). Structural graph matching using the EM algorithm and singular value decomposition. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 23(10):1120–1136.

Luo, B., Wilson, R., and Hancock, E. (2002a). Eigenspaces for graphs. *International Journal of Image and Graphics*, 2(2):247–268.

Luo, B., Wilson, R. C., and Hancock, E. R. (2002b). Spectral feature vectors for graph clustering. In *S+SSPR 2002*, pages 82–90.

McGregor, J. J. (1982). Backtrack search algorithms and the maximal common subgraph problem. *Software Practice and Experience*, 12:23–34.

Meilă, M. and Jordan, M. (1997). Triangulation by continuous embedding. In *NIPS 9*.

Messmer, B. T. and Bunke, H. (1999). A decision tree approach to graph and subgraph isomorphism detection. *Pattern Recognition*, 32(12):1979–1998.

Mingolla, E. and Todd, J. (1986). Perception of solid shape from shading. *Biological Cybernetics*, (53):137–151.

Mohar, B. (2000). *Graph Symmetry: Algebraic Methods and Applications*, chapter *Some Applications of Laplace Eigenvalues of Graphs*, pages 225–275. Kluwer.

Morgan, H. L. (1970). Spelling correction in systems programs. *Communications of the ACM*, 13(2):90–94.

Myers, R., Wilson, R. C., and Hancock, E. R. (2000). Bayesian graph edit distance. *PAMI*, 22(6):628–635.

Nyström, E. J. (1928). Über die Praktische Auflösung von Linearen Integralgleichungen mit Anwndungen auf Randwertaufgaben der Potentialtheorie. *Commentationes Physico-Mathematica*, 4(15):1–52.

Oommen, B. J. and Zhang, K. (1996). The normalized string editing problem revisited. *PAMI*, 18(6):669–672.

Parent, P. and Zucker, S. (1989). Trace inference, curvature consistency and curve detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11(8):823–839.

Peng-Yeng, Y. (1998). Algorithms for straight line fitting using k-means. *Pattern Recognition Letters*, 19:31–41.

Perona, P. and Freeman, W. T. (1998). Factorization approach to grouping. In *Proc. ECCV*, pages 655–670.

Poggio, T., Torre, V., and Koch, C. (1985). Computational vision and regularization theory. *Nature*, 317(6035):314–319.

Press, W. H., Teukolsky, S. A., Vetterlin, W. T., and Flannery, B. P. (1992). *Numerical Recipes in C*. Cambridge University Press.

Ragheb, H. and Hancock, E. R. (2001). Separating lambertian and specular reflectance components using iterated conditional modes. In *Proc. of the British Machine Vision Conference*, pages 541–552.

Ramachandran, V. S. (1988). Perception of shape from shading. *Nature*, (331):163–166.

Rice, S. V., Bunke, H., and Nartker, T. A. (1997). Classes of cost functions for string edit distance. *Algorithmica*, 18:271–280.

Risanen, J. (1989). *Stochastic Complexity in Statistical Enquiry*. World Scientific.

Robles Kelly, A. and Hancock, E. R. (2000). Grouping-line segments using eigenclustering. In *Proceedings of the British Machine Vision Conference*, pages 586–595.

Sakamoto, Y., Ishiguro, M., and Kitagawa, G. (1988). *Akaike Information Criterion Statistics*. KTK Scientific Publishers.

Sanfeliu, A. and Fu, K. S. (1983). A distance measure between attributed relational graphs for pattern recognition. *IEEE Transactions on Systems, Man and Cybernetics*, 13:353–362.

Sarkar, S. and Boyer, K. (1993). Integration, inference and management of spatial information using bayesian networks: Perceptual organization. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 15(3):256–274.

Sarkar, S. and Boyer, K. L. (1998). Quantitative measures of change based on feature organization: Eigenvalues and eigenvectors. *Computer Vision and Image Understanding*, 71(1):110–136.

Scott, G. and Longuet-Higgins, H. (1991). An algorithm for associating the features of two images. In *Proceedings of the Royal Society of London*, number 244 in B, pages 21–26.

Scott, G. L. and Longuet-Higgins, H. C. (1990). Feature grouping by relocalisation of eigenvectors of the proximity matrix. In *British Machine Vision Conference*, pages 103–108.

Sebastian, T. B. and Kimia, B. B. (2001). Curves vs skeletons in object recognition. In *IEEE Int. Conf. on Image Processing*, pages 22–25.

Sebastian, T. B., Klein, P. N., and Kimia, B. B. (2002). Shock-based indexing into large shape databases. In *European Conference on Conputer Vision*, volume 3, pages 731–746.

Sengupta, K. and Boyer, K. L. (1995a). Organizing large structural modelbases. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(4):321–332.

Sengupta, K. and Boyer, K. L. (1995b). Using geometric hashing with information theoretic clustering for fast recognition from a large cad modelbase. In *IEEE International Symposium on Computer Vision*, pages 151–156.

Sengupta, K. and Boyer, K. L. (1998). Modelbase partitioning using property matris spectra. *Computer Vision and Image Understanding*, 70(2):177–196.

Shapiro, L. G. and Haralick, R. M. (1982). Relational models for scene analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 4:595–602.

Shapiro, L. G. and Haralick, R. M. (1985). A metric for comparing relational descriptions. *IEEE Trans on Patterna Analysis and Machine Intelligence*, pages 90–94.

Shapiro, L. S. and Brady, J. M. (1991). A modal approach to feature-based correspondence. In *British Machine Vision Conference*, pages 78–85.

Shashua, A. and Ullman, S. (1988). Structural saliency: The detection of globally salient structures using a locally connected network. In *Proc. 2nd Int. Conf. in Comp. Vision*, pages 321–327.

Shearer, K., Venkatesh, S., and Bunke, H. (1998). An efficient least common subgraph algorithm for video indexing. *Proc. of the Int. Conf. on Pattern Recognition*, pages 1241–1243.

Shi, J. and Malik, J. (1997). Normalized cuts and image segmentations. In *Proc. of the IEEE Conf. on Comp. Vision and Pattern Recognition*, pages 731–737.

Shokoufandeh, A., Dickinson, S. J., Siddiqi, K., and Zucker, S. W. (1998). Indexing using a spectral encoding of topological structure. In *Proceedings of the Computer Vision and Pattern Recognition*, pages 491–497.

Soundararajan, P. and Sarkar, S. (2001). Investigation of measures for grouping by graph partitioning. In *IEEE Conf. on Comp. Vis. and Patt. Recogn.*, volume 1, pages 239–246.

Stewart, G. W. and guang Sun, J. (1990). *Matrix Perturbation Theory*. Academic Press.

Tangmunarunkit, H., Govindan, R., Jamin, S., Shenker, S., and Willinger, W. (2002). Network topology generators: Degree-based vs structural. In *SIGCOMM*.

Todd, J. T. and Mingolla, E. (1983). Perception of surface curvature and direction of illumnia-tion from patterns of shading. *Journal of Experimental Psyhology: Human Perception and Performance*, (9):583–595.

Todd, J. T. and Reichel, F. D. (1989). Ordinal structure in the visual perception and cognition of smoothly curved surfaces. *Psychological Review*, 96(4):643–657.

Torgerson, W. S. (1952). Multidimensional scaling i: Theory and method. *Psychometrika*, 17:401–419.

Ueda, N., Nakano, R., Ghahramani, Z., and Hinton, G. E. (2000). Smem algorithm for mixture models. *Neural Computation*, 12(9):2109–2128.

Ullman, S. (1976). Filling in the gaps: the shape of subjective contours and a model for their generation. *Biological Cybernetics*, 25:1–6.

Umeyama, S. (1988). An eigen decomposition approach to weighted graph matching problems. *PAMI*, 10(5):695–703.

Varga, R. (1962). *Matrix Iterative Analysis*. Prentice Hall.

Varga, R. S. (2000). *Matrix Iterative Analysis*. Springer, second edition.

Vlassis, N. and Likas, A. (1999). A kurtosis-based dynamic approach to gaussian mixture modeling. *IEEE Trans. in Systems, Man and Cybernetics*, 29(4):393 – 399.

Wagner, R. A. and Fisher, M. J. (1974). The string-to-string correction problem. *Journal of the ACM*, 21(1):168–173.

Wang, J. T. L., Shapiro, B. A., Shasha, D., Zhang, K., and Currey, K. M. (1998). An algorithm for finding the largest approximately common substructures of two trees. *PAMI*, 20(8):889–895.

Weiss, Y. (1999). Segmentation using eigenvectors: A unifying view. In *IEEE International Conference on Computer Vision*, pages 975–982.

Wilkinson, J. H. (1965). *The Algebraic Eigenvalue Problem*. Oxford University Press.

Williams, L. R. and Jacobs, D. W. (1997a). Local parallel computation of stochastic completion fields. *Neural Computation*, 9(4):859–882.

Williams, L. R. and Jacobs, D. W. (1997b). Stochastic completion fields: A neural model of illusory contour shape and salience. *Neural Computation*, 9(4):837–858.

Wilson, R. and Hancock, E. R. (1997). Structural matching by discrete relaxation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(6):634–648.

Wong, A. K. C. and You, M. (1985). Entropy and distance of random graphs with application to structural pattern recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 7:599–609.

Worthington, P. L. and Hancock, E. R. (1999). New constraints on data-closeness and needle map consistency for shape-from-shading. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21(12):1250–1267.

Young, G. and Householder, A. S. (1938). Discussion of a set of points in terms of their mutual distances. *Psychometrika*, 3:19–22.

Yuille, A. and Kosowsky, J. (1994). Statistical physics algorithms that converge. *Neural Computation*, (6):341–356.

Yuille, A., Stolorz, P., and Utans, J. (1994). Statistical physics, mixtures of distributions and the em algorithm. *Neural Computation*, (6):334–340.

Zhang, R., Tsai, P. S., Cryer, J. E., and Shah, M. (1999). Shape from shading: A survery. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 21(8):690–706.

Zucker, S. W., David, C., Dobbins, A., and Iverson, L. (1988). The organization of curve detection: Coarse tangent fields and fine spline coverings. In *Proc. of the IEEE Int. Conf. on Comp. Vision*, pages 568–577.