

Non-Rigid Motion Behaviour Learning:

A Spectral and Graphical Approach

Hongfang Wang

Submitted for the degree of Doctor of Philosophy

Department of Computer Science

THE UNIVERSITY *of York*

May 2007

Abstract

In this thesis graph spectral methods and kernel methods are combined together for the tasks of rigid and non-rigid feature correspondence matching and consistent labelling. The thesis is divided into five chapters.

In Chapter 1 we give a brief introduction and an outline of the thesis. In Chapter 2 we review the main techniques in the literature which are related to the developments in this thesis. Topics covered include data representation, the data classification methods, spectral graph matching, and the kernel methods.

Chapter 3 aims at developing a new feature correspondence matching algorithm for rigid and articulated motion. We focus on the point features extracted from consecutive image frames. Specifically, a graph structure is used to represent the data-sets, and spectral graph theory is used for the correspondence localization. The novelty is that a kernel viewpoint is adopted in constructing the proximity matrix, and a consistent labelling process is incorporated into the matching process when the objects under investigation undergoes articulated motion. The algorithm is successfully applied to synthetic data-sets and a group of feature point-sets extracted from real world image sequences.

Chapter 4 develops a new probabilistic relaxation labelling method, aimed at a broader range of applications for feature correspondence matching as well as data clustering. Here again the kernel methods are incorporated into the process, and the evidence combination and propagation steps are governed by a diffusion process defined on a support graph. The support graph is defined on the set of object-label assignments. The problem of consistent labelling thus becomes that of finding a state vector which gives the desired label probabilities. The newly developed algorithm is first applied to a toy labelling example which is taken from two classical relaxation labelling papers. Then the algorithm is applied to the problems of data classification and feature correspondence matching. In the

feature correspondence matching application, the current label probabilities are regarded as noisy vector from the original correct one. Thus the process is viewed as running the diffusion process backwards in time. Experimental results on synthetic and real world data show encouraging results in both of the two cases.

Finally, Chapter 5 concludes the thesis with a summary of the strengths and weaknesses of the thesis, and finally gives suggestions for future work.

Contents

1	Introduction	1
1.1	The problem	1
1.2	The objective	3
1.3	Thesis outline	4
2	Literature review	5
2.1	Data representation	6
2.2	Consistent labelling	10
2.2.1	Data clustering	11
2.2.2	Spectral graph methods	13
2.2.3	Relaxation Labelling	15
2.3	Feature correspondence matching	17
2.3.1	The problem of feature correspondence matching	17
2.3.2	Graph spectral methods for correspondence matching	19
2.3.3	Correspondence matching by relaxation labelling	21
2.3.4	Other correspondence matching methods	22
2.4	Kernel methods	24
2.4.1	The kernel methods	24
2.4.2	Kernels on graphs	26
2.4.3	Applications	27
2.5	Conclusions	29

3	Kernel spectral feature correspondence matching	31
3.1	The problem	32
3.2	Graph spectral matching	34
3.3	Kernel spectral methods	35
3.4	Label Process	38
3.5	Matching	41
3.5.1	Rigid Case	42
3.5.2	Articulated Case	44
3.6	Experiments	47
3.6.1	The data	47
3.6.2	Results	50
3.7	Conclusions	55
4	Probabilistic Relaxation Labelling by Diffusion	61
4.1	The problem	62
4.2	Relaxation Labelling	63
4.3	Diffusion Processes on Graphs	65
4.4	Relaxation labelling by diffusion	69
4.5	Experiments	72
4.5.1	Scene labelling — A toy example	74
4.5.2	Data classification	76
4.5.3	Feature correspondence matching	87
4.5.4	Computational complexity	90
4.6	Discussion	91
5	Conclusions	96
5.1	Correspondence matching	96
5.2	Consistent labelling	97
5.3	Future work	99

List of Figures

3.1	Synthetic data, data-set 6 in Table 3.4. <i>Left: the original generated data; Right: after rigid transformation.</i>	48
3.2	The hand image data (From left to right, top to bottom: frames 08, 09, 11, 25).	49
3.3	The CMU house data (From left to right, top to bottom: frames 01,02,03,04,10).	49
3.4	Effects of different σ value (From left to right, top to bottom: data-set 3, hand 08/09, hand 08/11, hand 08/25).	51
3.5	Synthetic data with 2 and 3 rigid components, and their labeling results. <i>Top: data-set pair 2 in Table 3.4; Bottom: data-set pair 1 in Table 3.4.</i>	52
3.6	Matching results with synthetic data.	56
3.7	Effects of Gaussian random position jitter with a single label.	56
3.8	Matching results, hand 08 and 09, occlusions.	57
3.9	Matching results, hand 08 and 11, occlusions.	57
3.10	Effects of random point deletions, multi-label.	58
3.11	The effect of occlusions with multi-labels.	58
3.12	Effects of random point jitter, multi-label.	59
3.13	Articulated matching results. <i>Top: data-set 3 in Table 3.4; Bottom: data-set 5 in Table 3.4.</i>	59
4.1	An example of a set of networked objects to be labeled.	64
4.2	The toy labeling problem: Label each of its edges to interpret the triangle. (This is taken from Rosenfeld, et. al. [106].)	75

4.3	The four possible labels '+, -, \leftarrow , \rightarrow ' and their state transition graph of the toy labeling example	76
4.4	Labeling results for synthetic and real world data-sets. <i>Left column: The original data-sets; Right column: The clustering results.</i>	79
4.5	Labeling results for synthetic and real world data-sets. <i>Left column: The original data-sets; Right column: The clustering results.</i>	81
4.6	The distribution of the first thirty eigenvalues of the Laplacian matrix in data-clustering (G4_1 synthetic data-set, with randomly assigned initial label probability values) at iterations 1, 7, 13, 19, 25, and 31.	82
4.7	The changing of entropy values during iteration (data-set G3). . . .	82
4.8	Evolution of the label probabilities (data-set G3). The horizontal axis gives the probability values and the vertical axis the components of the label probability vector.	83
4.9	Labeling results from kernel k -means.	84
4.10	Labeling results from kernel k -means.	84
4.11	The first two eigenvectors of G3.	85
4.12	The first three eigenvectors of G4_1.	85
4.13	The first two eigenvectors of RG.	86
4.14	Labelling results of real world data-sets. <i>Left: Iris; Right: Wine.</i> . . .	86
4.15	The first two eigenvectors of Iris.	86
4.16	The first two eigenvectors of Wine.	87
4.17	Matching three-way junctions. <i>(a),(b): Original image pair; (c),(d) & (e),(f): extracted edge contours; (g):results from contour pair (c),(d); (h): results from contour pair (e),(f).</i>	88
4.18	Matching results, CMU house data	89
4.19	Point correspondence matching results of image frames from a hand sequence.	94

4.20	The distribution of the first fifty eigenvalues of the weighted adjacency matrix \mathcal{A}_S in a feature correspondence matching example at iterations 1, 7, 13, 19, 25, 31, 37, 43, 49, 55, and 61. The data-sets involved are the hand sequences 09 & 11.	95
------	--	----

List of Tables

3.1	Kernel spectral matching, Algorithm I	43
3.2	Kernel spectral matching, Algorithm II	46
3.3	Matching results, Algorithm I (<i>Numbers of errors</i>).	60
3.4	Matching and labeling results, Algorithm II (<i>error%</i>)	60
4.1	The relaxation labelling algorithm	73
4.2	Relaxation labeling results on the toy example	77
4.3	Representations of label compatibilities	78

List of Notations

$G(V, E)$	A graph with node set V and edge set E , 7
$i \sim j$	Nodes v_i and v_j are neighbours of each other, 7
W_{ij}	The $(i, j)th$ element of edge-weight matrix W , 8
D	Diagonal matrix with entries the degrees of graph nodes, 8
L	The Laplacian matrix of a graph, 8
\mathcal{L}	The normalized Laplacian matrix of a graph, 9
P	The one-step transition matrix of a graph, 9
$K(\mathbf{x}_i, \mathbf{x}_j)$	A kernel function evaluated on data points \mathbf{x}_i and \mathbf{x}_j , 24
$\phi(x)$	The implicit map function in kernel methods, i.e., $K(\mathbf{x}_i, \mathbf{x}_j) = \langle \phi(\mathbf{x}_i), \phi(\mathbf{x}_j) \rangle$, 24
Y	The data point-set in feature correspondence matching, 32
X	The model point-set in feature correspondence matching, 32
Ω	The label set in relaxation labeling, 32
R	The label compatibility matrix, 40
$R_{ij}(\omega_i, \omega_j)$	The compatibility between labels ω_i at object i and ω_j at object j , 40
$R(\omega_i, \omega_j)$	The compatibility between labels ω_i, ω_j , 40
\mathcal{N}_i	The set of neighbours for object i , 40
$N(\mu, \Sigma)$	Gaussian distribution with mean μ and covariance Σ , 48
$P(t, \mathbf{x}, \Gamma)$	The semi-group transition function, 66
\mathcal{F}	The Fokker-Planck operator, 67
\mathcal{A}_S	The weighted adjacency matrix for the support graph G_S , 69

Acknowledgement

I thank my supervisor, Professor Edwin R. Hancock, for providing me the chance to studying in his group. I also thank him for his endless patient and very helpful advice during my study in this university. I also wish to thank my assessor Dr. Richard Wilson for all of his help. Many thanks to all the rest of the people in our group for their friendship and for all the discussions we had. Special thanks go to my sister, Hongqing Wang for her unconditional support to my every step.

Declaration

This thesis has not previously been accepted in substance for any degree and is not being concurrently submitted in candidature for any degree other than Doctor of Philosophy of the University of York. This thesis is the result of my own investigations, except where otherwise stated. Other sources are acknowledged by explicit references.

I hereby give consent for my thesis, if accepted, to be made available for photocopying and for inter-library loan, and for the title and summary to be made available to outside organisations.

Signed (candidate)

Date

List of Publications

1. Hongfang Wang, Edwin R. Hancock. A kernel view of point pattern matching. SSPR/SPR:361-369, 2004
2. Hongfang Wang, Edwin R. Hancock. Kernel Spectral Correspondence Matching Using Label Consistency Constraints. ICIAP, pages 503-510, 2005.
3. Hongfang Wang, Edwin R. Hancock. Correspondence matching using kernel principal components analysis and label consistency constraints. Pattern Recognition 39(6): 1012-1025 (2006)
4. Hongfang Wang, Edwin R. Hancock. Probabilistic Relaxation using the Heat Equation. ICPR (2) 2006: 666-669
5. Hongfang Wang, Edwin R. Hancock. A Graph Spectral Approach to Consistent Labelling. ICIAR (2) 2006: 57-68
6. Hongfang Wang, Edwin R. Hancock. Probabilistic relaxation labeling by Fokker-Planck diffusion on a graph. GbR 2007
7. Hongfang Wang, Edwin R. Hancock. Kernelised Relaxation Labelling using Fokker-Planck Diffusion. ICIAP 2007 (accepted)

Chapter 1

Introduction

1.1 The problem

The task of movement behaviour learning is concerned with understanding the contents of a given image sequence and to provide information of the interesting moving object in the scene. For this purpose, the motion analysis procedure usually starts with the selection and extraction of image features. This is usually followed by finding the correspondences between the extracted features in different image frames. After these two steps, further analysis can be performed, for example, to recover the 3D structure of the object, or the location of objects in the image frames. However, estimating the motion parameters and the structure of nonrigid objects by further analysis is not an easy task. To ease the problems, models are introduced for both the nonrigid objects and the motions that the objects are undergoing. These models are usually problem-specific and work under certain constraints. For example, to model the human body, stick figure models, 2D contour models and volumetric models [3] are used. To model handwritten digits, spline models are used in [65]. One important and influential example of a motion model is the diffusion process introduced by Isard and Blake in their CONDENSATION algorithm. The “point distribution model” developed by Cootes *et al.* in [29, 30] is an alternative way of modeling the motion of de-

formable objects.

Consequently, the aim of this thesis is to develop methods for locating correspondences between extracted image features using information concerning putative motion predictions elicited from the features. A key task under investigation in this problem is that of comparing detections in a scene at time t_1 with detections at an earlier time t_0 . This is approached using two different yet complementary methods, namely, feature correspondence matching and consistent labelling. When used alone, or when combined together, the two methods provide an effective way of tracking interesting objects in image sequences.

The main techniques of interest to us in this thesis are spectral graph theory and kernel methods on graphs. Graph theory has been extensively studied in the mathematics community for many decades. Graph spectral methods have been frequently used in sequence analysis problems such as image segmentation [111, 116] and feature correspondence matching [127, 112, 114, 22] over the past two decades. In fact, spectral graph theory is the first technique that is of interest in processing feature correspondence matching and unsupervised clustering tasks [9].

Kernel methods have been applied in statistical learning for problems such as classification and regression. Successful applications with supervised or unsupervised learning algorithms based on kernel methods, such as support vector machines [128, 108], Gaussian processes [85] and kernel PCA [109], can be found in the literature. They provide a way of applying linear classification or regression techniques to data in a non-linear way. It also provides a way for non-linear dimensionality reduction. Due to this ‘nonlinear’ capacity, there has been increasing interest in kernelizing traditional computer vision and pattern recognition algorithms recently, to develop new algorithms that are more efficient and robust tasks [108, 109, 81, 11, 28].

1.2 The objective

The objective of this thesis is to develop new graph spectral matching and relaxation labelling methods for the problems of feature correspondence matching and consistent labelling. The ‘nonlinear’ capacity of kernel methods is of particular interest here. Therefore the focus of this thesis is to explore kernel methods for the aforementioned problems. In particular, our first aim is to combine kernel methods with conventional graph spectral methods for the problem of rigid feature correspondence matching. This is based on the kernel principal components analysis (kernel PCA) method developed by Schölkopf, *et.al.*[109]. With the aid of a relaxation labelling process as well as kernel PCA, the second aim in this thesis is to develop a kernelized graph spectral matching algorithm for the problem of articulated feature correspondence matching.

Relaxation labeling is a powerful technique that was originally developed for assigning object labels to features extracted from images. It has also been used as an effective feature correspondence matching technique, e.g., [25, 136]. The second original contribution of the thesis is thus to develop a kernelized probabilistic relaxation labelling method for our consistent labelling and feature correspondence matching tasks. Drawing on the properties of random walks and diffusion processes on graphs from the literature, our new probabilistic relaxation labelling method is formularized as a diffusion process on a graph. By taking advantage of the eigenfunction representation of the kernel matrix, an efficient and effective relaxation labelling technique is developed.

The new algorithms are designed to accommodate uncertainties such as small deformation, occlusion, random point missing, and noise. Encouraging and effective results are thus expected from experiments on both synthetic and real world data-sets.

1.3 Thesis outline

The structure of the thesis is as follows. Chapter 2 first reviews the basic ideas of data representation in computer vision with an emphasis on graph representations. Then it provides a review of the main developments of the techniques of interest in this thesis, namely, spectral graph theory, consistent labelling, and kernel methods. Applications of these methods are also reviewed. Chapter 3 then presents the first contribution of the thesis, namely, generalized kernel spectral methods for rigid and articulated feature correspondence matching. The development is based on kernel principal components analysis on graphs and ideas concerning consistent labelling drawn from the literature on relaxation labelling. Chapter 4 presents a new probabilistic relaxation labelling method. The development is based on the idea of applying kernel methods with the diffusion equation. The state space of the diffusion process is represented by a graph, and the state-vector represent the probabilities of object-label assignment. The method is then applied to the problems of data classification and feature correspondence matching. That is, we develop a kernelized relaxation labelling method in a graph setting for a broad variety of tasks from computer vision and pattern recognition. Finally Chapter 5 concludes the thesis. The strengths and weakness of the described work are analyzed, and possible improvements and suggestions for future research are given.

Chapter 2

Literature review

This chapter reviews the main research literature related to the problem of motion behaviour learning in image sequences. The study of this problem has attracted considerable attention. Generally speaking, it depends on the type of motion and the structure of the object.

One way of learning the motion behaviour is to use parameterized models to describe the moving object itself, or to describe the motion of the object. Examples of the former include the point distribution model for describing deformable objects by Cootes, *et.al.* [30], the “card-board person model” by Ju, Black and Yacoob for describing a human being [75], and the “snake” model by Kass, Witkin and Terzopoulos for the representation of deformable objects in images [79]. Early work on motion analysis from the signal processing literature includes the Kalman filter [76]. The particle filtering approach was originally developed by Gordon, Salmond and Smith [51]. In computer vision the CONDENSATION algorithm developed by Isard and Blake [72] uses a diffusion equation to describe the motion of moving people in image sequences. Although attractive, model based object tracking requires prior knowledge of the extracted corresponding features between each image frame pair. The same requirement exists in many other approaches for the problem of motion behaviour learning. One example is the early approach of stereopsis described by Barnard and Fischler in [10]. An-

other example is the approach of structure and motion recovery by matrix factorization [125, 31, 98] which use landmarks to assist the localization of feature correspondences across image frames.

However, using landmarks is not convenient or even possible in many situations. Also tracking corresponding features is not an easy task. The reason for this is that uncertainties such as noise, occlusion, and outliers are introduced during the processes of image formation and feature extraction. Besides, objects that are of interest are usually not static in image sequences. Instead they undergo either rigid or non-rigid motions [3]. This chapter focuses on research described in the literature concerning rigid and articulated motion.

Many approaches have been developed for solving the feature correspondence matching problem. Among these, those based on spectral graph theory play an important role and are the main interest here. Consistent labelling methods have already been used effectively in feature correspondence matching [25, 137, 136, 49], as well as in other applications such as scene analysis and edge detection[9]. Consequently this literature review covers the main techniques and recent developments in the literature for these two areas. Specifically, an emphasis is placed on those techniques which combine kernel methods with various algorithms in a graph setting. Statistical and machine learning techniques, and various data representation models are also pertinent to the topic of the thesis.

2.1 Data representation

An important issue in computer vision and pattern recognition is the representation of data. Roughly speaking, data points are represented either by a collection of vectors, or by a graph with each graph vertex a data point in the original dataset and the edges represent affinity relationship. The former representation regards the data points residing in a d -dimensional data space for data points with d components. It is widely adopted in machine learning tasks.

Compared with the vector representation, graphs are richer and natural representations for data-sets. In addition to each individual data point, the pairwise relational information contained in a data-set is also represented. In this thesis the graph representation is adopted, and the emphasis of data representation is placed on affinity relations.

Among the various graph representations for data-sets, of interest in this thesis are those weighted simple graphs. That is, graphs which have no self-loops or multi-edges, and where each edge in the edge set is characterized by a weighting function. Suppose we have a given graph, denoted by $G(V, E)$, where $V = \{v_1, v_2, \dots, v_n\}$ is the finite set of vertices of the graph, and E is the edge-set which represents the structure and connection of the nodes in the graph. A node pair (v_i, v_j) is an element of the edge set E if and only if they are immediate neighbours of each other (denoted by $i \sim j$). Given a data-set, an intuitive graph construction is to take each data-point in the data-set as a vertex of the graph, and connect the vertices that are immediate neighbours to each other. A weight function may then be applied to each connection. However, the neighbourhood relationships are not always at hand and need to be extracted from the data. Many graph construction methods have been developed in the literature. The most commonly used include the Gabriel graph, the k -nearest neighbourhood graph, and the Delaunay graph [73]. Each of these constructions also defines a neighbourhood system on the graph, and thus the edge set. The Gabriel graph is defined as a set of point pairs in which an edge has a diametrical sphere (or hypersphere in higher dimensions) which does not contain any other points in the set. The Delaunay triangulation is another widely used technique to construct a graph representation for a given data-set. In a Delaunay graph, data-points x_i and x_j , which are represented by vertices v_i and v_j in the graph, are connected by an edge if and only if two cells of the Voronoi tessellation containing these two points share a common boundary. Both the Gabriel and the Delaunay graphs have been used in data classification or feature correspondence matching, e.g.,

[102, 45, 73]. The k -nearest neighbour graph is a more pervasively used representation. It is defined as a graph whose edge connections for each vertex are those k -closest neighbours to the vertex, with the ‘closest neighbour’ measured by some distance metric. Carreira-Perpiñán and Zemel [23] have a discussion of various graph constructions for different machine learning tasks, and suggest that graphs should be adapted locally to the structure of the data.

Once the graph is constructed, further graph theoretic analysis follows. One way is to generate minimal or maximal spanning trees for the graph [144, 139, 15]. A more frequent used approach is to use matrix representations of the graph. This approach offers the possibility of using the eigensystem of the graph to perform analysis. One of the most popular matrix representation for a graph is a ‘proximity’ or ‘affinity’ matrix, or the weighted adjacency matrix. In this thesis we are interested in weighted graphs. Thus hereafter we use the term ‘adjacency matrix’ and treat the traditional unweighted adjacency matrix as a special case of adjacency matrix of a graph. Denote the weighted adjacency matrix by W . Its elements W_{ij} are given by [27]

$$W_{ij} = \begin{cases} f_{ij} & \text{if } i \sim j; \\ 0 & \text{otherwise} \end{cases}$$

where f_{ij} is a weight function evaluated on the edge connecting vertices i and j , and reflects the strength of the connection. For undirected graphs, f_{ij} is symmetric; that is, $f_{ij} = f_{ji}$ and thus $W_{ij} = W_{ji}$. Consequently, the matrix W is also symmetric. This representation has been used in many computer vision tasks such as image segmentation [111] and feature correspondence matching [114, 127, 112].

Another pervasively used representation is the Laplacian matrix. Let $D = \text{diag}(\text{deg}_1, \dots, \text{deg}_n)$ be the diagonal degree matrix of the edge weight matrix W , where $\text{deg}_i = \sum_j W_{ij}$ is the degree of node i . The Laplacian matrix is then $L = D - W$. It is so named as it is commonly regarded as a discrete version of the Laplace operator on the graph. This matrix is of interest in various tasks such as spectral clustering, data embedding, and feature correspondence matching. In

the study of random walks on graphs, the Laplacian matrix also serves as a matrix of the rate of state transitions between each pair of states. It is also called the Q -matrix in the statistics and probability community. The second smallest eigenvalue of the Laplacian matrix also measures the mixing rate of a random walk. It is also called the Fiedler value, and its corresponding eigenvector is called the Fiedler vector. They are of broad interest since the early work by M. Fiedler [44], mostly in the applications for spectral graph partitioning [120] and spectral data clustering [131]. An alternative of the above matrix is the normalized Laplacian matrix of a graph, and is given by $\mathcal{L} = D^{-1/2}(D - W)D^{-1/2}$ [27]. The Laplacian matrix is closely related to the edge-vertex incidence matrix B [17]. The Laplacian matrix can be factorized into the edge incidence matrices using the relationship $L = BB^T$ (c.f.[27]). And the edge incidence matrix B can be regarded as the discrete version of a gradient on the graph G .

Another important matrix for a graph, which is of great interest in this thesis, is the (one-step) transition matrix P . The elements P_{ij} of this matrix represent the probability that a random walker moves to a node v_j in the next step given that he is currently at node v_i . That is, P governs the discrete time random walk on the graph G [32]. For weighted graphs, the elements of P are given by $P_{ij} = W_{ij} / \sum_{k, i \sim k} W_{ik}$. The matrix P can also be regarded as a special case of the adjacency matrix, with the matrix elements representing the similarity between each vertex pair of the given graph. The random walk is of interest in most scientific areas, and this matrix has been frequently used in data clustering [123], image segmentation [52], network partitioning [113], and VLSI design [54].

In computer vision, the graph representation of an image is usually constructed by taking its feature points as graph vertices. The edges of the graph then represent neighbourhood relationships between the vertices. The feature points can be the complete set of pixels in an image, or it can represent segmental entities such as edge points, corners, or image regions. In most situations, edges connecting the vertices corresponding to feature points in images are given according

to the definition of a neighbourhood system. The graph construction methods discussed above can also be applied to these image feature point-sets. A simple and widely used technique for defining a graph is to give each data-point pair in the data-set a connection; that is, the graph is assumed to be fully connected. Then one weights this connection by a similarity or dissimilarity measure for the two feature points. This defines the weighted adjacency matrix for a graph. Furthermore, a threshold value can also be specified *a priori* to discard those connections with the smallest similarity values. A commonly used weight function is the Gaussian $w_{ij} = e^{-d_{ij}^2/\sigma}$, where d_{ij} is the Euclidean distances between two points \mathbf{x}_i and \mathbf{x}_j , and σ is the parameter which controls the meaning of ‘closeness’ or ‘similarity’ between data-point pairs. The matrix representations of the graph discussed above can thus be easily computed from this weighted adjacency matrix.

2.2 Consistent labelling

Many problems in computer vision and machine learning can be posed as of consistent labelling. Techniques for achieving consistent and unambiguous labelling have been developed and applied in these areas since the very early stages [40]. Applications of these techniques such as edge detection [80], scene interpretation [106], image segmentation [60], and data clustering[101], where objects such as image pixel values, extracted image features, or data samples are assigned to their corresponding object or group labels are vast in the literature [74]. Among the large number of algorithms for labelling, of interest here are those techniques which have successful applications in the computer vision area. In particular, we are interested in unsupervised or semi-supervised data classification techniques in machine learning, and those which are based on contextual-constraints and graph theoretical techniques. This section focuses on graph based unsupervised and supervised classification techniques, and various relaxation labelling tech-

niques. Belief propagation is closely related to the spirit of relaxation labelling, and a brief review of classification using belief propagation is also included.

2.2.1 Data clustering

Data clustering is a general problem in pattern recognition. It is the process of classifying a given set of data into meaningful groups, by some similarity measure, where the number of groups may be known beforehand. Thus each data object is assigned a group label after a clustering process and, data objects within the same group are more similar than those belong to different groups. Many techniques have been developed for this problem in the literature [40] under the name of unsupervised classification. Early approaches of unsupervised data clustering methods include k -means, minimal spanning trees, and nearest neighbour based hierarchical clustering methods using dendrograms [74]. Later on statistical physics and random walk theories are also used to develop data clustering methods [67, 15, 54, 124]. When the labels of a small percentage of the data-points are known, semi-supervised classification methods are used. These two families of techniques can be applied with little or no alteration to applications in computer vision, for example, image segmentation (e.g., [52]). Among the above techniques, of interest in this thesis are graph based techniques. Graph theoretical clustering approaches can be found in the literature from the early 1970s. One example is the use of minimal spanning trees [144] for the problem of VLSI circuit design. Subsequent applications of using spanning trees for data clustering also exist, see [139].

Random walks are discrete time stochastic processes with a set of countable states. Their properties have been applied to clustering in various areas. As early as 1981, Hughes, Shlesinger and Montroll [70] presented a discussion of random walks which exhibited clustering properties. An early application of random walk based clustering is its use by Hagen and Kahng for VLSI circuit design [54]. In the machine learning area, Tishby and Slonim [124] used the adjacency matrix

of a graph as the transition probability model of a discrete time random walk, and the information loss during the Markov relaxation was used for data clustering. Recently random walk theory has also been used to develop semi-supervised classification techniques. For example, Szummer and Jaakkola [123] used the random walk technique to cluster “partially labelled data”. To do this they set their labelled data-points as the absorbing states of the walk and posed the clustering problem in terms of the probability of reaching one of these states. Yen, et.al. [141] used random walks on graphs for bipartite data clustering. These methods are mostly problem specific. Moreover, they do not address the problem of exploiting knowledge concerning the semantic constraints that exist between different class labels, nor do they use initial confidence in the assignment of class labels. One exception is the work of Zhu, Ghahramani and Lafferty [147], in which random walk theory is used for semi-supervised data clustering in a continuous state space. Their method also makes use of initial label confidence for each object, but like most of the data classification techniques, their model only deals with two possible labels. However, prior knowledge about semantic constraints between labels and initial label confidence are common in problem domains such as video surveillance [69] and target tracking [1, 91] where interesting objects are usually manually specified at the beginning of a data sequence.

While most of the random walk based data classification methods assume a discrete time transition step with a discrete state space, methods based on continuous random walk theory can also be found in the literature. The work by Hughes, Shlesinger and Montroll [70] we mentioned above is an early discussion of clustering using continuous time random walk. In [140], Yeang and Szummer also assumed a continuous distribution of the state space, and derived the random walk approach in the limit to a diffusion equation. In fact, it is a common assumption in the computer vision and pattern recognition community that given sufficient data points, a graph is capable of representing the underlying manifold. Correspondingly, the graph Laplacian on a graph converges to a con-

tinuous Laplace operator on the manifold [11, 64, 117]. In this thesis we also use a continuous Markov process for the tasks of labelling and feature correspondence matching.

2.2.2 Spectral graph methods

Graph spectral methods are classical tools for graph partitioning. As early as 1970, Hall [55] pointed out that the second smallest eigenvector of the Laplacian matrix (the “disconnection matrix”) gave the nontrivial optimal solution of minimizing the overall distances between connected nodes, and this eigenvector separated nodes into local clusters. Early development of using this eigenvector for data clustering can be traced back to the work by Donath and Hoffman in the early 1970s [38, 39]. Shortly after, Fiedler also used the second smallest eigenvector of the Laplacian matrix to bipartite data-sets [44]. Subsequent applications can be found in areas such as network partitioning [113], VLSI circuit design [6], and sparse matrix partitioning [99]. A theoretical analysis of spectral graph partitioning by using the Fiedler eigenvector on bounded planar graphs and finite element meshes can be found in the work of Spielman and Teng [120].

Recently, spectral clustering methods have also attracted widespread interest in the machine learning community [90, 77, 46, 8, 145]. Spectral methods share the feature of using a weighted graph to represent the data. They cluster data in an unsupervised manner using the eigenvectors of the weighted data proximity matrix for a given data-set. Apart from using the Fiedler vector of a Laplacian matrix or the largest eigenvector of an adjacency matrix, using multiple leading eigenvectors is suggested by several researchers [6, 90, 77]. Fischer and Poland in [46] gave a useful comparison of different spectral clustering methods.

In computer vision, an early application of spectral graph theory for performing image segmentation is introduced by Scott and Longuet-Higgins [111]. Here, a proximity matrix H was constructed from the extracted feature points to capture the similarities between the points. Then the first M eigenvectors of this

matrix were used to form a Q matrix which had elements equal to the cosine angles between the point pairs. These matrix elements indicated whether two points were in the same group (with a value close to 1) or not (with a value close to 0). Their experimental results showed good performance for feature grouping. Later on, Perona and Freeman also developed an algorithm based on graph spectra [96]. They performed spectral clustering by thresholding the components of the largest eigenvector of an affinity matrix computed from the given data-set. Weiss [132] reviewed various spectral clustering methods, and proposed using a normalized adjacency matrix instead of the unnormalized one in the algorithm of Scott and Longuet-Higgins. A recent interesting example of graph spectral image segmentation is the normalized cut (Ncut) of Shi and Malik [116]. They commence from a graph theoretical viewpoint, and defined an objective function based on the normalized cut quotient. They used the second largest generalized eigenvector of the graph to approximate the optimum bipartition of the graph. This eigenvector is in fact the largest eigenvector of the transition matrix of the graph, and their method is closely related to discrete random walk clustering on graphs, as analyzed in [86]. Although the Ncut algorithm cut an image into two segments, multiple segments of an image can be obtained by a recursive application of the algorithm. In an attempt to obtain multiple ($k > 2$) partitions in the same time, Yu and Shi extended the Ncut algorithm and proposed to use the k smallest eigenvectors instead of only a single eigenvector [143].

Despite the fact that the above algorithms give very good performances, the spectral clustering algorithms are heuristic in nature. As a result, the problem of how to characterize the behaviour of spectral clustering algorithms has attracted considerable interest in the literature [120, 53, 90, 77]. For example, based on an analysis of several spectral graph partitioning methods, Spielman and Teng [120] gave examples of when good spectral partitioning results could be expected on bounded planar graphs and finite element meshes. By contrast, Guattery and Miller [53] analyzed when spectral methods gave poor clustering results. To im-

prove the spectral partitioning and clustering results, Kannan, Vempala and Vetta proposed a bicriterion to guide the partitioning [77]. However, a full analytical understanding of spectral partitioning is still unclear. In addition, the use of the Fiedler vector for partitioning, although successful, still present problems especially when the data-set is very large. In such cases, the computation of the second smallest eigenvector is affected by round-off error and the precision of the computing machine. Thus the potential for improving spectral methods is still considerable.

2.2.3 Relaxation Labelling

Relaxation labelling processes are a class of mechanisms that were originally developed to deal with ambiguities and noise in object labelling. The key element is an update rule based on the contextual information for each node to be labelled ([71, 130, 80, 58]). They have been used successfully for edge labelling, image segmentation and feature pattern matching in the computer vision literature for more than three decades [130, 106, 47, 80, 25, 61, 146]. The origin of the process can be traced back to the line-labelling technique of Waltz [130] in the early 1970s. However, interest in developing the theory of relaxation labelling methods and applying them to pattern analysis tasks still continues [42, 146, 149].

The approaches of relaxation labelling may be classified as stochastic [47] or deterministic [80, 137, 25]. The difference between them is that in a stochastic relaxation process, the labelling probability may be chosen stochastically after some iterations to avoid local minimum which is unavoidable in deterministic processes. Relaxation processes may also be classified as discrete [130] or continuous [106, 71]. These processes share the common feature of seeking a globally consistent labelling. The definition of consistency is formally given by Hummel and Zucker [71]. They also showed that consistency can be achieved using optimization.

Discrete relaxation starts with a complete set of possible labels residing on

each object. During each iteration inconsistent labels are discarded using a constraint filtering process. At convergence the multiple labels may still reside on single objects. Waltz [130] reports a method for recovering multiple ambiguous labelling. Hancock and Kittler [57] formulated discrete relaxation labelling as a maximum *a posteriori* (MAP). The algorithm was successfully applied to the problem of edge labelling. An early example of the continuous case is the probabilistic relaxation labelling developed by Rosenfeld, Hummel and Zucker [106]. In [106] the object-label assignments are represented by a probability vector. The probability vector is updated in a non-linear fashion using a support function to combine evidence for different object-label assignments. Subsequent work has refined the process in a number of ways. For instance, Faugeras and Berthod [43] have posed relaxation labelling as an optimization process. Hummel and Zucker [71] have shown how relaxation labelling can be viewed as satisfying a set of variational inequalities. Hancock and Kittler [80] have posed the probabilistic relaxation as Bayesian evidence combination. This makes it different from the heuristic formulations in previous approaches. This development also allows support functions to be constructed for a number of different neighbourhood systems. Pelillo [94] has shown how probabilistic relaxation can be viewed in a game theoretic setting using the replicator equations. Christmas, Kittler and Petrou [25] developed a probabilistic relaxation framework for the specific problem of attributed relational graph matching. Using binary attribute relations, they have successfully applied relaxation labelling to the matching of features extracted from image pairs. A more recent example is the work by Zheng and Doermann [146] in which the process was used to aid the task of feature correspondence matching.

In relaxation labelling, it is common that objects being labelled are considered as connected by a network (with neighbourhoods) and viewed as a graph with nodes the objects and the arcs the compatibility relationships between nodes. In [71], the neighbourhood system is defined as the whole object set and the sup-

port for a label λ on object i is computed from all the remaining objects in the network. In [80], the neighbourhoods are distinguished as if whether they are directly interacting or non-interacting objects. The support comes only from those objects that are directly interacting. In the case of image labelling, the neighbourhoods are chosen to lie in a 3×3 window centred at the pixel of interest. This significantly reduces the computational overhead.

Belief propagation [93, 131, 122] is also a local evidence combination and propagation process that is widely used in machine learning. One of the attractive features of belief propagation networks is that they can be used for Bayesian inference. To construct belief networks Markov properties are usually assumed. As a result each node in the network depends only on its immediate parent nodes, and is independent of the nodes in the causal past given the current immediately connected (parent) nodes. An interesting example is given by Weiss in [131] where belief propagation is used for the task of image interpretation. An analysis of the relationship between classical relaxation methods based on pairwise arithmetic support and belief propagation is also included. An advantage of belief propagation over traditional relaxation labelling is that experimentally its rate of convergence is demonstrated to be faster. However, more complex relaxation labelling methods based on product support also converge more rapidly than those based on simple arithmetic support [80, 62].

2.3 Feature correspondence matching

2.3.1 The problem of feature correspondence matching

Generally speaking, the problem of feature correspondence matching is to find a one-to-one correspondence of each feature point from the model image in the data image. Ideally the process should also be capable of detecting outlier feature points from either of the image frames being matched. The features could be the detected interesting points, edge segments, corners, or other meaningful objects

in an image. The difficulties in solving the feature correspondence problem lie in the fact that uncertainties are unavoidable in the process of feature detection. For example, the camera used to capture the images, or the process of digitizing the images frequently introduce noise into images and this noise affects the accuracy of feature extraction. In addition, the feature extraction algorithms themselves also inevitably introduce uncertainties. Even if these processes are accurate, outliers and occlusions may still appear (and is usually the case in real world images). Besides, the objects are usually not static in different image frames. They may undergo rigid or nonrigid motion.

The idea of correspondence matching is to use the invariant properties of given unique features that are most similar under changes in viewpoint. The invariants are usually related to the object motion type which can be either rigid or non-rigid [2, 3]. In rigid motion, no deformation of the object will be detected. While only the rotation and translation are present, the relative distances between any two points on the object will be preserved. Because of these properties, it is not too complicated to model the shape and the motion of a rigid object. The problem of nonrigid motion analysis is generally harder than the rigid case since in addition to translation and rotation, there are additional geometric transformations present. Here we are particularly interested in articulated motion [2, 3]. Henceforth, we will use features extracted from objects that undergo affine rigid or articulated motion. For rigid motion, the Euclidean distance between two feature points is the most frequently used measure. For nonrigid motion, the measure is usually combined with a motion model. Other properties have also been used, for example, the relative position with respect to other features, the structure of the features, pixel intensity values, the shape, colour, or texture of the features have all been used in the literature [10, 110, 25, 107].

2.3.2 Graph spectral methods for correspondence matching

The literature contains several examples of using the eigenvalues and eigenvectors of point affinity matrices to locate the one-to-one correspondences between feature point-sets. These methods show the common feature of commencing from constructing a matrix representation of the given point-set using the distance between point features. This is similar to the representation used in spectral graph partitioning. The idea underpinning spectral methods is to embed point-sets into a common eigenspace, and to find correspondences by performing alignment in this space. This is usually accomplished by applying the eigendecomposition techniques to the point affinity matrix.

The work by Umeyama ([127]) is an early example of attempting to use eigendecomposition to find the one-to-one correspondences between point-sets. The work studies the weighted graph matching problem, where the graphs can either be directed or undirected. A weighted adjacency matrix is chosen as the graph representation: $A = [A_{ij}]$, $A_{ij} = w(v_i, v_j)$ if $i \neq j$, and $A_{ij} = 0$ otherwise, where v_i, v_j are the i th and j th vertices of the graph. Umeyama tried to find a permutation matrix P that minimized the quantity $J(P) = \|PA_G P^T - A_H\|^2$, where A_G and A_H were adjacency matrices of undirected graphs G and H , respectively. Umeyama showed that the permutation matrix P could be obtained by minimizing $\text{tr}(PU_H U_G)$, where U_H and U_G were the absolute matrices of the left singular vectors of A_H and A_G , respectively. He also showed that by constructing the corresponding Hermitian matrices from the adjacency matrices of graphs G and H , a nearly optimal solution for the permutation matrix could also be obtained for the case of directed graphs. The proposed method required that the two graphs had the same number of nodes and the method worked well when the two graphs were isomorphic or nearly isomorphic.

Scott and Longuet-Higgins ([112]) adopted a different approach to finding point correspondences by calculating matrix operations. They located feature correspondences by first building an inter-image proximity matrix by applying a

Gaussian weighting function to the Euclidean distances between feature points in the different images being matched. Correspondences were located by performing singular value decomposition on this inter-image proximity matrix. This method failed when the rotation or scaling between the images being matched was large.

To overcome this problem, Shapiro and Brady ([114]) constructed intra-image proximity matrices for the individual point-sets being matched with an aim to capturing relational image structure. The eigenvectors of the individual proximity matrices were used as the columns of a modal matrix. Correspondences were located by comparing the rows of the modal matrices for the point-sets under match. This method can be viewed as projecting the individual point-sets into an eigenspace, and seeking matches by locating for closest point correspondences. To make the algorithm more robust, Shapiro and Brady used the eigenvalues to scale their corresponding eigenvectors and place more emphasis on the more significant eigenvectors.

A common weakness with existing spectral methods is that they are particularly sensitive to structural variations in the point-sets. Carcassoni and Hancock have attempted to improve the robustness to point-jitter by using robust error kernels instead of the Gaussian [22]. They have overcome problems due to differences in the structure of the point-sets by using spectral clusters [21]. Kosibov and Caelli [18] have extended Shapiro and Brady's method by scaling the eigenvectors, and seeking correspondences by searching for matches that maximize the inner product of the normalized eigenvectors. A more sophisticated approach which allows for rotation and scaling in the eigenspace is to apply multidimensional scaling to the individual point-sets, so as to preserve their relational arrangements, and to seek correspondences by performing Procrustes alignment in the eigenspaces [83].

2.3.3 Correspondence matching by relaxation labelling

Another efficient way of finding feature correspondences between two images is to use relaxation labelling methods [25, 137, 45, 35, 4, 92]. The idea underlying this approach is that by taking advantages of the relational constraints or contextual information, data-points which have a similar context or neighbourhood support can be considered as corresponding point pairs in the two images.

Christmas, Kittler and Petrou [25] have developed a probabilistic relaxation process for the structural matching problem. In their work, each feature object is represented by a unary measurement x_i which normally takes the form of the object i 's coordinate values in an appropriate coordinate system, and a binary measurement \mathcal{A}_{ij} which represents the relationship between object i and object j . The support function in their updating function has the sum-product form, and includes the binary measurements \mathcal{A} . The density function $p(\mathcal{A}_{ij}|\theta_i = \omega_\alpha, \theta_j = \omega_\beta)$ is interpreted as the compatibility coefficients for label ω_α on object θ_i with a label ω_β assigned to its neighbour θ_j . Christmas, Kittler and Petrou claimed that their approach was suited to many-to-one matching. The method operated with different feature-set sizes. By using binary measurements in the support function, matching performance was improved.

In contrast with Christmas, Kittler and Petrou's probabilistic relaxation approach, Wilson and Hancock [137] have solved the structural matching problem using discrete relaxation labelling. A probability distribution for the relational errors that occur when there is significant structural corruption in the data-sets is included to improve the matching performance. To overcome the problem of outliers, measurement errors and other uncertainties, and to render the matching process more robust, relaxation labelling has been enhanced by using additional information. Cross and Hancock developed a dual-step EM algorithm in [35] for solving the correspondence matching problem and recovering the transformation matrix simultaneously. A probabilistic relaxation labelling process is included for gating the matching of the two point-sets under a transformation matrix. The re-

laxation process successfully computes the best correspondence matching and provides a basis for the entire correspondence, structure and motion recovery process. In the work by Pajares [92], the probabilistic relaxation labelling process was worked in conjunction with the fundamental matrix and epipoles for the problem of stereo image matching.

Recently, Ahmadyfard and Kittler [4] have applied a probabilistic relaxation technique to object recognition. They use affine invariant image regions segmented from images using the colours of the image pixels and the quantities are normalized to improve the robustness. Each feature-set is first represented by an attributed relational graph (ARG). Different object models are composed into a single graph representation. As in [25], in addition to unary measurements, binary measurements are also used in combining support from neighbourhood objects. To overcome problems of complexity, the admissible labels for each object in the scene are pruned before each iteration. A more recent application of using relaxation labelling for point feature correspondence matching can be found in the work by Zheng and Doerman [146].

2.3.4 Other correspondence matching methods

Apart from graph spectral methods and relaxation labelling, there are many alternative methods developed for the problem of feature correspondence matching. The EM algorithm is a natural choice for point pattern matching due to its capacity to deal with incomplete data. As a result there are many attempts to perform correspondence matching using the EM algorithm [35, 83, 36, 20, 84]. These approaches usually aim to recover transformations as well as locating the feature correspondences.

An early example of using the EM algorithm for feature correspondence matching is the work by Cross and Hancock [35]. In their work a Bayesian probabilistic framework is first built to cast the point matching problem into a modified EM algorithm. They refer to it as the dual-step EM algorithm. The approach simulta-

neously finds point correspondences and recovers the transformation matrix underlying the motion. The transformation matrix is defined for the feature point set from the model frame to the target frame. Commencing from an initial set of values, in each E-step a new expected log-likelihood value is computed from the resulting transformation matrix of previous step and the probability of feature point correspondences. In each M-step, new parameters are computed by maximizing the objective function. The correspondence probabilities are computed in the E-step. As mentioned above, a relaxation labelling process is incorporated into the updating processes to gate the contributions to the expected log-likelihood function. Carcassoni and Hancock [22] later extended this approach by using spectral information in the gating function.

Luo and Hancock [84] have also used the EM algorithm for solving the point pattern matching problem. Commencing from a probability distribution for matching errors, the graph matching problem is posed as a maximum likelihood problem and is solved using the EM algorithm. The aim is to accommodate significant levels of structural corruption. A singular value decomposition approach is adopted for finding the correspondences among the two data-sets. Another interesting correspondence matching method is the work by Dellaert, Seitz, Thorpe and Thrun [36] where the feature correspondences, the point structure and the camera motion are recovered simultaneously by using the EM algorithm.

Random walk theory has also been used to solve the correspondence matching problem. The idea is to locate correspondences by comparing the random walks (or their properties) on the graph. One example is the work of Robles-Kelly and Hancock [104] which compares the strings generated by two random walks on different graphs. Another example is the work by Qiu and Hancock [100] in which a spanning tree is constructed on the graph, and commute time is used as a similarity measure for the two graphs.

2.4 Kernel methods

Kernel methods have been attracting increasing interest recently in efforts to improve classical pattern analysis methods [109, 108, 81, 11, 28, 34, 8]. There are a number of reasons for this. Firstly, kernel methods are elegant and allow hitherto formally opaque techniques to be understood in a more transparent geometric way. Secondly, they frequently admit the use of eigenvector methods to locate the required solution. Finally, they offer efficiency advantages in the sense that using a kernel function takes the advantage of utilizing a high or even infinite dimensional space without the curse of dimensionality. A large variety of kernel based algorithms have been developed in the machine learning community for tasks such as classification, regression and dimensionality reduction. For example, support vector machines (SVM) [128], Gaussian processes [85], and kernel principal components analysis (kernel PCA) [109] are all effective kernel methods for various learning tasks. Applications of these methods in the computer vision area are also increasing in the literature. In the following review, first a general description of kernel methods in machine learning is given, then kernel functions with dynamic properties on graphs are reviewed. This section ends with a general discussion of applications of kernel learning methods in the literature including those in the computer vision area.

2.4.1 The kernel methods

Kernel methods usually involve a first step of applying a kernel function $K(\mathbf{x}_i, \mathbf{x}_j)$ to each observed data pair $(\mathbf{x}_i, \mathbf{x}_j)$ to form a Gram matrix of the observed data set. In this way, the original data are implicitly mapped into a higher, possibly infinite, dimensional feature space. That is, the operation with a kernel function can be regarded as a mapping $\mathcal{T} : \mathbf{x} \mapsto \phi(\mathbf{x})$ from the original data space \mathbb{R}^d to a new space \mathbb{F} . The kernel function acts equivalently as a dot product on each data pair in the feature space \mathbb{F} ; that is, $K(\mathbf{x}_i, \mathbf{x}_j) = \langle \phi(\mathbf{x}_i), \phi(\mathbf{x}_j) \rangle$. By this operation,

it is expected that the unknown intrinsic structure of the data can be uncovered in the new feature space. After the mapping, a second step is to apply various machine learning techniques to the data in the new feature space. The mapping operation \mathcal{T} could be linear, as in early traditional approaches. But for real world data, linearity does not always exist. Of more interest thus is to develop nonlinear mappings and to provide good embedding for various data in a new feature space.

A problem when choosing the function $K(\mathbf{x}_i, \mathbf{x}_j)$ is that not every function is guaranteed to give a valid feature space. One way of searching for a valid kernel function is to draw on Mercer's theorem [128] which states that any continuous symmetric function $K(\mathbf{x}_i, \mathbf{x}_j)$ that is positive semidefinite is ensured to be a kernel for some valid feature space. That is, any positive semi-definite and symmetric function $K(\mathbf{x}_i, \mathbf{x}_j)$ can be written in the form of a dot product $K(\mathbf{x}_i, \mathbf{x}_j) = \langle \phi(\mathbf{x}_i), \phi(\mathbf{x}_j) \rangle$. The original data can then be regarded as been mapped into a Hilbert space characterized by the dot-product. This provides a very flexible way of choosing the kernel mapping function.

The kernel function can also be regarded as a similarity or dissimilarity function in measuring the closeness of two data points. A great advantage, and in fact the original motivation, of using kernel functions is to represent the data by a group of simpler and more representative functions, referred to as the eigenfunctions of the kernel function. This is equivalent to performing an orthogonal transformation on the original data which transfers the data from the original space into another space spanned by the corresponding orthogonal basis. Viewed from the perspective of data classification, the new representation separates the original data in a way that the within class distances are made much smaller than the between-class distances. A good example of this behaviour is the kernel Fisher discriminant [87]. Usually a linear classification technique can be applied in the kernel space, and this gives better results than those obtained in the original space.

2.4.2 Kernels on graphs

Kernel methods provide promising techniques for a vast number of machine learning tasks, and given that graphs are natural representations of objects in the real world, it is of obvious interest to combine the two techniques together. Recently there has been interest in seeking effective kernels for graph representations of data [118, 81, 124, 63]. In fact, for discrete data samples a graph representation is implicitly included in almost all kernel based learning methods. The Gram matrix obtained from a kernel function is just the proximity matrix of the graph on the data-set. Recent efforts at designing kernels which exploit the discrete structure of the data, include that of Haussler who has developed a convolution kernel for structures such as strings, trees, and graphs [63].

The diffusion kernel developed by Kondor and Lafferty [81] takes advantage of the fact that the exponentiation of a symmetric matrix always gives a symmetric and positive semidefinite matrix to construct kernel matrices. Furthermore, this construction satisfies the diffusion equation, giving good properties and meaningful interpretations in terms of heat diffusion on a graph.

Random walk theory and other Markov processes have also been used to develop new kernel methods for data represented by a graph. In [24], Chapelle, Weston and Schölkopf used the radial basis function (RBF) kernel matrix as the transition matrix of a random walk. They used this to construct new kernels by transforming the eigen-spectrum of the RBF kernel matrix (that is, the affinity matrix). They also showed the connections between the random walk kernel and various spectral clustering methods. This construction of new kernels is central to other work too. For example, Smola and Kondor [118] established a connection between regularization theory and Laplace operators on graphs. They provide transformation functions which give the diffusion kernel as a special case. Belkin and Niyogi [12] also constructed kernels which was a transformation of the affinity matrix spectrum. More recently, Zhu, Kandola, Ghahramani and Lafferty [148] developed similar methods to obtain new kernels by performing transfor-

mation on the eigenspectrum of the Laplacian matrix of a graph representation. The kernel was then used for developing a semi-supervised learning method for data classification.

2.4.3 Applications

Applications of kernels on graphs can be found both in the machine learning and in the computer vision literature. In machine learning, kernel based clustering methods or semi-supervised learning by kernel methods are enormous [13, 24, 129, 123]. For example, Weston, *et.al.* [133] performed semi-supervised learning for protein classification by improving the cluster kernels proposed by Chapelle, Weston and Schölkopf [24], and Szummer and Jaakkola [123]. Another example is the work by Vert and Kanehisa [129] which applies the diffusion kernel to the problem of biological data analysis.

Kernel methods can also be used as a way of dimensionality reduction. An interesting development is kernel PCA developed by Schölkopf, Smola and Müller [109]. Conventional PCA provides a linear orthogonal transformation of the data from a high dimensional space to a low dimensional one which maximally preserves the variance of the original data. This is done by computing the eigenvalues and eigenvectors of the covariance matrix, and using the first few normalized eigenvectors of the covariance matrix as the principal projection axes for the training data. The method minimizes the residuals of the data points projected into the common eigen subspace, and thus gives an optimum representation of the original data in the projection space. Kernel PCA [109] can be regarded as a generalization of conventional PCA from a linear to a non-linear transformation space. In the literature, it has been shown to provide a better way of recovering the underlying principal components of the given data, see for example, the de-noising application in [88]. The main difference between kernel PCA and conventional PCA is that kernel PCA first applies a kernel function $K(\mathbf{x}_i, \mathbf{x}_j)$ on each data-pair of the given data-set, which is equivalent to a transformation function

$\mathcal{T} : \mathbf{x} \mapsto \Phi(\mathbf{x})$ to map the data from the low dimensional space into a new feature space \mathbb{F} of higher, possibly infinite, dimension. An eigendecomposition is then performed to obtain the principal components of the feature space and map the data to a lower dimensional space. This gives kernel PCA the property of extracting non-linear features from the data-set and makes it a powerful tool in many pattern analysis applications. The kernel function here plays an important role in kernel PCA. The most frequently used kernels are Gaussian kernels, polynomial kernels, and sigmoid kernels. However, the Gaussian kernel and the polynomial kernel are of particular interest in this thesis because of their invariance properties under rigid transformation and reflection.

Since kernel PCA provides a flexible way of embedding a given data-set into a new feature space according to the properties of the data, it provides a potentially important pre-processing step for many computer vision and pattern recognition tasks. A natural development would be to combine graph spectral methods and kernel PCA to develop new algorithms. In fact, such developments can already be found in the literature, especially for the task of data classification. The kernel Fisher discriminant analysis method developed by Mika, *et.al.* [87] is an attempt to combine kernel methods with traditional learning methods. It can also be used as a method for data dimensionality reduction. Brand and Huang [16] have analyzed the relationship between kernel methods and spectral clustering methods, and have developed a new method based on a polarization theorem for spectral embedding and clustering. They have viewed the affinity matrix as a kernel function $K(\mathbf{x}_i, \mathbf{x}_j)$, and the mapped feature space was represented in a polar form. The eigenvectors and eigenvalues were assumed to be sampled from the Fisher-Bingham distribution. Their experiments on “challenge problems” show interesting results for data clustering and embedding. Bengio, *et.al.* [14] have also noticed the equivalence of kernel PCA and spectral clustering. Its connection with the Nyström method has been pointed out by Williams and Seeger [134]. The technique is used to solve the out-of-sample problem using

kernel spectral methods with encouraging experimental results.

2.5 Conclusions

We have reviewed the main developments in the literature for the problems of consistent labelling and feature correspondence matching, with emphases on spectral graph theory, probabilistic relaxation labelling, and kernel methods.

Despite their heuristic nature, from the review, first of all we can have the conclusion that graph spectral methods are promising in a wide area of scientific research, including VLSI circuit design, network partitioning, data clustering, and feature correspondence matching. Interests in developing new spectral methods are still increasing. However, their ability to cope with structural distortion together with their heuristic nature prevent them from being applied to more difficult tasks.

Secondly, we conclude that consistent labelling techniques were also of great interest in the computer vision and pattern recognition literature. The labelling techniques themselves are very useful as they can be applied to solving tasks such as scene labelling, object recognition, and data clustering. More interesting is the fact that they can be combined with other techniques to provide more efficient solutions for difficult problems. For example, relaxation labelling utilizes the semantic constraints contained in the given object labels and thus provide more information.

Finally, we conclude that kernel methods are attracting considerable interest in computer vision and pattern recognition, as well as in machine learning. Their nonlinearity property offers the potentials for developing new algorithms for more “challenging” data analysis tasks. The flexibility choice of kernel function makes it possible to bring an almost unlimited variations into kernel methods. Graph kernels, especially those based on random walks and diffusion processes on graphs have already shown their promise as tools for analysing desired

data structures. We thus believe that by combining graph spectral theory, kernel methods and consistent labelling, more challenging object labelling and feature correspondence problems including our nonrigid motion behaviour learning problem, can be solved with satisfactory results.

Chapter 3

Kernel spectral feature correspondence matching

This chapter investigates spectral approaches to the problem of point pattern matching. Two contributions are made. First, we consider rigid point-set alignment. Here we show how kernel principal components analysis (kernel PCA) can be effectively used for solving the rigid point correspondence matching problem when the point-sets are subject to outliers and random position jitter. Specifically, we show how the point-proximity matrix can be kernelised, and how spectral correspondence matching can be transformed into one of kernel PCA. Second, we turn our attention to the matching of articulated point-sets. Here we show how label consistency constraints can be incorporated into the definition of the point proximity matrix. The new methods are compared with those of Shapiro and Brady [114] and Scott and Longuet-Higgins [112], together with multidimensional scaling [33]. Experiments on both synthetic and real world data are provided, with encouraging results.

3.1 The problem

The idea in this chapter is to allow label consistency constraints to be incorporated into the spectral point correspondence process. Specifically, the objects we are interested in here undergo articulated motion between image frames. That is, the objects under investigation are composed of rigid components, with each rigid component has its own rigid motion, but the overall motion is non-rigid. We start from the problem of objects undergoing rigid motion, and later on this is treated as a special case of our articulated motion process.

Suppose that feature points are already extracted from each image frame. In each data-set the feature points are given in the form of $Y = \{\mathbf{y}_1, \dots, \mathbf{y}_n\}$ and $X = \{\mathbf{x}_1, \dots, \mathbf{x}_m\}$ for the data point-set and the model point-set, respectively. The points $\mathbf{x}_i \in X, i = 1, \dots, m$ are represented by their image co-ordinate values; that is, $\mathbf{x}_i = (x_{i1}, x_{i2})$. Those of $\mathbf{y}_i \in Y, i = 1, \dots, n$ are also represented by their respective image co-ordinates. Furthermore, for non-rigid objects a label probability vector is also attached to each feature point. This label information specifies to which rigid component each feature point is likely to belong. Assume there are l labels in each feature point-set, which represent the l rigid components possibly existing in each image frame. Then an image point \mathbf{x}_i can be assigned to each label $\omega_j \in \Omega$, where $\Omega = \{\omega_1, \dots, \omega_l\}$, with a probability. Denote it by $P(\theta_i = \omega_j)$. Then the vector $\mathbf{p}(\theta_i) = (P(\theta_i = \omega_1), \dots, P(\theta_i = \omega_l))^T$ represents the probability of assigning each of the possible labels to the point, with $0 \leq P(\theta_i = \omega_j) \leq 1$, and $\sum_{j=1}^l P(\theta_i = \omega_j) = 1$. The collection of these label probability vectors $F = \{\mathbf{p}(\theta_1), \mathbf{p}(\theta_2), \dots, \mathbf{p}(\theta_n)\}$ represents the label probability distribution over the entire point-set. Our ultimate aim is to locate correspondences between two point-sets on the basis of the above-mentioned label probability information and the theory of graph spectra.

The idea underpinning spectral correspondence matching methods is to embed feature point-sets into a common eigenspace, and to find feature correspondences by performing alignment in this space. The key idea is that of finding the

appropriate function which captures the essential properties of the given data-set which should also be robust under uncertainties such as outliers, random position jitter, occlusion, etc., and be able to identify the common eigenspace. Also the captured properties should be common in both data-sets. The problem of how to select the best kernel function is a topic that has recently attracted considerable interest in kernel learning theory. The development of kernel PCA [109] provides us with a theoretically sound way of improving the existing spectral point pattern matching algorithms since it shares many features in common with spectral graph theory.

When the objects being tracked are not rigid, more considerations are required. In this chapter we intend to take advantage of the semantic relationships between different rigid components. The idea is that by utilizing the consistency constraints that are contained between rigid components, more effective matching results can be obtained. To do this we draw ideas from probabilistic relaxation labelling [106, 80, 95]. We characterize each point by augmenting the positional information with a vector of label probabilities. In addition, the arrangement of the points is represented using a Gaussian point proximity matrix. First we show how the point proximity matrix can be incorporated into the definition of the support function for relaxation labelling. In this way when the label probabilities are updated, then the strengths of the proximity relation are brought to bear on the computation of label support. Next we show how the label probabilities can be used to refine the point correspondence process. Here a kernelized version of the Shapiro and Brady algorithm is used. The label probabilities are used to refine the kernel matrix which will be used to locate point correspondences. The matching process is realized in an iterative fashion where there are interleaved steps for label probability update and for point correspondence matching. In both of the rigid and articulated cases, we focus in detail on Gaussian and polynomial kernels because of their transformation invariant properties.

3.2 Graph spectral matching

Graph spectral methods for point pattern matching solve the point correspondence problem by first constructing a graph representation for each data-set. Each graph node corresponds to an image feature point. Each edge corresponds to a spatial relationship between two feature points at the end of the edge. After graph construction, we represent each point-set by a matrix on the basis of the graph. We then find feature correspondences using matrix eigendecompositions. These methods aim to embed the similarity (or dissimilarity) properties of the original data into a common space in which correspondence matching can be performed. For the case of matching feature points undergoing rigid motion, the two essential ingredients are the similarity function and the embedding procedure. Usually the similarity properties are regarded as weights of the edges, and are expressed in the form of a proximity matrix W . In the literature both the adjacency matrix and the Laplacian matrix have been used as the proximity matrix for this problem. The elements W_{ij} of the edge weight matrix W represents the similarity relationship between feature points \mathbf{x}_i and \mathbf{x}_j and various similarity or dissimilarity measurement functions can be used.

Our aim in constructing the matrix representations for the point-sets is to provide a basis for the correspondence process. As we are dealing with objects subject to transformations such as translation, rotation, scaling, and reflection as well as small deformation, it is desirable for the similarity function to be invariant under these transformations. It is known from geometry that the Euclidean distance is invariant to the similarity transformation. Hence, similarity functions underpinning many existing methods are related to the Euclidean distance between feature points. One example is the pervasively used Gaussian function which has also been used in [114, 112]:

$$W_{ij} = e^{-d_{ij}^2/\sigma} \quad (3.1)$$

where d_{ij} is the Euclidean distance between the two feature points \mathbf{x}_i and \mathbf{x}_j ,

and σ is a constant parameter. Another interesting example of the invariance of the rigid transformation, is the directional properties of the feature points. This property can also be considered as a good candidate for constructing a suitable similarity function for spectral point pattern matching.

In this chapter we are interested in using the label information in the weighted adjacency matrix for non-rigid motion. That is, we propose to use the label probability values for the problem of articulated feature correspondence matching. The label of each feature point is also used to define the neighbourhoodship of it with other points when building the matrix representation.

3.3 Kernel spectral methods

When viewed from the perspective of kernel principal components analysis (kernel PCA, [109]), applying a dissimilarity or similarity function to the original data set is equivalent to the process of using a kernel function to map the data into a higher, possibly infinite, dimensional space. Moreover, this mapping interpolates the data in the new space according to their transformation invariant properties when an appropriate kernel function is applied. From this perspective, kernel PCA appears to provide us with a sound theoretical basis for spectral pattern matching.

Kernel PCA [109] can be regarded as a generalization of conventional principal components analysis (PCA) from a linear to a non-linear transformation. In the literature, it has been shown to provide a better way of recovering the underlying principal components of the given data, e.g., the de-noising application in [88]. Conventional PCA provides an orthogonal transformation of the data from a high dimensional space to a low dimensional one which maximally preserves the variance of the original data. This is done by first computing the eigenvalues

and eigenvectors of the covariance matrix

$$\mathbf{C} = \frac{1}{M-1} \sum_{i=1}^M (\mathbf{x}_i - \bar{\mathbf{x}})(\mathbf{x}_i - \bar{\mathbf{x}})^T,$$

where M is the dimension of the given data-set. Then the first N normalized eigenvectors (where $N \leq M$ and the eigenvalues are sorted in descending magnitude order) of the covariance matrix are used as the principal projection axes for the data. Since the method minimizes the residual covariance of the data points projected into the common eigen-subspace, it thus gives an optimum representation of the original data in the chosen projection space.

The main difference between kernel PCA and conventional PCA is that kernel PCA first uses a function $\mathcal{T} : \mathbf{x} \mapsto \Phi(\mathbf{x})$ to map the data from the low dimensional space into a new feature space \mathbb{F} of higher, possibly infinite, dimension. Conventional PCA is then performed on the transformed data matrix to obtain the data projection. This gives kernel PCA the property of extracting non-linear features from the data-set, and makes it a powerful tool in many pattern analysis applications. In practice, the mapping is performed implicitly by choosing a suitable kernel function $K(\mathbf{x}_i, \mathbf{x}_j)$ for the data points \mathbf{x}_i and \mathbf{x}_j . Given that the chosen function $K(\mathbf{x}_i, \mathbf{x}_j)$ is continuous and symmetric, and also satisfies the positive semidefinite condition

$$\int_{X \times X} K(\mathbf{x}_i, \mathbf{x}_j) f(\mathbf{x}_i) f(\mathbf{x}_j) d\mathbf{x}_i d\mathbf{x}_j \geq 0, \quad (3.2)$$

it is ensured that $k(\mathbf{x}_i, \mathbf{x}_j)$ is a valid kernel for some feature space. This provides a flexible way of choosing kernel mapping functions.

To extract the principal components of the mapped data, first a covariance matrix needs to be constructed for the data in the feature space \mathbb{F} . Suppose that the data $\{\mathbf{x}_1, \dots, \mathbf{x}_m\}$ in space \mathbb{F} are centred, then the covariance matrix of the mapped data in this space is:

$$\overline{\mathbf{C}} = \frac{1}{m-1} \sum_{i=1}^m \Phi(\mathbf{x}_i) \Phi(\mathbf{x}_i)^T.$$

Then the eigenvalues and eigenvectors of $\bar{\mathbf{C}}$ are to be computed by the following equation:

$$\bar{\mathbf{C}}\mathbf{v} = \lambda\mathbf{v}. \quad (3.3)$$

Since the explicit mapping \mathcal{T} is possibly unknown, computing the covariance matrix directly is not feasible. Schölkopf, Smola, and Müller showed in [109] that by solving the eigen-equation $m\lambda\mathbf{u} = K\mathbf{u}$ in which K is the Gram matrix computed from the kernel function $K(\mathbf{x}_i, \mathbf{x}_j)$, \mathbf{u} is one of its eigenvectors, and $m\lambda$ is its corresponding eigenvalue, the p_{th} principal component takes the form

$$\langle \mathbf{v}_p, \Phi(\mathbf{x}) \rangle = \frac{1}{\sqrt{\lambda_p}} \sum_{i=1}^m \mathbf{u}_{p,i} K(\mathbf{x}_i, \mathbf{x}),$$

which can be further simplified to ([56])

$$\langle \mathbf{v}_p, \Phi(\mathbf{x}) \rangle = \frac{1}{\sqrt{\lambda_p}} (K\mathbf{u}_p)_n = \sqrt{\lambda_p} \mathbf{u}_{p,n}. \quad (3.4)$$

Here \mathbf{v}_p is the p_{th} eigenvector of $\bar{\mathbf{C}}$.

To generalize the method to non-centred data, the kernel matrix K becomes [109, 56] $K' = (I - ee^T)K(I - ee^T)$ where $e = m^{-1/2}(1, 1, \dots, 1)^T$. In the case when more than one rigid component is present in the data point-set, the data need to be centered onto their respective subpart centre of movement. Thus, the mean value of each data group in the feature space \mathbb{F} needs to be computed and subtracted from $\Phi(\mathbf{x}_i)$ in the covariance-matrix above. We make this separation using the label probabilities. For the group with label λ , the mean-position (i.e. subgroup centre) is given by

$$\mu_\lambda = \frac{1}{\sum_i P(\theta_i = \lambda)} \sum_i \Phi(\mathbf{x}_i) P(\theta_i = \lambda), \quad \text{for each } \lambda \in \Omega.$$

Let $\tilde{\Phi}(\mathbf{x}_i) = (\Phi(\mathbf{x}_i) - \sum_\lambda \mu_\lambda P(\theta_i = \lambda))$, then the covariance matrix of centered data is then given by

$$\tilde{\mathbf{C}} = \frac{1}{m-1} \sum_{i=1}^m \tilde{\Phi}(\mathbf{x}_i) \cdot \tilde{\Phi}(\mathbf{x}_i)^T, \quad (3.5)$$

where

$$\begin{aligned}
\tilde{\Phi}(\mathbf{x}_i) \cdot \tilde{\Phi}(\mathbf{x}_i)^T &= K(\mathbf{x}_i, \mathbf{x}_i) \\
&- \sum_{\lambda \in \Omega} \frac{P(\theta_i=\lambda)}{\sum_j P(\theta_j=\lambda)} \sum_j P(\theta_j = \lambda) K(\mathbf{x}_i, \mathbf{x}_j) \\
&- \sum_{\lambda \in \Omega} \frac{P(\theta_i=\lambda)}{\sum_k P(\theta_k=\lambda)} \sum_k P(\theta_k = \lambda) K(\mathbf{x}_k, \mathbf{x}_i) \\
&+ \sum_{\lambda \in \Omega} \frac{P^2(\theta_i=\lambda)}{\sum_j P(\theta_j=\lambda) \sum_k P(\theta_k=\lambda)} \sum_j \sum_k P(\theta_j = \lambda) P(\theta_k = \lambda) K(\mathbf{x}_k, \mathbf{x}_j)
\end{aligned} \tag{3.6}$$

and $K(\mathbf{x}_i, \mathbf{x}_j)$ are the entries of the above kernel matrix K .

Based on their transformational invariants, two kernel functions, the Gaussian kernel and the polynomial kernel, are of interest in this work. The Gaussian kernel has the form defined in Equation (3.1). Since it is based on the Euclidean distance between two feature points, it is invariant to the similarity transformation. The polynomial kernel has the form

$$K(\mathbf{x}_i, \mathbf{x}_j) = (\mathbf{x}_i \cdot \mathbf{x}_j + c)^d, \tag{3.7}$$

where c and d are constants ($d \neq 0$). This kernel captures the directionality of the data which is a useful property for correspondence matching. However, the scalar or dot product is not invariant under scaling, and so there is still a magnitude problem to be considered. To solve this problem, one way is to normalize the scaled and truncated eigenvectors. Another method is to scale both of the two eigenvector matrices by the eigenvalue matrix of the model data-set. This is based on the following matrix property. Suppose that two eigen-decomposable symmetric matrices A and B , are related by the scalar multiplier s , i.e. $A = s \cdot B$. Let the eigen-decompositions of the two matrices be $A = U_A \cdot D_A \cdot U_A^T$ and $B = U_B \cdot D_B \cdot U_B^T$, where U_A and U_B form the orthonormal bases for the two matrices. When these conditions are satisfied then $D_A = s \cdot D_B$. In this work, we use this property to overcome the scale problem.

3.4 Label Process

Relaxation labelling is one of the most extensively studied approaches to the consistent labelling problem in computer vision. For a given object-set X , and label

set Ω_i for each object $x_i \in X$, the task is to assign a consistent and unambiguous label to each object in X . The process of relaxation labelling can either be an “offline” belief propagation one that distributes the previously learned labelling confidence over the entire feature set, e.g., [95], or an “online” learning process that learns the labelling information on the fly, e.g., [106, 80]. In discrete relaxation process (e.g., [130]), initially each node is assigned with all possible labels. During the iterative relaxation procedure, inconsistent labels are discarded until a final consistent label distribution is obtained. In the continuous or probabilistic case, each node is assigned an initial weight or probability distribution. Iteratively, the label probabilities or weights are updated, again until a consistent distribution is reached. However, whichever labelling process is used, the performance depends critically on the compatibility coefficients adopted and the support function used to combine evidence in the iterative process. The compatibility values represent our prior knowledge concerning the constraints between different labels. In [80], a dictionary is used. A more interesting example is given in [95] where the compatibility coefficients are represented as a vector which is learned offline. These definitions are not suitable for our point-correspondence problem. However, our compatibility model does share some properties in common with the compatibility vector in [95].

The labelling process that is being developed here is an evidence combining one that propagates label constraints. In our developments for feature correspondence matching, we incorporate label information into the kernel function to represent constraints on articulated motion for the feature correspondence matching problem. Our label consistency model is derived from one of the feature point-sets, which we refer to as the model. We hence learn the label compatibility information from the model point-set, before attempting to match it against the data point-set. Thus the first step is to collect label information and learn the compatibility values of each label pair from the model point-set. Then the learned label compatibility model is applied in the second step of the process which involves

assigning consistent point labels to the “data” point-set.

For simplicity, in this chapter we assume that the label sets Ω_k are identical for the different objects, as is the case of most relaxation labelling applications, e.g., [80]. We denote this universal label-set by $\Omega = \{\omega_i\}_{i=1}^l$ where l is the number of distinct labels. The compatibility between each label pair, that is, the contextual information, is represented using compatibility coefficients or functions in the form of a matrix, $R = \{R_{ij}(\omega_i, \omega_j)\}$, where $R_{ij}(\omega_i, \omega_j)$ represents the compatibility between the node x_i being assigned a label ω_i and the node x_j being assigned a label ω_j . We also assume spatial homogeneity, that is, the entries $R_{ij}(\omega_i, \omega_j)$ are invariant to object location [43]. For brevity, we denote the entries as $R(\omega_i, \omega_j)$.

The compatibility matrix $R \in \mathbb{R}^{l \times l}$ is of dimension $l \times l$, and embodies knowledge of the number of rigid components, i.e. labels, in each image, and the semantic constraints between each pair of object-labels. In the following development in this chapter, the matrix has elements

$$R(\omega_i, \omega_j) = \begin{cases} 1 & \text{if } \mathbf{x}_i \text{ and } \mathbf{x}_j \text{ come from the same rigid part;} \\ -1 & \text{otherwise.} \end{cases}$$

This definition restricts the nodes to give total positive support to the nodes in the same group and to contribute a negative support to nodes outside the group. The proximity constraint is also acquired from the model image. We assume that in any two consecutive image frames, the relative position of the rigid components of the object under study will not change significantly.

With these ingredients the next step is to compute the support from the neighbourhood for the label assignment θ_i to point \mathbf{x}_i . Let us denote the neighbourhood system for a point \mathbf{x}_i by $\mathcal{N}_i = \{\mathbf{x}_{i_1}, \dots, \mathbf{x}_{i_k}\}$, where k is the number of each node’s nearest neighbours. Here we use the Euclidean distance between each point pair \mathbf{x}_i and \mathbf{x}_j to define the neighbourhood. We further weight the support that each object \mathbf{x}_i can obtain from its neighbours \mathcal{N}_i using a similarity measure between \mathbf{x}_i and each neighbour. This choice comes from the intuition that closer neighbours should contribute more support. Unlike the traditional sum-product

support (e.g., [71]), the support values are exponentiated and normalized further. In this way, our computing of the support is in the spirit to the technique of probabilistic nearest neighbours introduced by Holmes and Adams in [68]. The new support function is as follow:

$$S_{i,\omega_i} = \frac{\exp\{\sum_{j \in \mathcal{N}_i} \sum_{\omega_j \in \Omega} P(\theta_j = \omega_j) R(\omega_i, \omega_j) W_{ij}\}}{\sum_{\omega_i \in \Omega} \exp\{\sum_{k \in \mathcal{N}_i} \sum_{\omega_k \in \Omega} P(\theta_k = \omega_k) R(\omega_i, \omega_k) W_{ik}\}}, \quad (3.8)$$

where $R(\omega_i, \omega_j)$ are the elements of the label compatibility matrix R which measure the compatibility of the label pair ω_i and ω_j . Here the elements of the proximity matrix W are defined using Equation (3.1) and are used to weight the label-support.

With the label compatibility information learned from the model point-set and the support values computed, the label probabilities can be updated for each point using the formula:

$$P^{(n+1)}(\theta_i = \omega_i) = \frac{P^{(n)}(\theta_i = \omega_i) + \alpha S_{i,\omega_i}^{(n)}}{\sum_{\omega_j \in \Omega} (P^{(n)}(\theta_i = \omega_j) + \alpha S_{i,\omega_j}^{(n)})}, \quad (3.9)$$

where α is a constant parameter and n is the iteration index.

3.5 Matching

In this section, we describe our point matching algorithm and detail how it uses kernel PCA and spectral graph theory. We develop two different algorithms. The first of these is designed to work with single objects undergoing rigid motion, while the second is designed to work with objects which have rigid components that undergo articulated motion relative to one another. We commence by developing an efficient point pattern matching process for feature points under rigid motion. Here we require a kernel function that captures the transformation invariants of the object movement, and allow the feature points to be embedded into a lower dimensional feature space in a manner that provides a basis for one-to-one correspondence matching. The contribution is to first develop a point pattern matching method that can be applied to feature point-sets from objects that

undergo rigid motion, and then extend the algorithm for locating feature correspondences for feature points undergo articulated movement. The extension is accomplished by incorporating label consistency constraints into the rigid matching process.

3.5.1 Rigid Case

The idea underpinning the use of kernel spectral methods is to first represent the transformation invariant relationships between the feature points in terms of a proximity matrix for each feature point-set. Then correspondences are located by using the eigen-decompositions of the matrix pairs. That is, we first choose an appropriate kernel function to extract the transformation invariant feature properties from the point-sets. By performing kernel PCA on the given data-sets, we extract a transformation invariant basis set for the feature points. For the feature points from rigid motion, the Gaussian and polynomial kernel functions satisfy our requirements. The procedure for performing rigid matching by kernel PCA is described in the pseudo-code given in Table 3.1.

This matching algorithm is motivated by the approach described by Shapiro and Brady [114]. In [114], a proximity matrix W is first constructed for each image with the matrix elements given by $W_{ij} = \exp\{-d_{ij}^2/2\sigma^2\}$, where d_{ij}^2 is the Euclidean distance between points x_i and x_j , and σ is a constant. Shapiro and Brady explain this as the mapping of the original two dimensional data to a higher dimensional space, and thus capture the structural arrangement of the feature points. They then perform eigen-decomposition on matrix W to obtain its eigenvalues and eigenvectors. For each point set, a new modal matrix is constructed with the eigenvectors sorted in descending eigenvalue order as its columns. The rows of the matrix are then considered as the projections of the feature points into the eigenspace. When the data sets are of different size, only the first k leading eigenvectors from each data sets are used where k is the size of the smaller data set. To make the algorithm more robust, Shapiro and Brady also suggest

Table 3.1: Kernel spectral matching, Algorithm I

1. Compute the proximity matrices:

$$W_{ij}^1 = K(\mathbf{y}_i, \mathbf{y}_j),$$

$$W_{ij}^2 = K(\mathbf{x}_i, \mathbf{x}_j);$$

2. Centre the obtained proximity matrices:

$$W^1 = (I - ee^T)W^1(I - ee^T)$$

$$W^2 = (I - ee^T)W^2(I - ee^T)$$

3. Eigen-decomposition:

$$W^1 = U^1 \Lambda^1 (U^1)^T;$$

$$W^2 = U^2 \Lambda^2 (U^2)^T;$$

4. Compute the projections:

$$\mathbf{y}' = U^1 \sqrt{\Lambda^1}$$

$$\mathbf{x}' = U^2 \sqrt{\Lambda^2}$$

5. Correspondence localization: $\mathbf{x}'_i = \min_{\mathbf{y}_j, j=1, \dots, n} \text{dist}(\mathbf{x}'_i, \mathbf{y}'_j)$.

that the eigenvalues may be considered to scale the corresponding eigenvectors. However, no further discussion is given. Instead, they focus on placing more emphasis on the more significant eigenvectors. When the eigenvalues are involved in terms of scaling the corresponding eigenvectors, this approach is similar to kernel PCA. When compared with the kernel PCA method described above, it is clear that the Shapiro and Brady method is a special case where the data in the mapped space has a mean zero and uses the Gaussian as the kernel function.

3.5.2 Articulated Case

In the case of matching feature point-sets resulting from articulated motion, the above matching method can not be used directly since the motion of each individual rigid component changes the relative positions of the feature points from the different components. Thus, the overall structural change is large. Our idea is to use label consistency constraints to construct a modified proximity matrix. In particular, we propose to use the weighted adjacency matrix \widetilde{W} for each point-set in which each element of the matrix is computed using the formula:

$$\widetilde{W}_{ij} = \sum_{\omega_k \in \Omega} P(\theta_i = \omega_k) P(\theta_j = \omega_k) W_{ij} \quad (3.10)$$

where W_{ij} is defined in Equation (3.1). However, in order for \widetilde{W} to be a valid kernel matrix, it must satisfy the conditions that it is symmetric and positive semidefinite. Since W_{ij} is symmetric, and each such element is re-weighted by both the label probabilities of node \mathbf{x}_i and node \mathbf{x}_j , the symmetric property is preserved. We know that a matrix is positive semidefinite if all its eigenvalues are non-negative. Although at the moment it is not clear of a proof of the matrix's semi-definiteness, by computing its eigen-decomposition we found that all its eigenvalues are non-negative. Thus we can say that \widetilde{W} is a valid kernel. This matrix is then subjected to the kernelisation procedure outlined in Equation (3.6) and its eigen-decomposition computed. The mapping of the feature vectors $\tilde{\mathbf{y}}_j$ and $\tilde{\mathbf{x}}_i$ are thus computed by using Equation (3.4) for the respective modal and

data point-sets; that is,

$$m\lambda\mathbf{u} = \widetilde{W}\mathbf{u}, \quad \tilde{\mathbf{x}}_{i,n} = \sqrt{\lambda_i}\mathbf{u}_{i,n},$$

and a similar procedure is applied to $\tilde{\mathbf{y}}_j$. The next step is to compute the association matrix $M_{ij} = \exp(-d_{ij}^2/\sigma)$, where $d_{ij}^2 = \|\tilde{\mathbf{y}}_i - \tilde{\mathbf{x}}_j\|^2$ is the distance of the point pairs. Let us denote the label agreement of the point pair \mathbf{y}_i and \mathbf{x}_j by $P(\theta_j = \theta_i = \omega_k, \forall \omega_k \in \Omega)$. The association of the two feature vectors is further gated by this constraint:

$$\widetilde{M}_{ij} = P(\theta_j = \omega_k, \theta_i = \omega_k, \forall \omega_k \in \Omega)M_{ij}, \quad (3.11)$$

The correspondences are defined as the most similar node pairs. That is, for each node \mathbf{x}_i in the data point-set, the correspondence in the model set is the node \mathbf{y}_j that has the largest association \widetilde{M} . If we assume that the labels on each feature point are independent of one-other, the consistency of the label assigned to point \mathbf{x}_i and the label assigned to \mathbf{x}_j is given by:

$$P(\theta_i = \omega_k, \theta_j = \omega_k, \forall \omega_k \in \Omega) \equiv \sum_{\omega_k \in \Omega} P(\theta_i = \omega_k)P(\theta_j = \omega_k) \quad (3.12)$$

The matching process is an iterative one in which at each step new label probabilities are incorporated to improve matching. Since as an increasing number of correspondences are found, the value of the quantity

$$E = \sum_i \exp(\|\mathbf{x}_i - \mathbf{y}_i\|_F / 2\sigma^2), \quad (3.13)$$

where \mathbf{x}_i and \mathbf{y}_i are the correspondence pair from data point-set and model point-set, respectively, will increase, and ultimately reach a maximum value. Thus we use this quantity as one of our stopping criteria for the matching process. The other is a predefined iteration number. The matching process is summarised in the pseudo-code listed in Table 3.2.

Table 3.2: Kernel spectral matching, Algorithm II

1. Initialize $\mathbf{p}_X, \mathbf{p}_Y^{old}$;
2. Initialize $t = \text{predefined-iteration-number}$, $E_{old} = 0$;
3. Learn the compatibility information R from X ;
4. Use \mathbf{p}_d^{old} to compute \tilde{C}_X, \tilde{C}_Y using Eq. (3.5) and Eq. (3.6), and compute the principal components for each data-set;
5. Compute \tilde{M} using the obtained principal components and Eq. (3.11);
6. Find for each $\mathbf{x}_i \in X$ its correspondence $\mathbf{y}_j = \max_j \tilde{M}_{ij}$;
7. Compute E using Eq. (3.13), set $diff = E - E_{old}$;
8. If ($diff < \text{threshold}$) or ($\text{iteration} > t$) return;
9. Run the labelling process, compute \mathbf{p}_Y^{new} using the matching results;
10. Update $\mathbf{p}_Y^{old} = \mathbf{p}_Y^{new}$;
11. Go to step 3.

3.6 Experiments

In this section we present experimental evaluation of both the rigid and articulated matching methods. Our experiments are performed with both synthetic and real world data. We also compare the proposed algorithms with algorithms of Shapiro and Brady [114], Scott and Longuet-Higgins [112], and embedded point-set matching using the multidimensional scaling. MDS is also a method widely used for data dimensionality reduction, and is also based on eigenvalues and eigenvectors of a dissimilarity matrix [33]. It attempts to preserve the pairwise relationships between the data points while mapping the data into a low dimensional space. The experiments are performed using the classical MDS in which the Euclidean distance is taken as the dissimilarity measure.

The experiments focus on the performance of the algorithms when the data are subjected to similarity transformations and, contain uncertainties such as outliers, random position jitter, and small deformations.

3.6.1 The data

The experiments are performed on both synthesized and real world data-sets. They are chosen to cover the possible uncertainties mentioned above and which are common in real cases. The experimental designs are as follows:

1. **Synthetic data:** Here we assume that the point sets are subject to a two dimensional affine transformation. Given a point-set $X = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n\}$ sampled from a rigid object, a synthetic data-set $X' = sTX + \mathbf{t}$ is generated for testing the algorithms, where s is a scaling parameter, \mathbf{t} is the translation vector, and $T = \begin{bmatrix} \cos \theta & -\sin \theta \\ \sin \theta & \cos \theta \end{bmatrix}$ is the 2D rotation matrix. In our experiments with Algorithm I (the rigid variant), the rotation angle $\theta = \frac{10^\circ}{180} \times \pi$. In the experiments for Algorithm II (the articulated variant), the transformation parameters are set as $s = 0.8$, $\mathbf{t} = (10, 15)^T$, and $\theta = (20^\circ/180)\pi$, and in

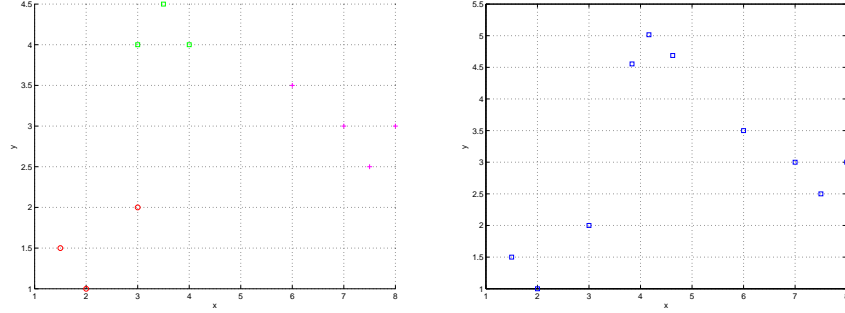


Figure 3.1: Synthetic data, data-set 6 in Table 3.4. *Left: the original generated data; Right: after rigid transformation.*

the third component, $s = 1.2$, $\mathbf{t} = (10, 15)^T$, and $\theta = (30^\circ/180)\pi$.

2. **Real data:** Here we use two sequences for Algorithms I. The first of these is a sequence of infra-red images of a hand shown in Figure 3.2 which is used as an example where the small geometric deformations are present. The second sequence is the CMU house sequence ([21]) shown in Figure 3.3, which is used to study the effects of matching point-sets of different sizes and where there is significant positional jitter. The image pairs in Figure 3.13 are used to experiment with Algorithm II on scenes where there is articulated object movement. The top image shows two rectangular objects that move relative to each other on the ground-plane. The bottom image shows a pair of spectacles, where one of the limbs moves.
3. **Noisy data:** Here Gaussian noise is added to the data set to test the robustness of the algorithm. The data is synthesized in the following way. First point position jitter is synthesized by generating a matrix $A \sim N(\mu, \Sigma)$ whose elements are Gaussian random variables with mean μ and covariance matrix Σ . The point jitter is added to the matrix of feature point-set positions for the second point set X_2 using the equation $X_2 = X_2 + A$.
4. **Data sets with different sizes:** To simulate structural errors we delete a controlled fraction of the feature points from the data point-set. This is done

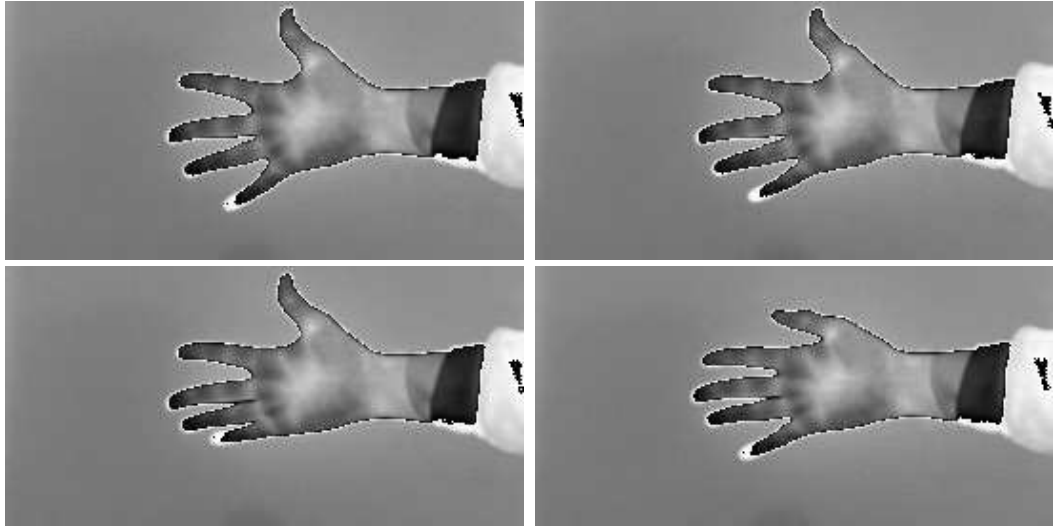


Figure 3.2: The hand image data (From left to right, top to bottom: frames 08, 09, 11, 25).

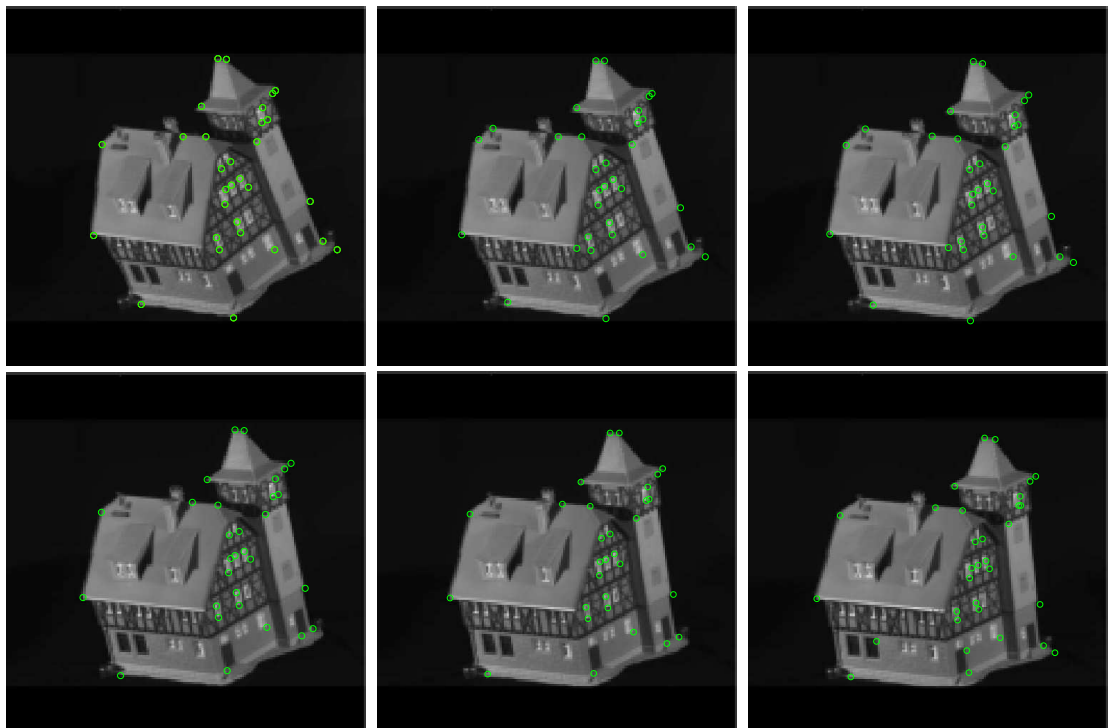


Figure 3.3: The CMU house data (From left to right, top to bottom: frames 01,02,03,04,10).

in two different ways. Firstly q consecutive points, where q starts from 1, to the integer number most close to 12% of the data-set size, are deleted to simulate occlusion. Secondly, q points are deleted from random locations to simulate the effect of segmentation errors. For the CMU house sequence and the glasses sequence, the feature points are extracted using a corner detector which produces errors and hence the point-sets are of different sizes. For instance, in frames 01, 02, 03, 04, 05, and 06 of the CMU sequence displayed in Figure 3.3, the sizes of the point-sets are 30, 32, 32, 30, 30, and 32, respectively.

3.6.2 Results

To compare the performance of the kernel approaches when deformations are present, experiments are performed on synthetically generated data where a 2D translation, rotation and isoscaling are added. The effect of missing points and random point position jitter in terms of the 2-D Gaussian random matrices with different covariance matrices as described above are also tested.

The σ value

When using the Gaussian kernel, the choice of the σ value significantly affects the performance of the algorithm when the data points contain significant uncertainties. We commence by investigating the effect of varying the parameter. Intuitively, we expect the value of σ to be strongly dependent on the pairwise distances between the feature points. Here we use the formula

$$\sigma = N\bar{d}_{ij}^2, \quad \bar{d}_{ij}^2 = \sum_{i,j} d_{ij}^2 / (n^2 - n)$$

to estimate the parameter, where N is a scalar parameter used to control the value of σ , d_{ij} is the Euclidean distance between the points \mathbf{x}_i and \mathbf{x}_j , and n is the size of the feature point-set. For four different feature point-set pairs, in Figure 3.4 we show the effect of varying the parameter σ on the percentage of correctly

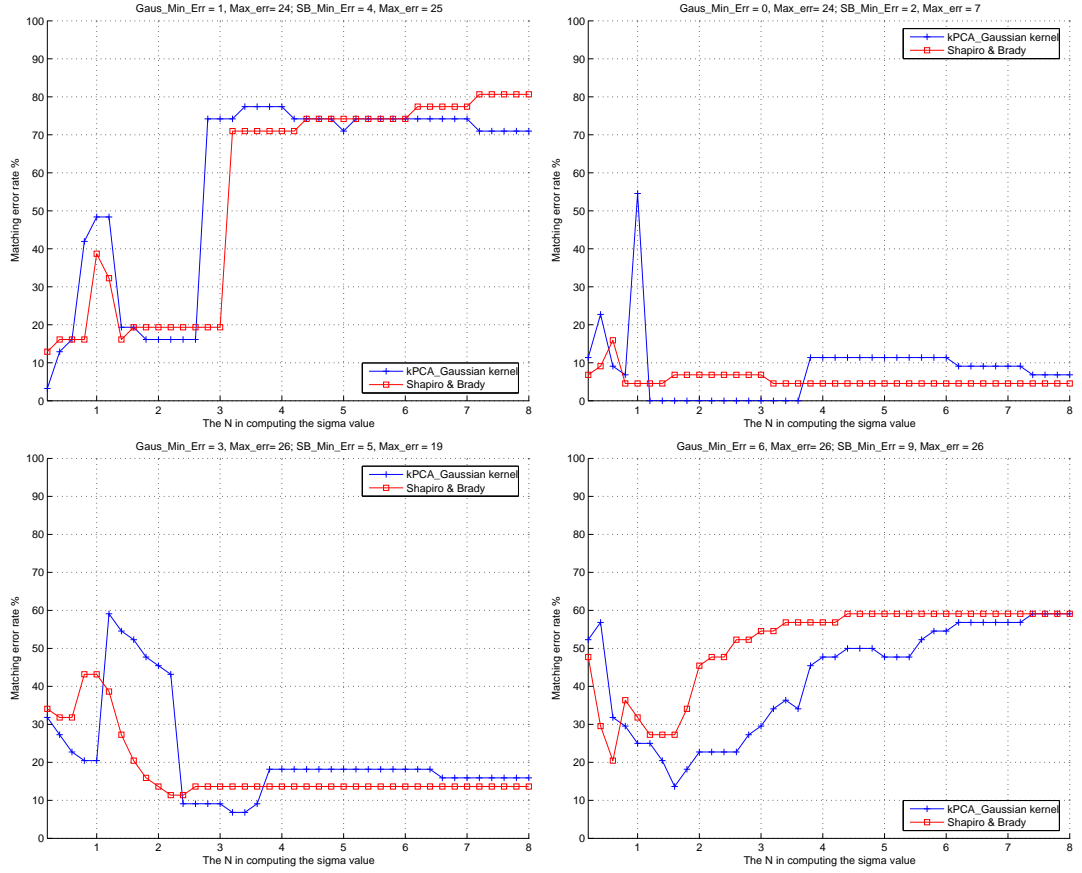


Figure 3.4: Effects of different σ value (From left to right, top to bottom: data-set 3, hand 08/09, hand 08/11, hand 08/25).

matched points. In these experiments, the value of N is set to start from 0.2 to 8, and is increased by 0.2 in each experiment. The curve marked with crosses is for the kernel PCA method with a Gaussian kernel and the curve marked with circles is for the Shapiro and Brady method. In all four cases, the best performance of the kernel PCA method is better than that delivered by Shapiro and Brady.

Label update process

We now turn our attention to experiments which focus on the performance of the label process for both synthetic and real image point-sets. The initial values of the label probabilities have an effect on the number of iterations and the rate

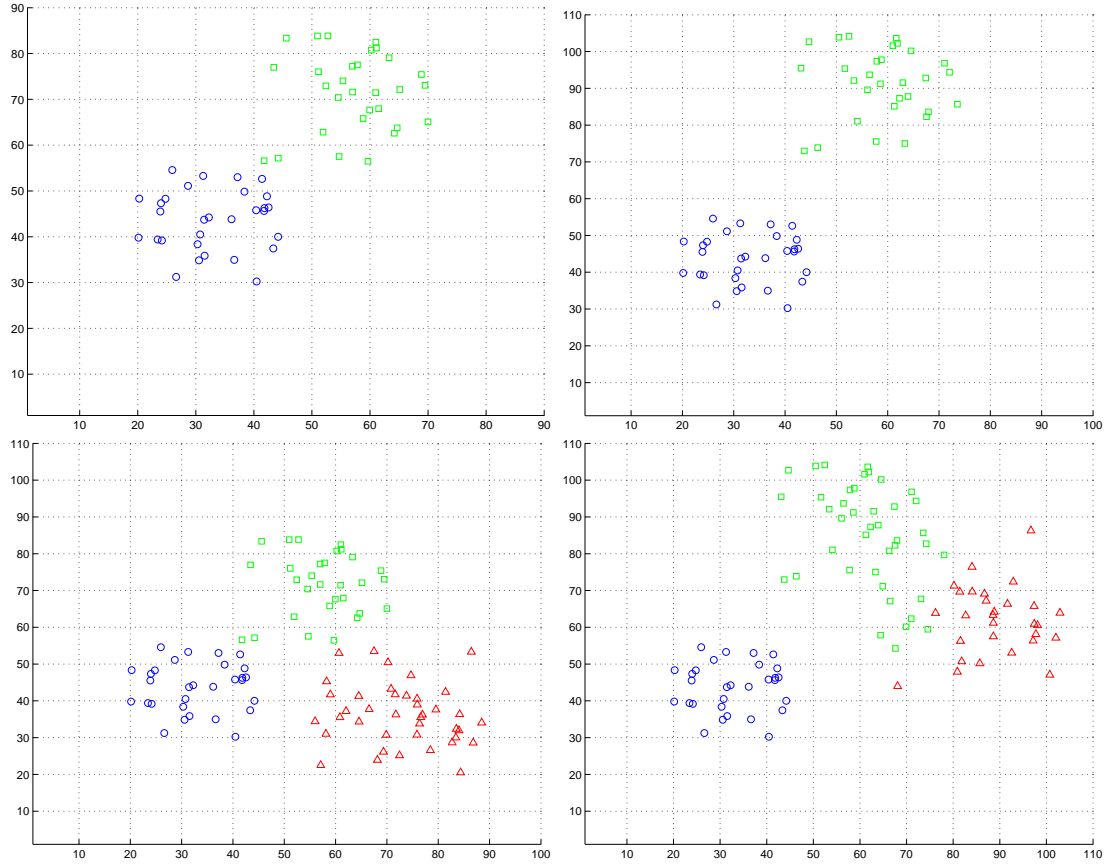


Figure 3.5: Synthetic data with 2 and 3 rigid components, and their labeling results. *Top: data-set pair 2 in Table 3.4; Bottom: data-set pair 1 in Table 3.4.*

of convergence of the method. Figure 3.5 shows the result of labelling random point-sets. The initial label probabilities were assigned uniformly in this instance. The different symbols correspond to the labels assigned to the points. As the clusters of points become more overlapped, then the number of labelling errors increase.

Matching

We commence by considering the case of rigid correspondence. We first experiment on the single component synthetic data generated using the methods described above. For this pair of data-sets, our algorithm for rigid motion, Shapiro & Brady's method, and Scott & Longuet-Higgins's method all give correspon-

dences that are 100% correct, while the MDS approach results in a 5% error rate. The effect of missing points for rigid point matching are shown in Figure 3.6. The experimental results are obtained by consecutively deleting a predetermined number of points from the data point-set. We then experiment with this data-set on the problem of feature points with random position jitter. Results are shown in Figure 3.7. Here we compare the results of using the Gaussian and polynomial kernels on single label point-sets with increasing jitter. The experimental results are the averages of 100 runs for each covariance matrix used for generating the Gaussian random jitter. There is relatively little to distinguish their performance.

Matching results of applying the algorithms to real data sets with feature points from rigid objects are shown in Figures 3.8 and 3.9. Here we show the fraction of correspondence matching errors as a function of the number of missing points (performed in the same way as for the synthetic data). The different curves in the two figures are for the different algorithms. As the number of deleted points increases, then the best performance is obtained when kernel PCA and a polynomial kernel are used. The poorest results are obtained with kPCA and a Gaussian kernel in the experiments on synthetic data-sets. But in the experiments on real image data, the kPCA with a Gaussian kernel performs better than the approaches in [33, 112, 114]. In all of the experiments using Shapiro and Brady’s method, the eigenvalues are used to normalise the corresponding eigenvectors in order to improve the matching results. This gives results that are comparable to MDS. More results from experiments on other data-sets are shown in Table 3.3.

After experiments on feature point-sets with rigid motion, we turn to the problem of articulated feature correspondence matching. Figures 3.10 and 3.11 respectively show the fraction of correct matches as function of the fraction of random point deletions, and the fraction of points occluded. The former is experimented by randomly deleting a fraction of the feature points from the model point-set. Each point in the diagram is the average value of 100 runs. The lat-

ter is simulated by deleting a fraction of consecutive feature points from the data point-set (as described previously). The ability of the algorithm to cope with the problem of random position jitter is also experimented for articulated case. Figure 3.12 shows more experiments on instances of increasing random position jitter with three articulated data-set pairs. Random position jitter is simulated by adding randomly generated position errors sampled from a 2D Gaussian distribution to the data point-set (as in the rigid matching experiments). The different curves in the plots are for different numbers of components (labels). In each case as the noise increases, then so the error also increases. The numbers used in these plots are summarised in Table 3.4. We can see that the performance of the algorithm copes well with this uncertainty. Figure 3.13 shows two examples of using Algorithm II for matching feature points extracted from real image pairs.

From these experiments, it is clear that the kernel PCA approach gives encouraging results when compared with the approaches of Shapiro & Brady [114], Scott & Longuet-Higgins [112], and the MDS method. Moreover, the kernel method is less sensitive to noise than the alternatives.

The main computational overheads of the algorithms described are associated with constructing the kernel matrices, and computing their eigenvalues and eigenvectors. This may prove burdensome for very large data-sets. If the feature point-sets of a moderate size (e.g., less than 10^3 points) then computing the full eigensystem for a matrix of size 5000×5000 takes less than half a minute on a desktop PC with an AMD Athlon 2000 CPU and 256MB RAM. The computation of the kernel matrix can be performed with time complexity that is polynomial in the size of the point-sets. The number of iterations required is also an important consideration. Algorithm II usually requires only three iterations. For the rigid point correspondence matching problem, our algorithm is non-iterative.

3.7 Conclusions

In this chapter we have made two contributions. First, we have explored the use of kernel PCA with a polynomial kernel function and a Gaussian for finding correspondences between two feature point sets. The relationship of the methods with Shapiro and Brady's correspondence method [114] is established and discussed. Experimental results reveal that the method offers performance advantages over a number of alternative methods. Here the polynomial kernel proves to be the most stable for point-sets of different sizes, and even in worst cases it gives a tolerable error rate. The performance of our algorithm is also comparable to the approaches described in [22, 21, 83]. One weakness of the Gaussian kernel is the selecting of the width parameter σ . In [114], the value is chosen manually. In this chapter, we use a heuristic formula based on the inter-point pairwise Euclidean distance matrix to compute σ automatically. The kernel functions used in this chapter are possibly not the best functions for extracting invariant properties from the feature point-sets. Other kernels may be used in the matching process to improve the results.

Our second contribution has been to extend the kernel method to articulated point-sets. Here we show how label compatibility coefficients can be used to refine the computation of the kernelised proximity matrix. The method improves the correspondence process when there are different moving components of a scene.

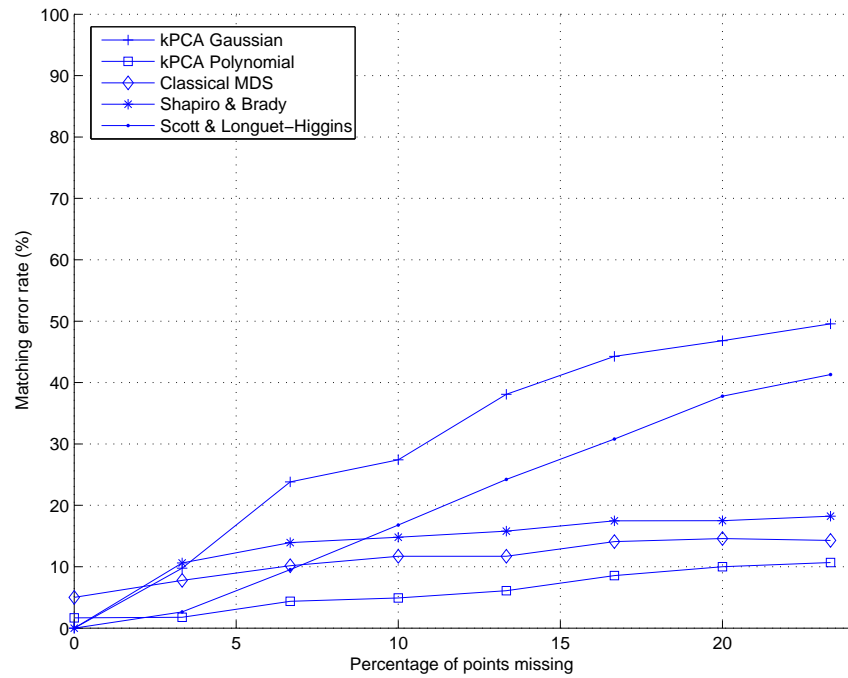


Figure 3.6: Matching results with synthetic data.

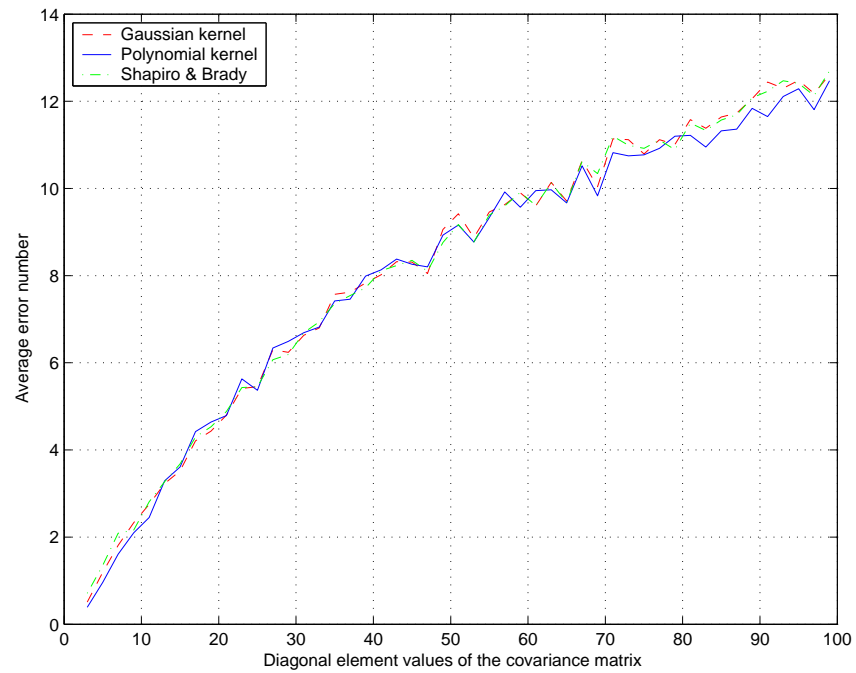


Figure 3.7: Effects of Gaussian random position jitter with a single label.

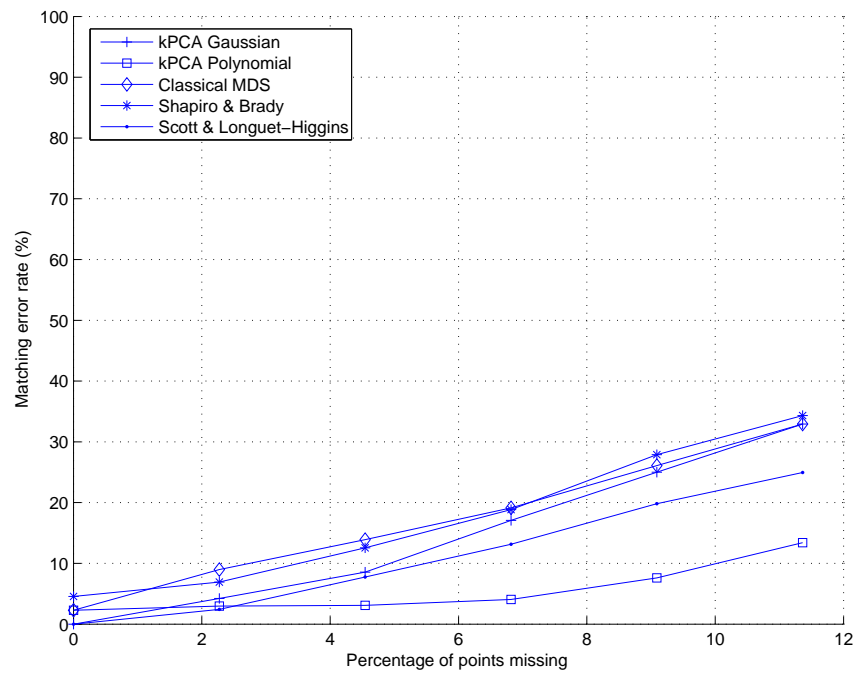


Figure 3.8: Matching results, hand 08 and 09, occlusions.

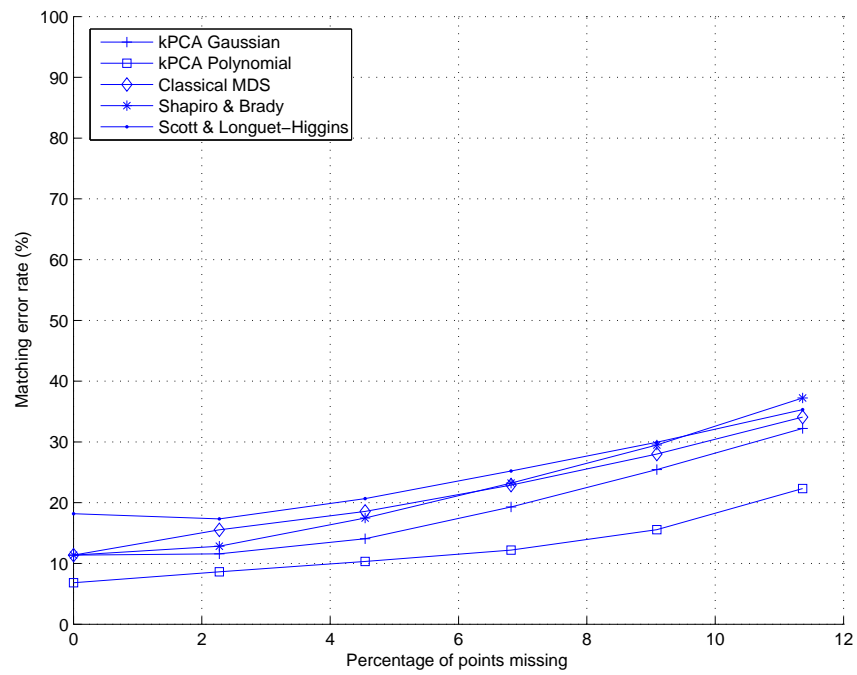


Figure 3.9: Matching results, hand 08 and 11, occlusions.

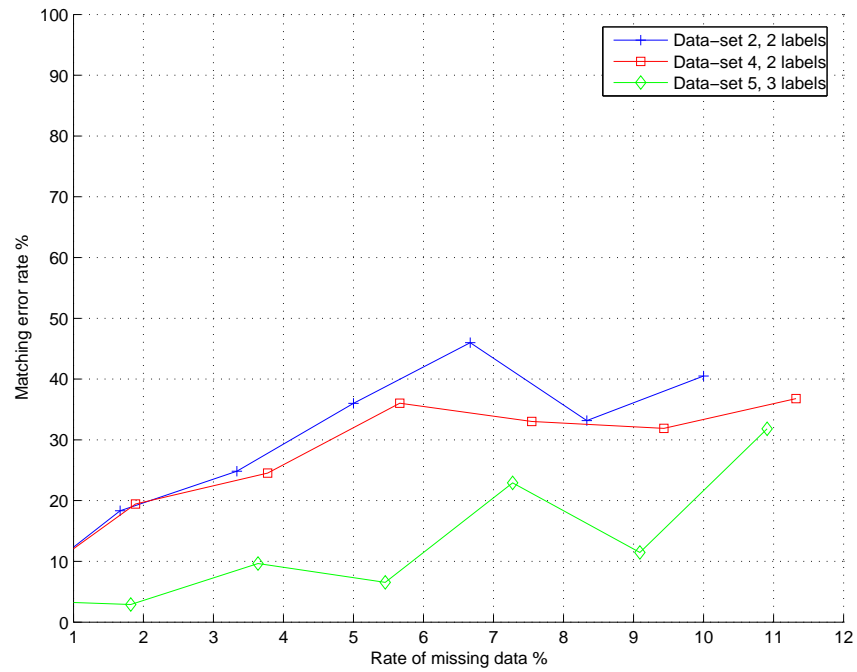


Figure 3.10: Effects of random point deletions, multi-label.

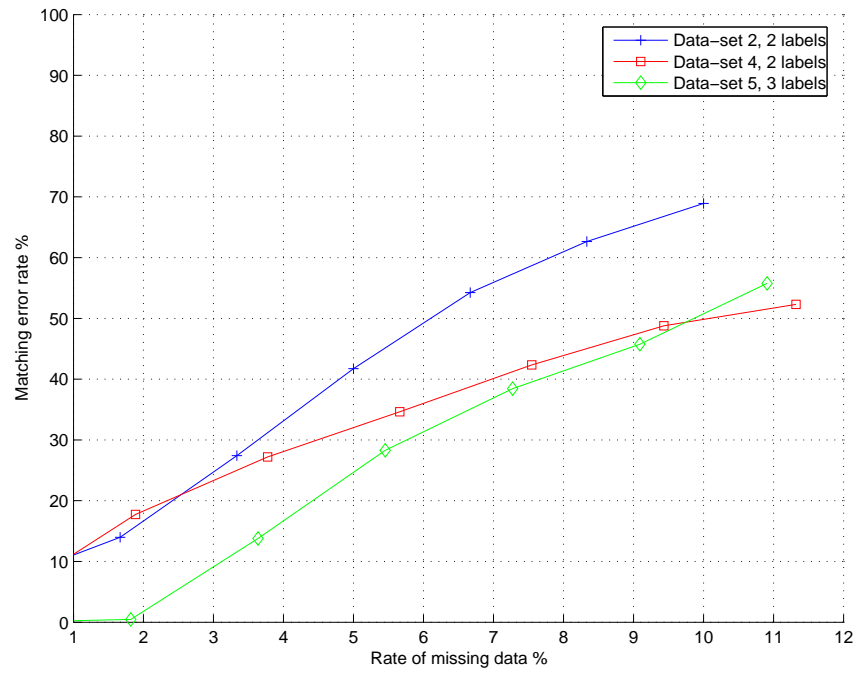


Figure 3.11: The effect of occlusions with multi-labels.

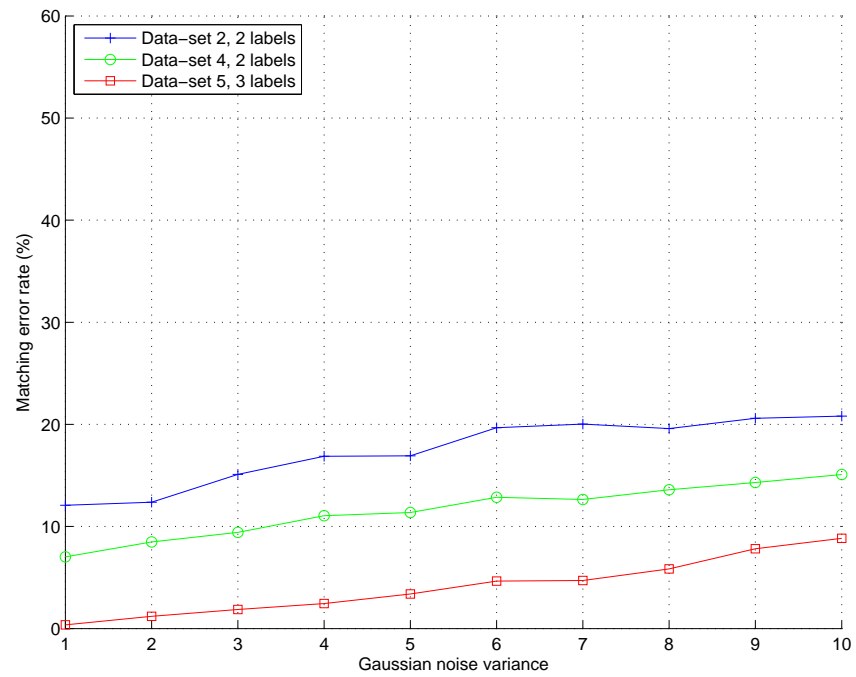


Figure 3.12: Effects of random point jitter, multi-label.

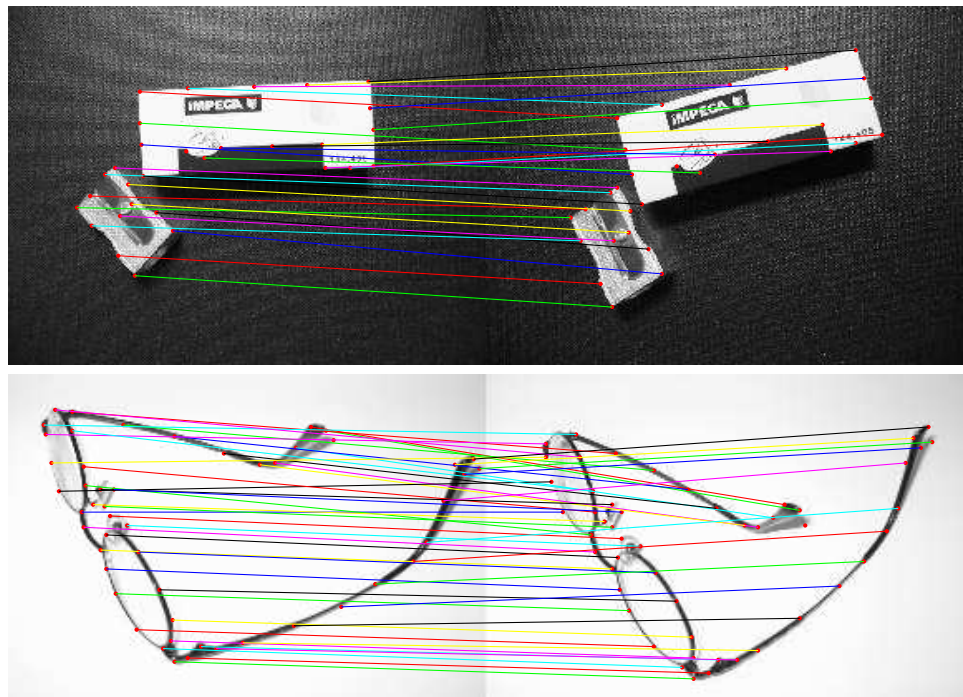


Figure 3.13: Articulated matching results. *Top: data-set 3 in Table 3.4; Bottom: data-set 5 in Table 3.4.*

Table 3.3: Matching results, Algorithm I (*Numbers of errors*).

	Hand data				CMU House				
Frames	08/25	09/11	09/25	11/25	01/02	01/03	01/04	01/05	01/06
KPCA,Gaussian	6	4	4	11	2	4	2	2	7
KPCA,Polynomial	5	7	6	12	4	5	3	5	13
MDS	35	5	26	27	5	5	25	25	28
Shapiro&Brady	9	6	8	17	3	5	2	2	9
SLH	4	3	5	10	7	6	3	7	9

Table 3.4: Matching and labeling results, Algorithm II (*error%*)

Data-set*	Num of points	Num of labels	No Label Information	Articulated matching(1)**	Articulated matching(2)***	Labeling process results
1	100	3	95	93	16	7
2	60	2	53.33	13.33	13.33	0
3	30	2	35.48	0	0	0
4	55	2	18.18	7.27	3.64	1.82
5	53	3	45.28	81.13	3.77	7.55
6	10	3	10	0	0	0
Note:	*: The data-sets are shown in Figures 3.1,3.5,and3.13; **: Results obtained based on the label information from the label process; ***: Results obtained when correct label information is assumed.					

Chapter 4

Probabilistic Relaxation Labelling by Diffusion

Motivated by the properties of diffusion processes and the properties of kernel methods, this chapter aims to develop a new formulation of probabilistic relaxation labelling for the purposes of feature labelling and correspondence matching. We use similar assumptions as classical relaxation labelling methods in our new development. That is, we assume that we have a set of objects (i.e., extracted image features or a set of data points), and we are given a set of all possible object labels together with some initial confidences of the object-label assignments. However, the pairwise compatibility relationships between the labels are not necessarily given *a priori*. The main contribution is that we first place the probabilistic relaxation labelling process into a graph setting, and use a diffusion equation to guide the processes of evidence combination and propagation. To do this, the kernel methods are used to construct a continuous time diffusion process on a graph. In this way we place the relaxation labelling in a kernel framework. Initial object-label probabilities then evolve across the graph according to an infinitesimal generator matrix constructed from the vertex and edge attributes of the graph. The newly developed algorithm is applied to data classification and feature correspondence matching problems. It is well-known that diffusion pro-

cesses have smoothing effects, and tend to stabilise on a trivial uniform distribution in the long run. Thus in our data classification case, when running for a short time period, local grouping features will emerge in the given data. For the feature correspondence matching case, we wish to sharpen the data, that is, the current label probabilities can be viewed as a blurred version of the original ones. Our objective is to de-blur the probabilities and recover the true pairwise cluster labels. This pairwise grouping is found by running the diffusion backwards in time. This technique has been used for image and signal enhancing, sharpening, and restoration over the last few decades [19, 48]. Although it is well known that the process of running the diffusion backwards in time is unstable, by applying certain constraints the process can be stabilized and interesting results are obtained. An early discussion of this issue can be found in the work by Carasso, Sanderson and Hyman [19] and references therein. Recent work of Gilboa, Sochen and Zeevi [48] also have a discussion on this problem. In addition to the development of the algorithm, its computational overheads are also discussed and several approximation algorithms are suggested. For both data classification and feature correspondence matching cases, experiments are performed on different data-sets and encouraging results are obtained.

4.1 The problem

The problem studied in this chapter can be described as follows. Suppose we are given a label-set which contains all possible labels for objects in a given object-set. Denote the label-set by $\Omega = \{\omega_k\}_{k=1}^l$, and the object-set by $X = \{\mathbf{x}_i\}_{i=1}^n$, respectively. Here l is the number of labels, and n is the number of objects to be labeled. Suppose also that some initial confidences of each object \mathbf{x}_i 's label assignments are given in terms of an object-label probability assignment vector, denoted by \mathbf{p}_i . In a probabilistic setting, this vector has the form $\mathbf{p}(\theta_i) = [P(\theta_i = \omega_1), \dots, P(\theta_i = \omega_l)]$ which represents our confidence of assigning a label $\omega_k \in$

$\Omega, k = 1, \dots, l$, to object \mathbf{x}_i . As in the previous chapter, the pairwise relations between each label pair, that is, the consistency constraints between the labels, are expressed using a label compatibility matrix. The objective in this chapter is then to assign each object in the given object-set a label which satisfies the consistency constraints produced by the label pairs, in an unambiguous and consistent way.

As pointed out by Hummel and Zucker in their work [71], relaxation labelling implicitly uses graphs as the representation for the object-set. Thus it is natural to combine relaxation labelling methods with graph theory for our problem. In addition, we also wish to model the local evolution of the label probabilities using the diffusion equation. Here we will use the diffusion equation to combine local evidence and to propagate the evidence globally with time. Given these conditions the basic setting is to construct a support graph on the basis of the given object- and label-sets. The relaxation of the label probabilities is then defined as a sequence of continuous time Markov diffusion processes on this support graph, under the action of a diffusion equation. The initial object-label probability assignment vector is thus evolved with time during the diffusion process. The final labels are obtained from the resulting probability distribution vector. The task of assigning labels to nodes becomes that of finding the desired state probability distribution of the corresponding continuous time Markov process on the support graph.

4.2 Relaxation Labelling

Suppose the object-set $X = \{\mathbf{x}_i\}_{i=1}^n$ are given with some structural information, as shown in the example in Figure 4.1. As mentioned in the previous chapter, the task of relaxation labelling is to assign a consistent and unambiguous label $\omega_{i_k} \in \Omega_i$ to each object $\mathbf{x}_i \in X$ on the basis of contextual information between the objects, and prior knowledge concerning the compatibility structure of the labelling (i.e., label consistency constraints). For simplicity, in this chapter we

also assume that the label sets Ω_i are identical for the different objects, as is the case of most relaxation labelling applications. We denote this universal label-set by $\Omega = \{\omega_k\}_{k=1}^l$ where l is the number of distinct labels. The compatibility matrix in this chapter is also denoted by R , and will have the same meaning as before. The elements $R_{ij}(\omega_i, \omega_j)$ are not necessarily be given beforehand. They may also be learned from the given data-sets. It is also convenient to assume spatial homogeneity, that is, the elements $R_{ij}(\omega_i, \omega_j)$ are invariant to location [43].

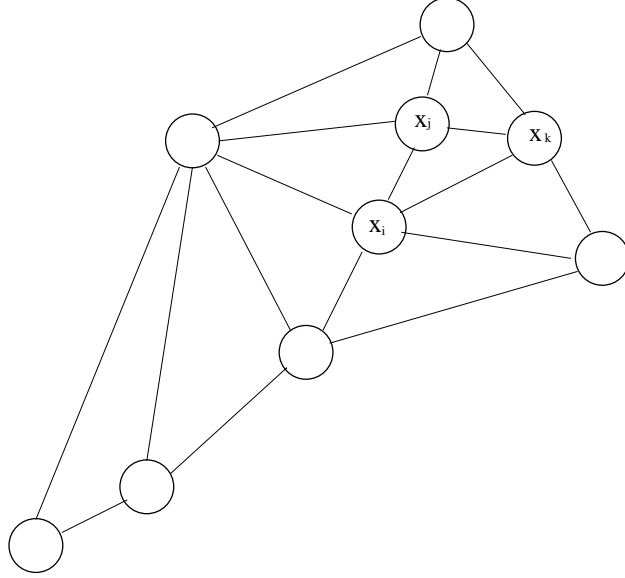


Figure 4.1: An example of a set of networked objects to be labeled.

As we have introduced before, relaxation labelling can be cast into either a discrete or a continuous setting. In this chapter, we keep our interest in the continuous case, and interpret the label confidences in a probabilistic manner. When posed in this way the classical relaxation labelling process involves the two distinct steps, namely, evidence combination and propagation. In the evidence combination step, a support function is used to compute the total contextual support using the current label probability estimates and the label compatibilities. In [71], Hummel and Zucker used the following arithmetic average support function for

assigning label ω_j to object j :

$$S^{(k)}(j \leftarrow \omega_j) = \sum_{i \in \mathcal{N}_j} \sum_{\omega_i \in \Omega} R_{ij}(\omega_i, \omega_j) P^{(k)}(i \leftarrow \omega_i) \quad (4.1)$$

where \mathcal{N}_j is the set of neighbours of node j , and $P^{(k)}(i \leftarrow \omega_i)$ is the probability that node i is assigned label ω_i at the k th iteration. A powerful alternative to the arithmetic average support in Equation (4.1) is to use the product support function (e.g., [80]) given by:

$$S^{(k)}(j \leftarrow \omega_j) = \prod_{i \in \mathcal{N}_j} \sum_{\omega_i \in \Omega} R_{ij}(\omega_i, \omega_j) P^{(k)}(i \leftarrow \omega_i) \quad (4.2)$$

With the support function to hand, the label probabilities are revised by using the update equation, e.g., [106]:

$$P^{(k+1)}(j \leftarrow \omega_j) = \frac{P^{(k)}(j \leftarrow \omega_j) S^{(k)}(j \leftarrow \omega_j)}{\sum_{\omega_i \in \Omega} P^{(k)}(j \leftarrow \omega_i) S^{(k)}(j \leftarrow \omega_i)} \quad (4.3)$$

The process is iterated until a consistent and unambiguous labelling is found.

4.3 Diffusion Processes on Graphs

As mentioned above, in this chapter we aim to pose relaxation labelling as a diffusion process on a graph whose nodes represent possible object-label assignments. Before we embark on this endeavor, we review some of the related theory of diffusion processes on graphs, and their links with spectral graph theory [27].

Diffusion processes are Markov processes with continuous time parameters and continuous state spaces [78]. They are local probability evolution processes with the property that the behaviour in the future is independent of the past given the current probabilities. During probability evolution, the local confidence of being in each state is propagated through time using a neighbourhood system defined on the state space. In this chapter, we cast the problem of updating object-label probabilities as a diffusion process on a support graph. We adopt this approach for a number of reasons. First, a graph provides a natural representation

of the topology of object arrangement. Interactions between different states can be easily represented by edge connections throughout the definition of a neighbourhood system on the graph. Second, diffusion processes share the common feature with relaxation labelling of propagating local confidence globally via local computation. Finally, diffusion processes on graphs are of increasing topical interest and have strong connection with the kernel methods [81, 11, 27, 28]. They have already given rise to a number of successful applications, e.g., [126, 129].

A number of alternatives exist to define a diffusion process [5]. In this thesis we follow the formulation based on the semi-group function. Such formulation can be found in the book by Yoshida [142]. Given a subset Γ of the entire state space \mathbb{S} and a suitable probability measure, a Markov process $\{y(t); t \geq 0\}$ is uniquely defined by an initial state probability vector \mathbf{p}_0 and a semi-group transition function $P(t, \mathbf{x}, \Gamma), t \in [0, \infty)$ defined on the state space. The semi-group transition function $P(t, \mathbf{x}, \Gamma)$ represents the probabilities for a state \mathbf{x} to be in a state in Γ at time $t + s$, starting from some time $s, s \geq 0$. It satisfies the following properties:

$$P(t, \mathbf{x}, \Gamma) \leq 1; \quad (4.4)$$

$$P(t + s, \mathbf{x}, \Gamma) = \int P(t, \mathbf{x}, d\mathbf{y}) P(s, \mathbf{y}, \Gamma), \quad (s, t \geq 0) \quad (4.5)$$

$$P(0, \mathbf{x}, \Gamma \setminus \{\mathbf{x}\}) = 0. \quad (4.6)$$

The property given in Equation (4.5) is also called the Chapman-Kolmogorov equation, and forms a semi-group of the transformation of $P(t, \mathbf{x}, \Gamma)$. Further in our application, we also have $\int_{\Gamma} P(t, \mathbf{x}, d\mathbf{y}) = 1$; that is, we assume conservative Markov processes. In addition, the semi-group operator P satisfies the property that $P(t, \mathbf{x}, \Gamma) \rightarrow I$ when $t \rightarrow 0$, where I is the identity transformation.

The probability transition function $P(t, \mathbf{x}, \Gamma)$ evolves according to the diffusion equation:

$$dP(t, \mathbf{x}, \Gamma)/dt = -\mathcal{F}P(t, \mathbf{x}, \Gamma), \quad \mathcal{F} = \frac{d}{d\mathbf{x}} \left\{ a(\mathbf{x}) \frac{d}{d\mathbf{x}} + b(\mathbf{x}) \right\}, \quad (4.7)$$

where \mathcal{F} is the so-called the Fokker-Planck operator. The state probability transition function at time t is then the solution of Equation (4.7), and takes on the exponential form

$$P(t, \mathbf{x}, \Gamma) = e^{-t\mathcal{F}}. \quad (4.8)$$

For simplicity, we write $P(t, \mathbf{x}, \Gamma)$ as $P(t)$. Given the initial state probability distribution \mathbf{p}_0 , the probability distribution at time t is then computed from the formula:

$$\mathbf{p}_t = P(t) \cdot \mathbf{p}_0 = e^{-\mathcal{F}t} \cdot \mathbf{p}_0 \quad (4.9)$$

In the discrete approximation, the operator \mathcal{F} is explicitly represented by a matrix, and the following formula can be used to compute the exponential:

$$e^{-t\mathcal{F}} = \sum_{k=0}^{\infty} \frac{(-1)^k t^k \mathcal{F}^k}{k!} \quad (4.10)$$

However, the convergence of the series depends on the values of t and the matrix \mathcal{F} . Slow convergence of the series usually leads to expensive computations (see [89] for a detailed discussion). When both the graph from the object set and the graph from the label set are regular, an efficient way of computing the eigenvectors can be found in the work by Chung and Yau [26]. Approximation methods for computing eigensystems of large matrices can also be used. An example is the Nyström method discussed in [134].

When \mathcal{F} is real and symmetric, we can perform the eigen-decomposition $\mathcal{F} = U\Lambda U^T$, where $U = (\mathbf{u}_1, \dots, \mathbf{u}_n)$ is the matrix formed from the eigenvectors of \mathcal{F} , and Λ is the diagonal matrix containing the corresponding eigenvalues. The solution vector of the diffusion equation is thus

$$\mathbf{p}_t = Ue^{-\Lambda t}U^T \cdot \mathbf{p}_0. \quad (4.11)$$

From a kernel perspective, the above development can be regarded as defining a kernel function for the graph representation, and seeks a solution from the mapped higher dimensional feature space. In fact, the construction of effective

kernel functions from exponentiation can be found in recent literature. For example, in [118, 81, 128] a new kernel matrix representation K' of a given data-set is given by:

$$K' = U\rho(\Lambda)U^T.$$

where $\Lambda = (\lambda_1, \dots, \lambda_m)$ are the eigenvalues, and U is the corresponding eigenvector matrix, of the matrix K , from the kernel functions. The function $\rho(\lambda_h)$, $h = 1, \dots, m$ is required to satisfy the condition that $\rho(\lambda_h) \rightarrow 0$ as $h \rightarrow \infty$. In [118], several functions are introduced from regularization theory, including the regularized Laplacian, $\rho(\lambda_h) = 1 + \sigma^2 \lambda_h$, and the diffusion kernel, $\rho(\lambda_h) = \exp(\lambda_h/\sigma^2)$. The latter cases corresponds to Equation (4.11).

Of these methods, the diffusion kernel [81] deserves special note. In [81], Kondor and Lafferty derived the diffusion kernel using kernel theory. According to this viewpoint the diffusion kernel is a symmetric and positive semi-definite function. For any square matrix \mathcal{H} whose elements are determined by a weight function, e.g., $[\mathcal{H}]_{\mathbf{x},\mathbf{y}} = f(\mathbf{x}, \mathbf{y})$, we can compute the matrix exponential

$$K_\beta = e^{\beta\mathcal{H}} \quad (4.12)$$

The resulting matrix K_β is a kernel function provided that the matrix \mathcal{H} is symmetric. That is, K_β is guaranteed to be symmetric and positive semi-definite if \mathcal{H} is symmetric. This kernel function also satisfies the diffusion equation:

$$\frac{d}{d\beta}K_\beta = \mathcal{H}K_\beta. \quad (4.13)$$

In [81], the Laplacian of the graph is used as an example of a diffusion kernel \mathcal{H} on a graph. Equation (4.13) is also referred to as the backward equation for a Markov process [32] (pp.181) if the graph nodes are interpreted as the discrete state space of the process. In this case, \mathcal{H} is analogous to the so-called Q -matrix for a continuous time Markov chain, with the matrix entries determining the transition rates between states. When Q is conservative, i.e. $q_{ii} = -\sum_{i \neq j} q_{ij}$, it can be shown that the corresponding Markov process is unique, and the matrix ex-

ponentiation in Equation (4.12) is convergent [121]. The matrix K_β also satisfies the initial condition $K_\beta(0) = I$ [32].

4.4 Relaxation labelling by diffusion

In this section, we show how the theory of diffusion processes on graphs can be used to formulate a new method for probabilistic relaxation labelling. To do this, first of all we need to construct a support graph as the underlying state space of the process. Denote the graph by $G_S(V_S, E_S, \mathcal{A}_S)$. The node-set $V_S = X \times \Omega$ is the Cartesian product of the object-set X and the label-set Ω . That is, each vertex of the support graph $v_{i\omega_i} = (\mathbf{x}_i, \omega_i) \in V_S$ represents the assignment of label $\omega_i \in \Omega$ to object $\mathbf{x}_i \in X$. We then define a label probability vector of our relaxation labelling process as $\mathbf{p}_t = [P_t(\theta_1 = \omega_1), \dots, P_t(\theta_1 = \omega_l), \dots, P_t(\theta_n = \omega_1), \dots, P_t(\theta_n = \omega_l)]$. Each component $P_t(\theta_i = \omega_i)$ of this vector represents the confidence of assigning a label $\omega_i \in \Omega$ to an object $\mathbf{x}_i \in X$. The components of the state-vector represent the probability of a random walker or a particle under diffusion residing at the node $v_{i\omega_i}$ at time t .

To satisfy the Markov property, a neighbourhood system needs to be defined. Here we use the object arrangement topology to define the neighbourhood. In some applications, the neighbourhood can be defined using the spatial proximity of the objects, and the requirement that two objects are neighbours can be established by thresholding the distance function. As in the previous chapter, we denote the set of neighbours of object \mathbf{x}_i as \mathcal{N}_i . Hence, the objects can be represented by an arrangement graph $G_X = (X, E_X, W_X)$ with node-set X , edge-set $E_X = \{(\mathbf{x}_i, \mathbf{x}_j) | \mathbf{x}_j \in \mathcal{N}_i\}$ and edge-weight function W_X . We will give explicit examples of how to compute the edge-weight matrix W_X when we discuss applications of our method in Section 5.

To pose the problem of relaxation labelling as a diffusion process on the support graph next we need to compute the infinitesimal generator \mathcal{F} and this re-

quires the weighted edge adjacency matrix \mathcal{A}_S defined in Equation (4.14). We assign edge-weights to the support graph so as to incorporate the label compatibilities R and the current label probabilities \mathbf{p}_t . Following conventional relaxation labelling methods, the edge weight on the support graph is set equal to the support associated with the object-label assignments for pairs of neighbouring nodes. According we let

$$\mathcal{A}_S(v_{i\omega_i}, v_{j\omega_j}) = \begin{cases} P(\theta_i = \omega_i)P(\theta_j = \omega_j)W_X(\mathbf{x}_i, \mathbf{x}_j)R_{ij}(\omega_i, \omega_j) & \text{if } \mathbf{x}_j \in \mathcal{N}_i \\ 0 & \text{otherwise} \end{cases} \quad (4.14)$$

The second step is to define the infinitesimal generator matrix \mathcal{F} for the support graph G_S . This is derived from the one-step transition matrix $P = D_S^{-1}\mathcal{A}_S$ of the support graph:

$$\mathcal{F} = I - D_S^{-1}\mathcal{A}_S \quad (4.15)$$

where D_S is the degree matrix of \mathcal{A}_S (as defined in Section 2.1, Chapter 2), and I is the identity matrix. The probability of each object-label assignment is then updated by Equation 4.9. To update the state-vector we can either perform matrix exponentiation directly or compute it from the eigensystem. As noted previously, the matrix P is generally not symmetric. Thus when computing the updated label probability vectors using the operator matrix's eigensystem, the relationship between its eigensystem and the symmetric adjacency matrix \mathcal{A}_S can be used. Suppose that the weighted adjacency matrix \mathcal{A}_S has the eigen-decomposition $\mathcal{A}_S = U\Lambda U^T$, and suppose that \mathcal{F} has the eigen-decomposition $\mathcal{F} = V^r\Lambda(V^l)^T$. Since $V^r \cdot (V^l)^T = (V^l)^T \cdot V^r = I$, we have:

$$\begin{aligned} \mathbf{p}_t &= e^{-t\mathcal{F}} \cdot \mathbf{p}_0 \\ &= e^{-t \cdot V^r \cdot \Lambda (V^l)^T} \cdot \mathbf{p}_0 \\ &= V^r \cdot e^{-t\Lambda} \cdot (V^l)^T \cdot \mathbf{p}_0 \end{aligned}$$

It is well known that the diffusion equation describes a smoothing effect, with a short time behaviour defined by its local structure [105]. For the feature cor-

respondence matching problem, smoothing or blurring is not the effect that we want. We regard the current label probabilities as the smoothed version of the initial sharp values. What we want is then to ‘push’ the diffused particles (probabilities) back to the right position. This can be viewed as running the diffusion process backwards in time [19, 48]. Thus now what we have is the probability vector \mathbf{p}_t . Instead of computing \mathbf{p}_t from Equation (4.9), now we wish to obtain \mathbf{p}_0 from this equation. Intuitively, we may multiply the inverse of the probability transition matrix $P(t)$ from the left on both sides of Equation (4.9) and obtain:

$$\begin{aligned} (P(t))^{-1} \mathbf{p}_t &= (P(t))^{-1} \cdot P(t) \cdot \mathbf{p}_0 \\ e^{(t\mathcal{F})} \cdot \mathbf{p}_t &= \mathbf{p}_0 \end{aligned} \quad (4.16)$$

It is well known that the matrix P needs to be symmetrized in order to be a valid kernel. Two frequently used methods for symmetrizing the probability transition matrix P are [121]

$$P_{sym} = \frac{P + P^T}{2} \quad \text{or} \quad P_{sym} = P^T P, \quad (4.17)$$

The latter is more familiar and is conventionally used in the machine learning community. In our experiments, we use this approach in the feature correspondence matching and toy labelling applications. In the application to data classification, in addition to the asymmetric probability transition function, a symmetrized generator matrix is also experimented with. However, the symmetrization process is applied to the adjacency matrix of the object-set since the local average distance between objects is used to scale the weight function. Specifically the matrix $W_X = \tilde{W}_X^T \tilde{W}_X$ where $\tilde{W}_{Xij} = \exp(-d_{\mathbf{x}_i, \mathbf{x}_j} / \sigma_i)$ and σ_i is a function of the average distance from object x_i to its neighbours.

Once the matrix operator \mathcal{F} is set up, the transition matrix $P(t)$ can be computed using Equation (4.8). The label probabilities are then updated iteratively using Equation (4.9). During each iteration, the label probabilities from the previous iteration are used to re-compute \mathcal{F} .

We also require a stopping criterion to halt the iteration of the labelling process. One simple approach is to halt the process after a fixed number of iterations. A more principled approach is to use the asymptotic properties of the label probabilities. To this end we make use of the entropy associated with the label probability distribution:

$$H_t = - \sum_{i=1}^n \sum_{\omega_i \in \Omega} P_t(\theta_i = \omega_i) \ln P_t(\theta_i = \omega_i), \quad (4.18)$$

where n is the number of objects in the object set X . The entropy can be regarded as the amount of disorder of the given system of label probabilities. It reaches its largest value if $P_t(\theta_i = \omega_i)$ has a uniform distribution, and decreases to zero if $P_t(\theta_i = \omega_i) \in \{0, 1\}, \forall \mathbf{x}_i, \omega_i$. The iteration of the relaxation process is halted if the total entropy decreases below a threshold, or if its change between two consecutive iterations is small. The process is also halted if the number of iterations exceeds a predefined value. The algorithm is summarized in Table 4.1.

4.5 Experiments

We experiment with our new relaxation labelling method on scene labelling and data classification, and feature correspondence matching problems. These problems have all been extensively studied in the literature. For a review see in Chapter 2. We commence our experiments with scene labelling using a toy labelling example, which has been used previously in the literature on relaxation labelling papers [106, 43]. We then consider the more general data classification tasks. Here we experiment on both synthetic and real world data-sets which are publicly accessible [46, 37]. For the feature correspondence matching problem, we first investigate the matching of three-way junctions in infra-red images (Figure 4.17), taken from Wilson and Hancock’s work [136]. We then use the new algorithm to locate one-to-one point feature correspondences for points between frames in an image sequence.

Table 4.1: The relaxation labelling algorithm

1. Initialization: Initial object-label assignment probabilities \mathbf{p}_0 , t , N , H , ΔH , H_{thresh} , ΔH_{thresh} ;

2. Compute the weight matrix W_X for the object graph;

3. Set $\mathbf{p}^{old} = \mathbf{p}_0$;

4. Compute the weighted adjacency matrix \mathcal{A}_S in current iteration using Eq. (4.14) and \mathbf{p}^{old} :

$$\mathcal{A}_S(v_{i\omega_i}, v_{j\omega_j}) = \begin{cases} P^{old}(\theta_i = \omega_i)P^{old}(\theta_j = \omega_j)W_X(v_i, v_j)R_{ij}(\omega_i, \omega_j) & \text{if } v_{j\omega_j} \in \mathcal{N}_{i\omega_i} \\ 0 & \text{otherwise} \end{cases}$$

5. Compute the infinitesimal generator matrix \mathcal{F} using the weighted adjacency matrix \mathcal{A}_S ;

6. For each iteration k , compute the updated label probabilities using Eq.(4.9):

$$\mathbf{p}^{new} = e^{-t\mathcal{F}} \cdot \mathbf{p}^{old};$$

7. Compute the entropy H using Eq. (4.18), and the variation $\Delta H = H^{(n)} - H^{(n-1)}$. If $(H < H_{thresh})$ or $(\Delta H < \Delta H_{thresh})$, go to step 10;

8. Set $k = k + 1$, and if $k \geq N$, go to step 10;

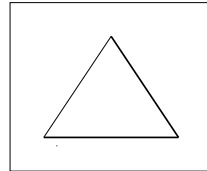
9. Set $\mathbf{p}^{old} = \mathbf{p}^{new}$, and go to step 4;

10. Assign maximum probability label to each object.

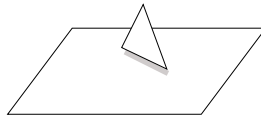
4.5.1 Scene labelling — A toy example

The problem of scene labelling can be broadly characterized as that of interpreting the structure of a scene by labeling image features extracted from it. We first experiment our newly developed relaxation method on a toy example of scene labeling which has been well-studied in several classical relaxation labeling works. One of the problems that hinders this task is that some of the feature labels may not appear in the scene, and it is possible that the number of labels is greater than the number of objects, i.e. features extracted. This may result in Markov processes with disjoint state spaces. An example is shown in Figure 4.3, which is from the classical relaxation labeling literature [106, 43]. In fact, relaxation labeling is one of the earliest techniques developed for solving this problem. For instance, Waltz’s discrete relaxation [130] and Rosenfeld, Hummel and Zucker’s probabilistic approach [106] both focused on interpreting objects in images.

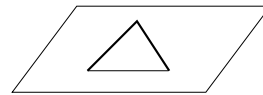
In this section we explore how our relaxation process performs when applied to the toy labelling problem which has been used in a number of early papers on relaxation labelling [106, 43]. The problem is to assign labels to the three edges of a triangle in order to interpret whether the object is a cutout floating above the background, a hole in the background, or a triangular flap that is either folded toward the viewer along one of its edges, or folded back onto the background [106]. Figure 4.2 shows the original problem and the possible interpretations. It is assumed that the label of each edge of the triangles belongs to a label set $\Omega = \{+, -, \leftarrow, \rightarrow\}$ of four labels, which have the meaning ‘convex’, ‘concave’, ‘occluding with the right-hand region’, and ‘occluding with the left-hand region’, respectively, as defined in [106]. We use the same compatibility matrices of these labels as given in [43]. They are shown as the matrices $R1$ and $R2$ in Table 4.2. The initial label probability assignments are also chosen the same as given in [43]. They are shown in the column ‘Initial Conditions’ in Table 4.2. In general not all line combinations are possible for a real world triangle, and if there is high probability of an edge-label assignment, the label probabilities of the other two edges



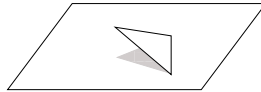
(a) The original image



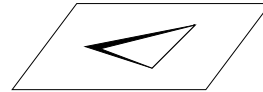
(b)



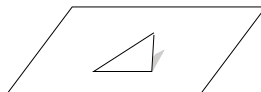
(c)



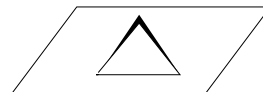
(d)



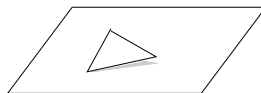
(e)



(f)



(g)



(h)



(i)

Figure 4.2: The toy labeling problem: Label each of its edges to interpret the triangle. (*This is taken from Rosenfeld, et. al. [106].*)

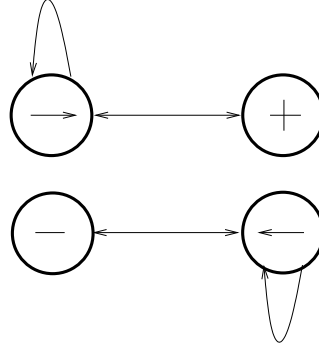


Figure 4.3: The four possible labels ‘ $+$, ‘ $-$, ‘ \leftarrow , ‘ \rightarrow ’ and their state transition graph of the toy labeling example

shall be affected. These constraints are represented by the compatibility matrices. We expect to reach the same consistent labeling which also satisfies these constraints as the results in [43] for the three edges of a triangle. It is interesting to note that some of the labels are exclusive of each other, as one can see from Figure 4.3. The entries in both the compatibility matrices and the initial label probability matrices are indexed in the order of $+$, $-$, \leftarrow , \rightarrow .

To set up the weighted adjacency matrix \mathcal{A}_S we assume that all three objects (here the three edges of a triangle) are neighbours to each other, and the corresponding elements of the edge weight matrix W_X are set equal to 1. We update the label probabilities using equation (4.11). The state transition graph for all the possible labels is shown in Figure 4.3. From this figure it is clear that the state space of the corresponding Markov process has two disjoint subsets. Experimental results are shown in Table 4.2. From the table we can see that our results all agree with those given in [43]. In other words, our diffusion-based relaxation process locates the same consistent solutions.

4.5.2 Data classification

We now turn our attention to the problem of data-classification. This is a general problem in pattern recognition that requires the data-points to be classified into

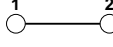
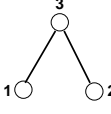
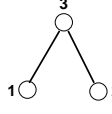
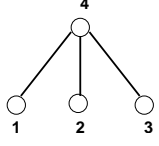
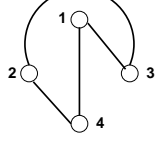
Table 4.2: Relaxation labeling results on the toy example

Initial Conditions	$R1 = \begin{bmatrix} 0.5 & 0 & 0.5 & 0 \\ 0 & 0.5 & 0 & 1 \\ 0.5 & 0 & 0 & 0 \\ 0 & 0.5 & 0 & 0 \end{bmatrix}$	$R2 = \begin{bmatrix} \frac{2}{3} & 0 & 1 & 0 \\ 0 & \frac{2}{3} & 0 & 1 \\ \frac{1}{3} & 0 & 0 & 0 \\ 0 & \frac{2}{3} & 0 & 0 \end{bmatrix}$		
	Our results*	Faugeras&Berthod ($\alpha = 1$)	Our results	Faugeras&Berthod ($\alpha = 1$)
$P_1 =$				
.25 .25 .25 .25	.3623 .3623 .1377 .1377	.375 .375 .125 .125	.4214 .4214 .0786 .0786	.3333 .3333 .1667 .1667
.25 .25 .25 .25	.3623 .3623 .1377 .1377	.375 .375 .125 .125	.4214 .4214 .0786 .0786	.3333 .3333 .1667 .1667
.25 .25 .25 .25	.3623 .3623 .1377 .1377	.375 .375 .125 .125	.4214 .4214 .0786 .0786	.3333 .3333 .1667 .1667
$P_2 =$				
.5 0 .5 0	.7245 0 .2755 0	.75 0 .75 0	.8428 0 .1572 0	.6667 0 .3333 0
.5 0 .5 0	.7245 0 .2755 0	.75 0 .75 0	.8428 0 .1572 0	.6667 0 .3333 0
.5 0 .5 0	.7245 0 .2755 0	.75 0 .75 0	.8428 0 .1572 0	.6667 0 .3333 0
$P_3 =$				
.5 0 .5 0	.7282 0 .2718 0	.75 0 .75 0	.8450 0 .1550 0	.6667 0 .3333 0
.4 0 .6 0	.7153 0 .2847 0	.75 0 .75 0	.8351 0 .1649 0	.6667 0 .3333 0
.5 0 .5 0	.7282 0 .2718 0	.75 0 .75 0	.8450 0 .1550 0	.6667 0 .3333 0
$P_4 =$				
.5 0 .5 0	.7323 0 .2677 0	.75 0 .75 0	.8475 0 .1525 0	.6667 0 .3333 0
.3 0 .7 0	.7054 0 .2946 0	.75 0 .75 0	.8475 0 .1730 0	.6667 0 .3333 0
.5 0 .5 0	.7323 0 .2677 0	.75 0 .75 0	.8475 0 .1525 0	.6667 0 .3333 0
$P_5 =$				
.3 0 .7 0	.7135 0 .2865 0	.75 0 .75 0	.8322 0 .1678 0	.6667 0 .3333 0
.3 0 .7 0	.7135 0 .2865 0	.75 0 .75 0	.8322 0 .1678 0	.6667 0 .3333 0
.5 0 .5 0	.7396 0 .2604 0	.75 0 .75 0	.8519 0 .1481 0	.6667 0 .3333 0
$P_6 =$				
.2 0 .8 0	.7022 0 .2978 0	.75 0 .75 0	.8229 0 .1771 0	.6667 0 .3333 0
.3 0 .7 0	.7186 0 .2814 0	.75 0 .75 0	.8356 0 .1644 0	.6667 0 .3333 0
.5 0 .5 0	.7442 0 .2558 0	.75 0 .75 0	.8548 0 .1452 0	.6667 0 .3333 0
$P_7 =$				
.3 .2 .3 .2	.4347 .2898 .1653 .1102	.375 .375 .125 .125	.5057 .3371 .0943 .0629	.3333 .3333 .1667 .1667
.3 .2 .3 .2	.4347 .2898 .1653 .1102	.375 .375 .125 .125	.5057 .3371 .0943 .0629	.3333 .3333 .1667 .1667
.3 .2 .3 .2	.4347 .2898 .1653 .1102	.375 .375 .125 .125	.5057 .3371 .0943 .0629	.3333 .3333 .1667 .1667
$P_8 =$				
.3 .2 .3 .2	.4216 .3067 .1555 .1162	.375 .375 .125 .125	.4902 .3550 .0892 .0657	.3333 .3333 .1667 .1667
.25 .25 .25 .25	.4118 .3165 .1507 .1210	.375 .375 .125 .125	.4791 .3660 .0859 .0690	.3333 .3333 .1667 .1667
.2 .2 .4 .2	.4049 .3107 .1666 .1178	.375 .375 .125 .125	.4739 .3616 .0974 .0671	.3333 .3333 .1667 .1667

Note: The results list the final label probabilities. The columns represent all possible labels in the order of +, −, ←, →, and the rows represent the three edges.

different groups, where the number of groups may be known beforehand. Here we evaluate our newly developed algorithm on both synthetic and real world data-sets. In order to compare our method with alternative data clustering methods, we use publicly available data-sets [46, 37]. We also experiment with the same data-sets using spectral clustering and kernel k -means clustering methods to provide comparison. We study the performance of the algorithm with different numbers of clusters, different cluster size, and different cluster distributions. These are considered to be the most common variations in real world data-sets.

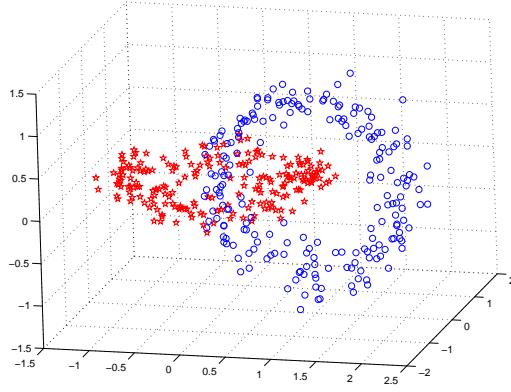
Table 4.3: Representations of label compatibilities

R2	G3	RG	G4_1	G4_2
				
$\begin{bmatrix} 1 & .3 \\ .3 & 1 \end{bmatrix}$	$\begin{bmatrix} 1 & 0 & .5 \\ 0 & 1 & .5 \\ .5 & .5 & 1 \end{bmatrix}$	$\begin{bmatrix} 1 & 0 & .5 \\ 0 & 1 & .5 \\ .5 & .5 & 1 \end{bmatrix}$	$\begin{bmatrix} 1 & 0 & 0 & .5 \\ 0 & 1 & 0 & .5 \\ 0 & 0 & 1 & .5 \\ .5 & .5 & .5 & 1 \end{bmatrix}$	$\begin{bmatrix} 1 & 0 & .1 & .5 \\ 0 & 1 & .1 & .5 \\ 0 & .1 & 1 & 0 \\ .5 & .5 & 0 & 1 \end{bmatrix}$

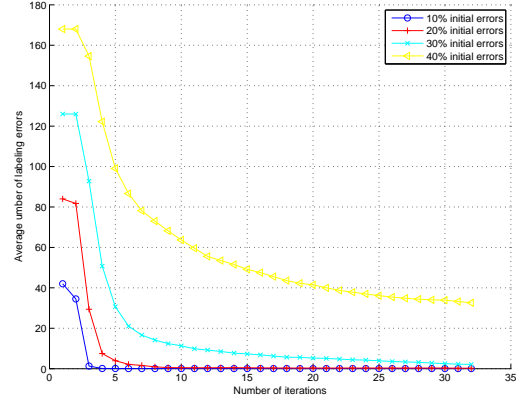
One advantage of relaxation labelling is its ability to reduce ambiguities in label assignments. For this reason, we first assign all the objects in our data-sets the correct labels. Next we experiment with our algorithm by randomly selecting a fraction of the assigned object labels and flipping the label probabilities randomly to take on new values. The fraction of labels to be flipped ranges from 10% to $\frac{1}{|Q|}$, i.e., the probability of uniform label assignment. We use a Gaussian weight function to compute the weighted adjacency matrix \mathcal{A} in Equation (4.14):

$$W_X(\mathbf{x}_i, \mathbf{x}_j) = \exp\left(-\frac{d_{ij}}{\sigma_i}\right), \quad (4.19)$$

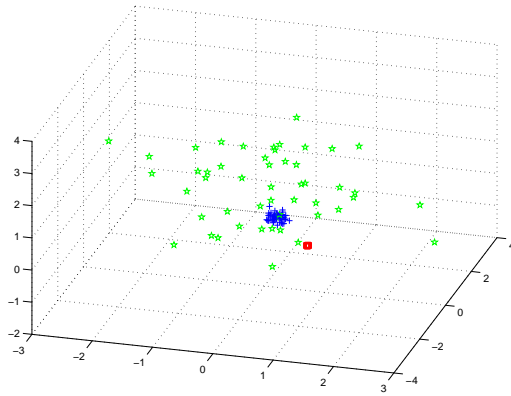
where d_{ij} is the Euclidean distance between objects $\mathbf{x}_i, \mathbf{x}_j \in X$. The values of σ_i in the weight function Equation 4.19 are chosen to be a function of the average distance between nearest neighbours. The cluster label compatibility coefficients for each synthetic data-set are shown in Table 4.3.



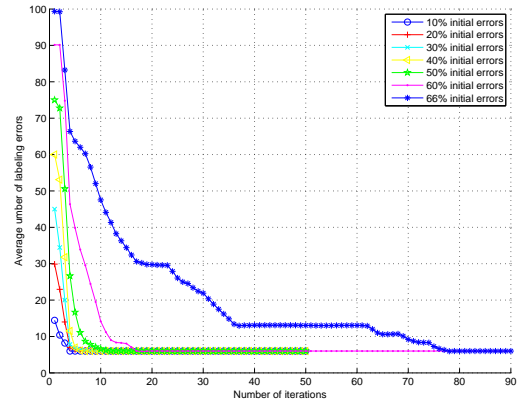
(a) Two Rings data-set (R2)



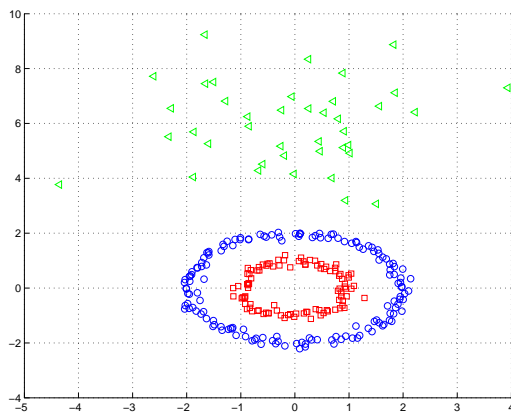
(b) Labeling results of R2



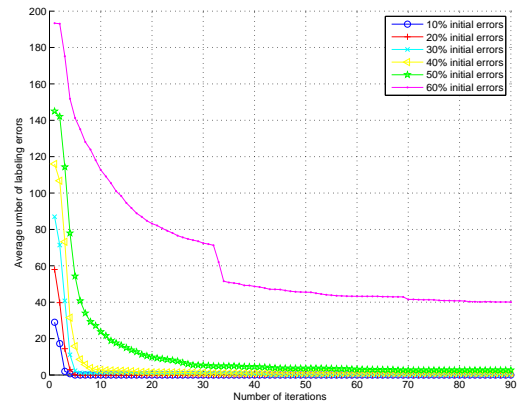
(c) Three-Gaussian data-set (G3)



(d) Labeling results of G3



(e) Ring-Gaussian data-set (RG)



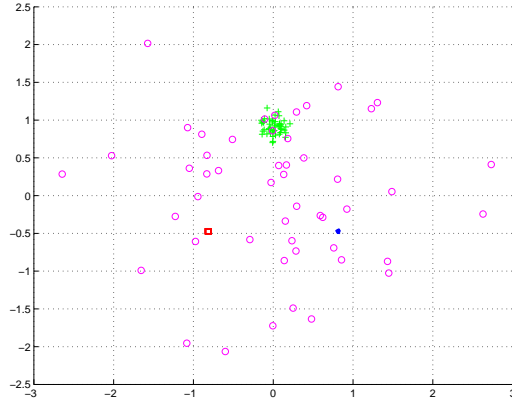
(f) Labeling results of RG

Figure 4.4: Labeling results for synthetic and real world data-sets. *Left column: The original data-sets; Right column: The clustering results.*

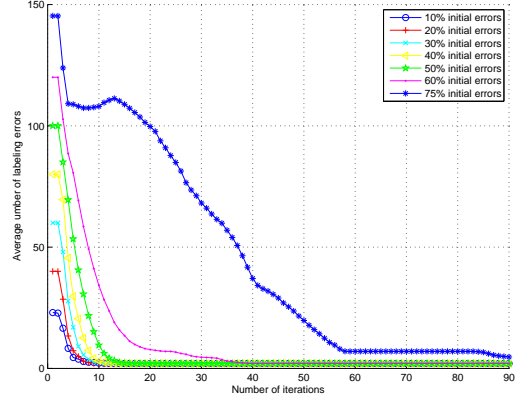
We first experiment with the five synthetic data-sets shown in the left column of Figures 4.4 and 4.5. The average clustering error on 20 runs for each data-set with different initial label probabilities are shown in the right column of Figures 4.4 and 4.5. For each case, as the iteration number increases then so the error falls rapidly. The eigenvalue distribution at each iteration of the G4 data-set is shown in Figure 4.6. As the number of iterations changes then so the distribution of eigenvalues become more peaked towards the origin. As the magnitude of eigenvalues illustrates the significance of the corresponding eigenvectors, the cluster information becomes more concentrated within the first few eigenvectors. Figure 4.7 shows the change of total entropy during iterations of the relaxation process. As the algorithm iterates then so the entropy approaches a stable minimum value. Figure 4.8 shows an example of the evolution of the label probability vectors. Here we have used a random initialization. For ease of visualization, the probability vectors in the diagrams are arranged as three groups for three clusters. In each group, there are three subgroups which represent the probability of assigning the objects to three different cluster labels. Initially the clusters are poorly defined, but as the process iterates they become more clearly defined.

For comparison, we have also experimented on these data-sets using kernel k -means and spectral clustering. For kernel k -means, the code (except the computing of the kernel matrix) was taken from [115]. The kernel function we choose here is the Gaussian kernel, and the initial cluster centres are chosen randomly, as implemented in the code described in [115]. To avoid divide by zero errors, we further added $1E - 5$ to each entry in the initial cluster centre indicator matrix. Since the performance of k -means depends on initial values, we provide the average values of 20 runs for each of the data-sets G3, G4.1, and G4.2. The number of errors in these experiments are 2, 13.7, and 43.45, respectively. The results of kernel k -means clustering are shown in Figure 4.9. The results for the RG and R2 data-sets are shown in Figures 4.10(a) and 4.10(b).

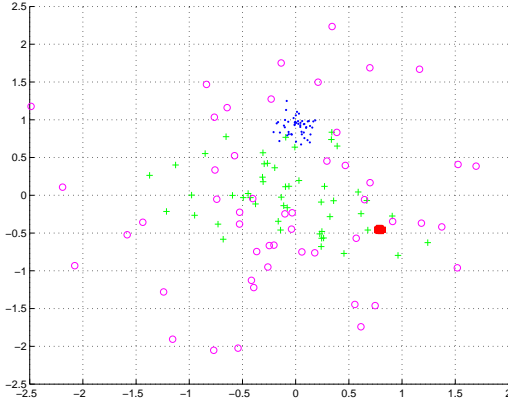
We also computed the eigenvectors of the data-sets G3, G4.1 and RG. The



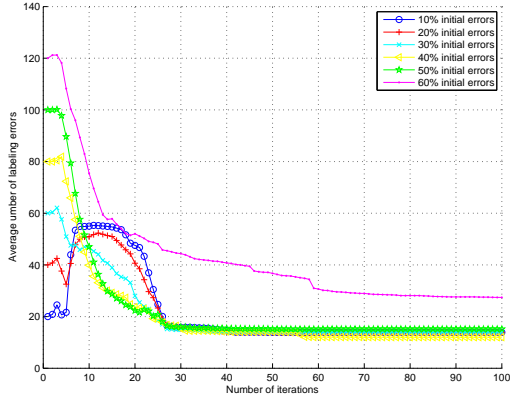
(a) Four-Gaussian data-set (G4_1)



(b) Labeling results of G4_1



(c) Four-Gaussian data-set (G4_2)



(d) Labeling results of G4_2

Figure 4.5: Labeling results for synthetic and real world data-sets. *Left column: The original data-sets; Right column: The clustering results.*

proximity matrix we use is the weighted adjacency matrix (since the first eigenvector of the adjacency matrix corresponds to the Fiedler vector of the Laplacian matrix). The first $m - 1$ eigenvectors, where m is the number of clusters in each data-set, for each data-sets are shown in Figures 4.11, 4.12, and 4.13, respectively.

We next experiment with our algorithm on two real world data-sets, namely Iris and Wine [37]. They have already been studied in a variety of papers, e.g., [46]. Both data-sets contain three clusters. The dimensionality of the Iris data is four, while that of the Wine data is nine. The weight function is again chosen to be the Gaussian, and the compatibility matrices are identical to those used with synthetic data-sets. The results are shown in Figure 4.14. The average clustering

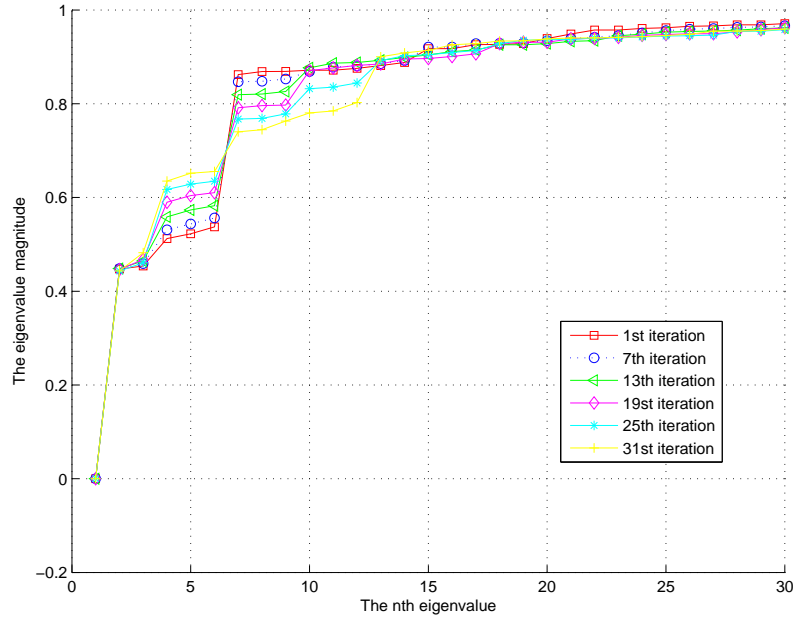


Figure 4.6: The distribution of the first thirty eigenvalues of the Laplacian matrix in data-clustering (G4_1 synthetic data-set, with randomly assigned initial label probability values) at iterations 1, 7, 13, 19, 25, and 31.

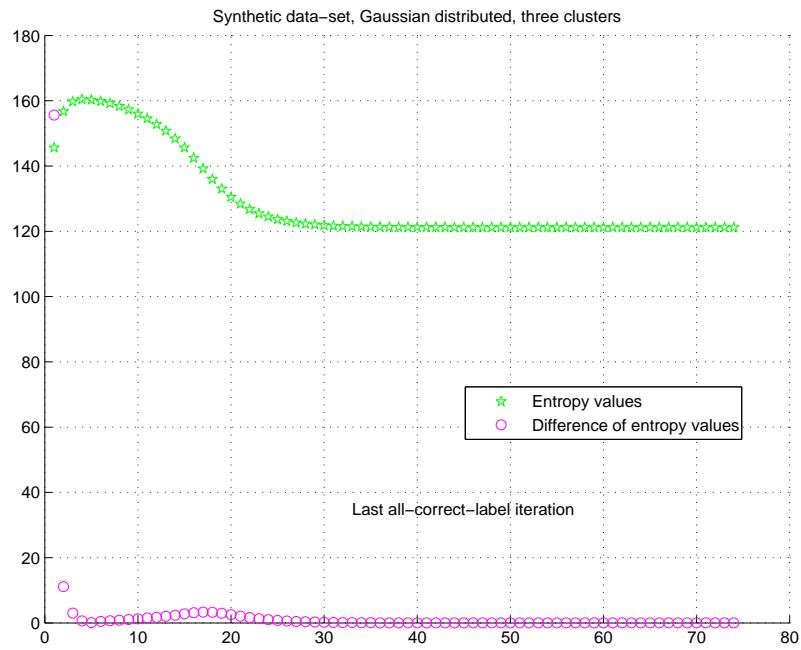


Figure 4.7: The changing of entropy values during iteration (data-set G3).

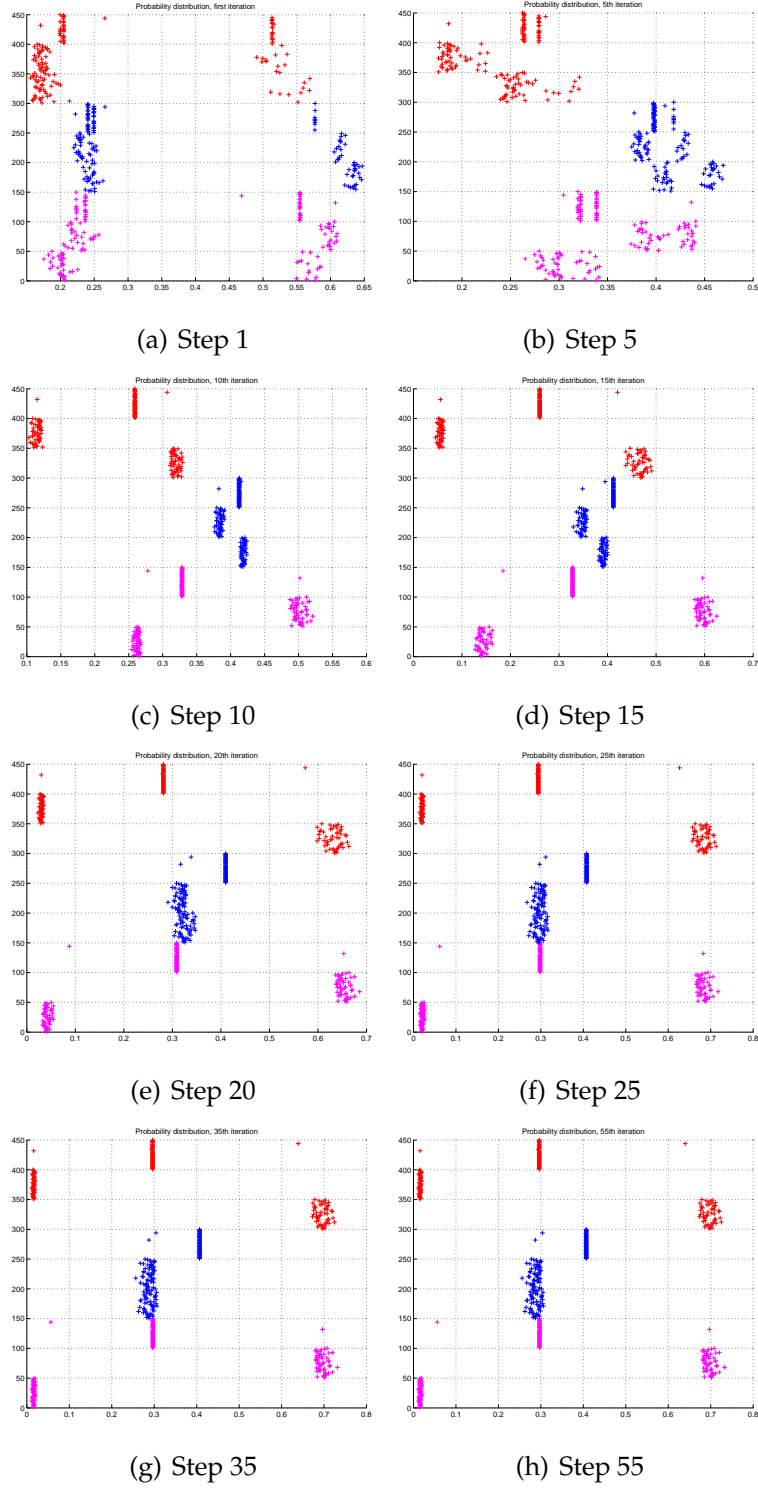
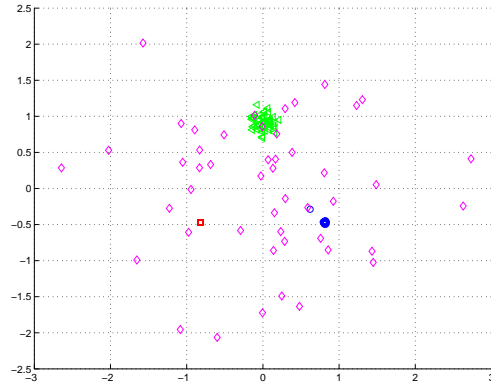
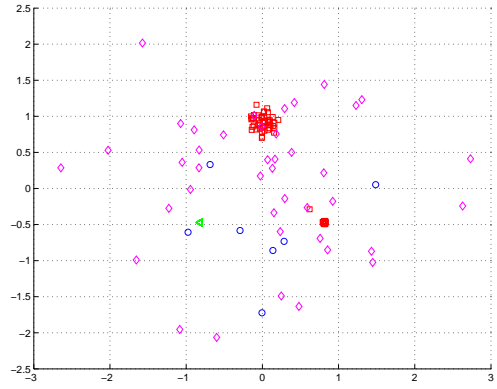


Figure 4.8: Evolution of the label probabilities (data-set G3). The horizontal axis gives the probability values and the vertical axis the components of the label probability vector.

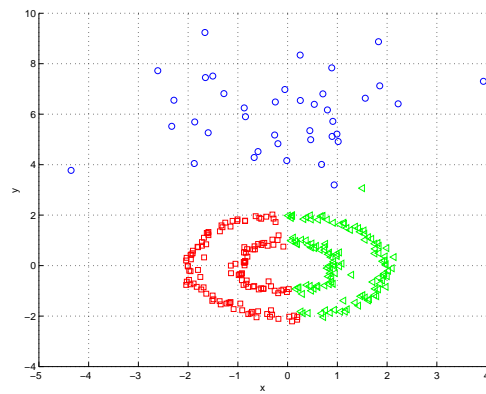


(a) G4-1, good case

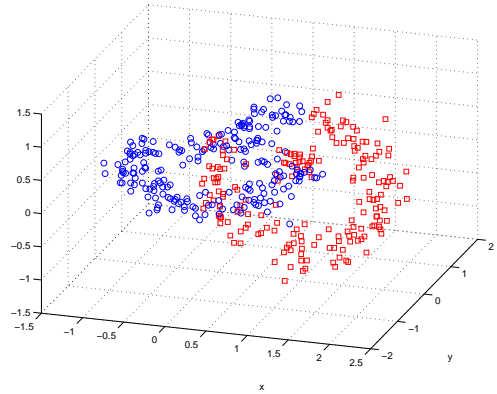


(b) G4-1, bad case

Figure 4.9: Labeling results from kernel k -means.



(a) RG



(b) R2

Figure 4.10: Labeling results from kernel k -means.

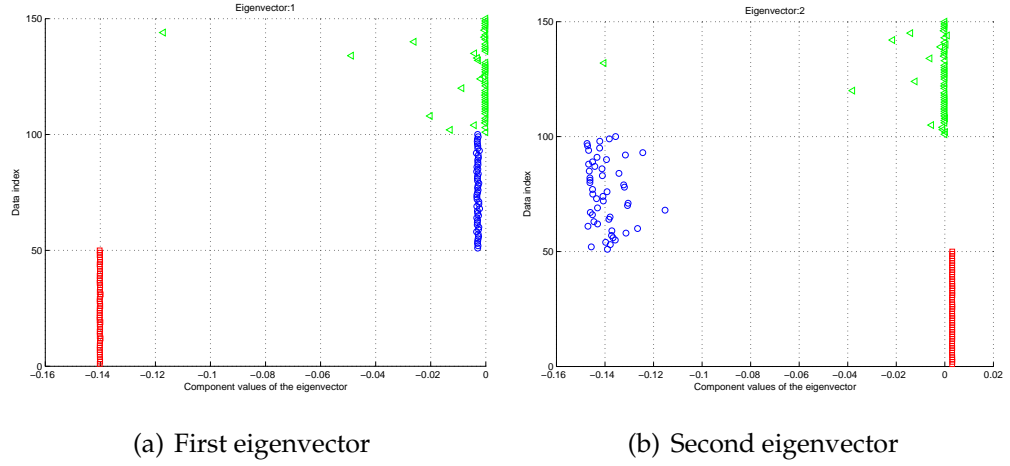


Figure 4.11: The first two eigenvectors of G3.

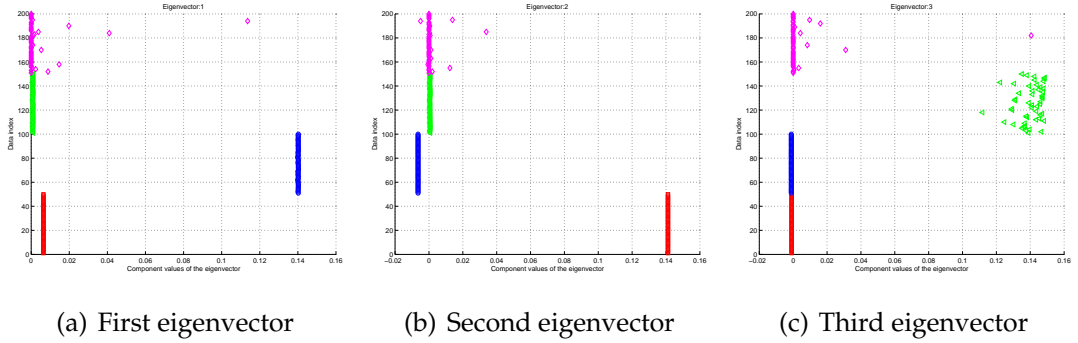
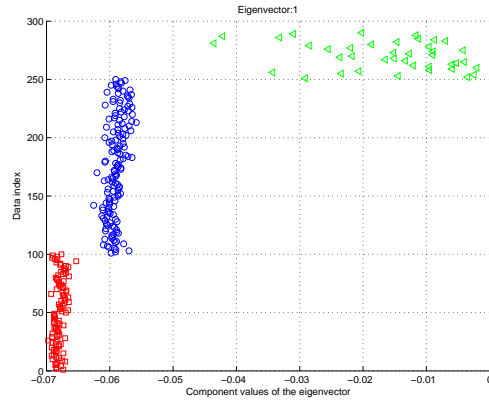


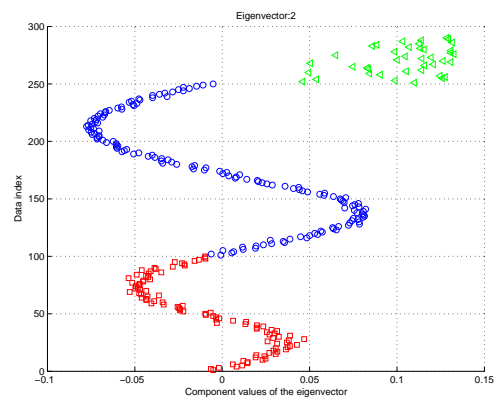
Figure 4.12: The first three eigenvectors of G4.1.

errors of the 20 runs from the kernel k -means algorithm for the Wine and Iris data-sets are 6.3 and 17, respectively. The first two eigenvectors of the adjacency matrix of the two data-sets are shown in Figures 4.15 and 4.16.

From the above experimental results we can see that the algorithm converges rapidly in the above experiments to the desired distribution, where the majority of the labels are assigned correctly. When less than 50% of the labels are initialized in error, the algorithm corrects them in just a few iterations. The numbers of iterations required are significantly smaller than for traditional relaxation labelling algorithms. The accuracy of the clustering are comparable with those obtained using alternative data-clustering algorithms, e.g., the results given by the work by Fischer and Poland [46]. Comparing with the kernel k -means and

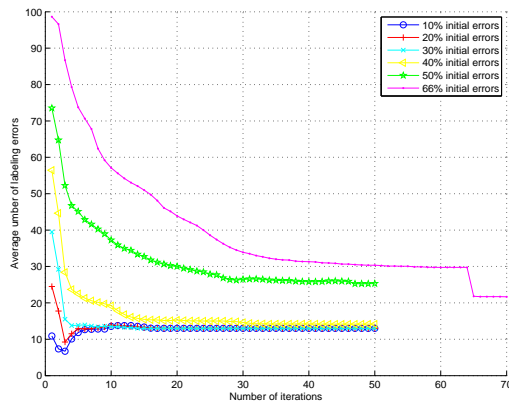


(a) First eigenvector

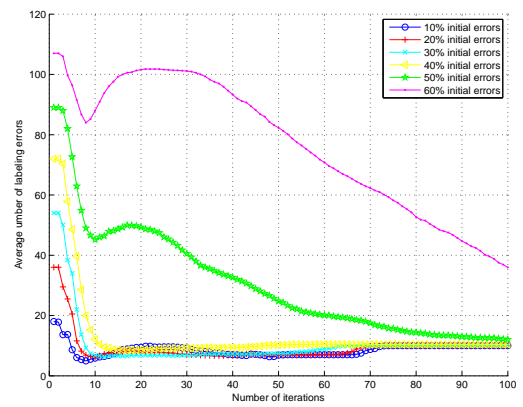


(b) Second eigenvector

Figure 4.13: The first two eigenvectors of RG.

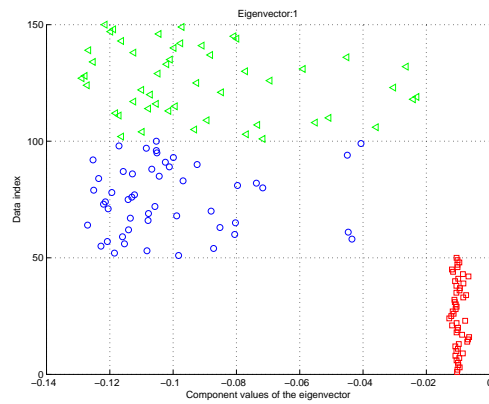


(a) Labelling results, Iris

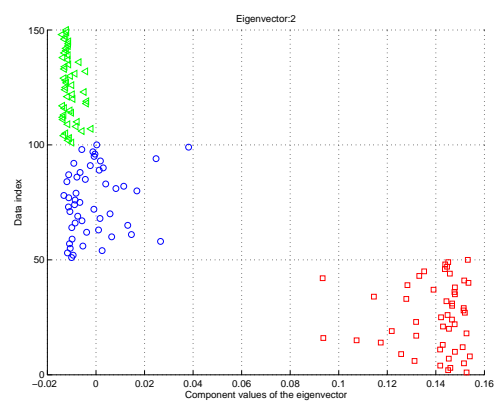


(b) Labelling results, Wine

Figure 4.14: Labelling results of real world data-sets. *Left: Iris; Right: Wine.*



(a) First eigenvector



(b) Second eigenvector

Figure 4.15: The first two eigenvectors of Iris.

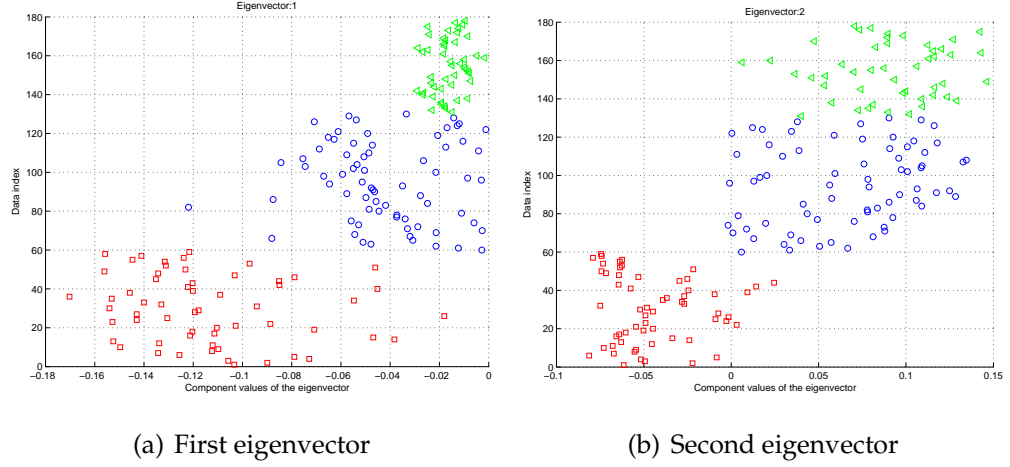


Figure 4.16: The first two eigenvectors of Wine.

graph spectral clustering methods, we can see that our algorithm can cope with a great variety of data distribution shapes.

4.5.3 Feature correspondence matching

The final experimental evaluation of our new relaxation labelling algorithm is concerned with feature correspondence matching. This involves finding a bijective correspondence mapping between two feature point-sets. We provide experiments on three different image data-sets. Here the first example is to match three-way junctions in aerial images of road network. This example is taken from the work of Wilson and Hancock [136]. The next two examples involve feature points extracted from two image sequences.

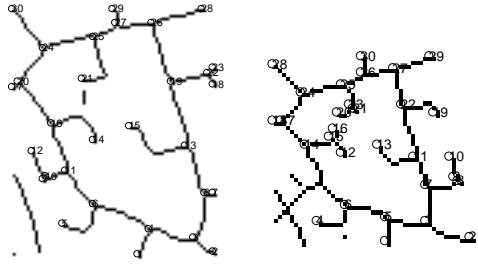
Suppose that the features extracted from the pair of images to be matched are represented by the undirected graphs $G_M = (V_M, E_M)$ and $G_D = (V_D, E_D)$ where V_M and V_D are the node sets representing the features and E_M and E_D are the edge sets representing the existence of connecting structure in the image. To perform feature correspondence matching, we treat the nodes in the second graph as our label set, and the nodes in the first graph as the objects being labelled. As a result $V_S = V_M \times V_D$ and $E_S = E_M \times E_D$. Following Wilson and Hancock [136] the label compatibility coefficients $R_{ij}(\omega_i, \omega_j)$ of the matrix R are computed from the edge



(a)



(b)



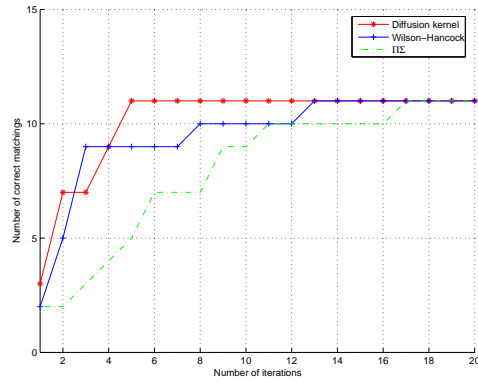
(c)

(d)

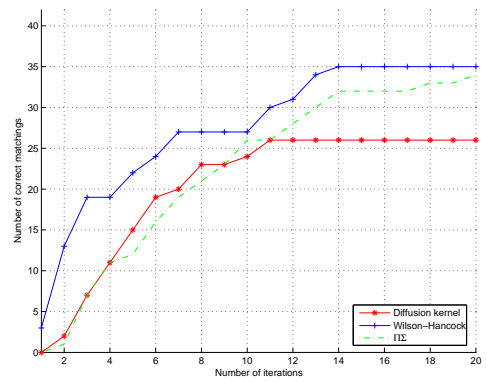


(e)

(f)

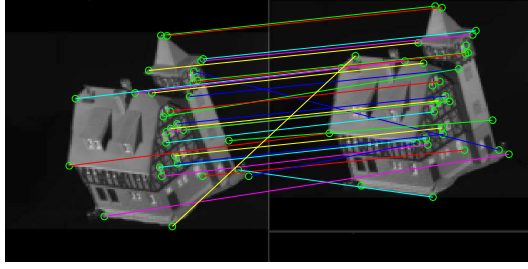


(g)

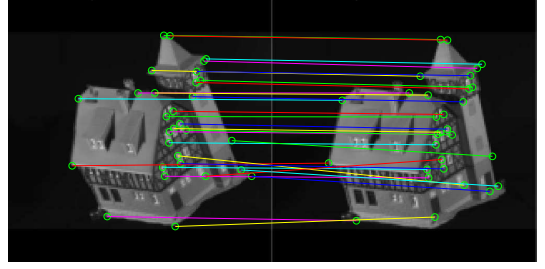


(h)

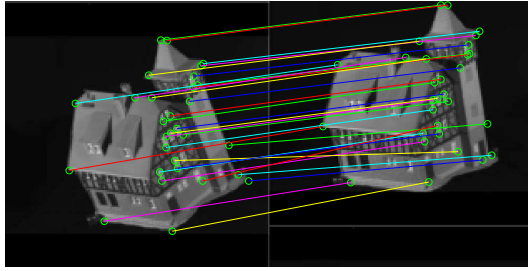
Figure 4.17: Matching three-way junctions. (a),(b): Original image pair; (c),(d) & (e),(f): extracted edge contours; (g): results from contour pair (c),(d); (h): results from contour pair (e),(f).



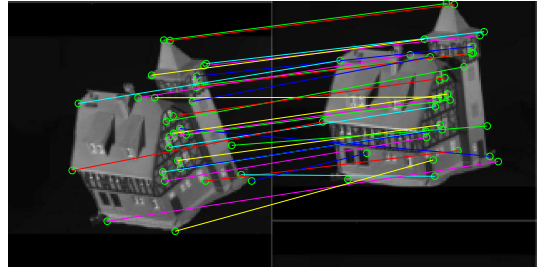
(a) Frames 01 and 03



(b) Frames 01, 04



(c) Frames 01, 05



(d) Frames 01, 06

Figure 4.18: Matching results, CMU house data

densities of the graphs being matched:

$$R_{ij}(\omega_i, \omega_j) = \begin{cases} \frac{|V_M|^2}{|E_M|} & \text{if } (\omega_j, \omega_i) \in E_M \\ 1 & \text{if } (\omega_j, \omega_i) \in (\phi \times V_M) \cup (V_M \times \phi) \\ 0 & \text{if } (\omega_j, \omega_i) \in V_M \times V_M - E_M \end{cases} \quad (4.20)$$

The edge-weight matrix W_X for the graph G_D is computed using Equation (4.14) in which the σ_i parameter is chosen as a constant proportional to the average pairwise distance of all the nodes. The initial label probabilities are computed from the feature measurement values using the road-length and the angle of each junction following the method outlined in [136]. Results from the first and second image pairs are shown in Figure 4.17(g) and Figure 4.17(h), respectively. The matching results from two classical relaxation labelling methods using the support functions (4.1) and (4.2) are also shown for comparison (the probabilities are updated by Equation (4.9)) and the results are taken from [136].

Our second experiment involves the problem of point feature correspondence matching. Here we also choose the Gaussian function in Equation (4.19) as the

weight function in computing the weighted adjacency matrix in Equation (4.14). The label compatibility matrices are set as the simple adjacency matrices with matrix components equal to 1 if the two labels are immediate neighbours of each other. The immediate neighbours are defined as the four nearest Manhattan neighbours of each node. We first choose four image pairs from the CMU house sequence. These are the same as the examples chosen in Luo and Hancock’s work of feature matching by using the EM algorithm and SVD methods [84]. Our experimental results of the matching are shown in Figure 4.18. From the figures we can see that the matching results are significantly improved comparing with the results in [84]. Another example is of applying the method on point-sets from two images of a hand sequence. The matching results are shown in Figure 4.19. In order to give the evolving behaviour of the eigenspectrum, Figure 4.20 shows the first fifty eigenvalues of the adjacency matrix obtained from feature point-sets of image frames 09 and 11 at iterations. Again as iteration proceeds, the distribution becomes more peaked at the origin.

In these examples, the three-way junction matching examples and the hand sequence data in Figure 4.19 contain visible shape deformations while in the example of Figure 4.18 the number of feature points in each data-set are not the same.

As we can see from the figures, the method copes well with these situations. From these results we can see that the new relaxation labelling method is effective in labelling and feature correspondence matching tasks.

4.5.4 Computational complexity

In classical relaxation labelling algorithms such as the one introduced in [80], the computational complexity is of $O(n^2m^2)$ where n and m are the number of objects and labels, respectively. The computational overhead in our newly developed relaxation labelling algorithm comes mainly from computing the matrix exponential $e^{-t\mathcal{F}}$. One frequently used method is scaling and squaring, as is the

MATLAB implementation of the function `expm` [50]. The computation complexity is of $O((mn)^3)$. When taking advantage of the symmetry of the given matrix, the main computation is the eigen-decomposition and a matrix-vector product at each iteration. The computational complexity is also of $O((mn)^3)$. In our experiments, when the MATLAB `expm` function is used, the data-clustering of G4 the data-set takes 216.8 seconds which involves 31 relaxation iterations. The feature correspondence matching of the hand sequence data experiments takes 8465.3 seconds for matching each data-set pair, and requires 69 iterations. These figures were obtained with a desktop PC with an AMD Athlon Dual Core Processor 4200+ and 1GB memory. However, since our matrices here are generally sparse, more efficient methods can be used to approximate the eigen-systems. For example, the Lanczos method may be applied, which involves $m \cdot n$ matrix-vector multiplications for our Cartesian product matrix of dimension $(mn) \times (mn)$ [50]. The Krylov approximation method can be applied to compute the product of the probability vector and the matrix exponential. In [66] the values of $e^{-t\mathcal{F}}\mathbf{v}$ are computed directly by approximating the product of the exponential of the matrix and the vector by finding an element of the Krylov subspace $K_m = \{\mathbf{v}_0, (t\mathcal{F})\mathbf{v}_0, \dots, (t\mathcal{F})^{k-1}\mathbf{v}_0\}$ that best approximates $\mathbf{v}_t = e^{-t\mathcal{F}}\mathbf{v}_0$. Generally k is much smaller than the original dimension mn of matrix \mathcal{F} . This gives the potential of improving the efficiency of our algorithm significantly.

4.6 Discussion

In this chapter we have developed a new relaxation labelling algorithm using the diffusion equation. The starting point is a support graph whose nodes represent possible object-label assignments and whose edges encode adjacency relations together with label compatibility constraints. The state vector of the diffusion process represents the object labelling probabilities. Under the diffusion equation, the updating of this probability is determined by the adjacency matrix of

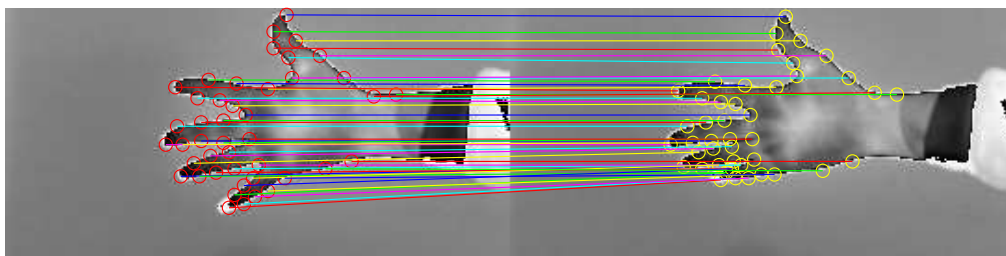
the support graph. Two distinguished cases were introduced, namely, the data classification and the feature correspondence matching problems. The latter was regarded as running the diffusion process ‘backwards in time’. Experiment with the new relaxation labelling methods on a toy labelling problem, a group of data classification tasks, and a set of feature correspondence matching problems gave good performances.

The diffusion process setting of the development ensures the global propagation of local labelling confidence and the reduction in the number of iterations. The essential component is the kernel function which defines the diffusion process. For the tasks we applied in this paper, the exponential kernels are used which is the natural formula for a diffusion process.

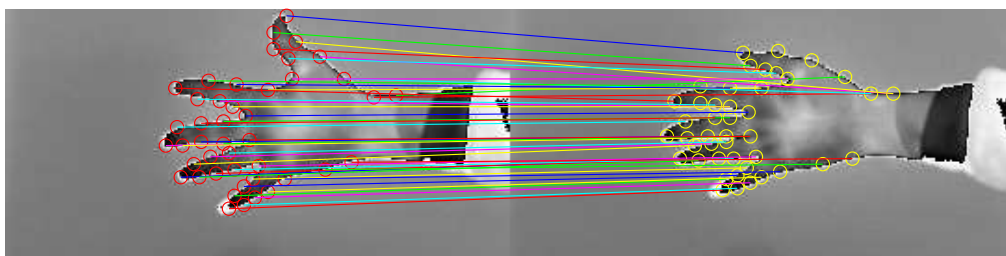
The work reported here can be extended in a number of ways. One obvious direction is to use the framework to develop a discrete relaxation algorithm [130, 57]. Here, rather than dealing with label probabilities, a single label is assigned to each object at each epoch. To realize such an algorithm using a diffusion process, an efficient sampling process is required. Alternatives here include particle filters [7] with an appropriate sampling method such as the importance sampling [103]. A second avenue for investigation would be to investigate ways of rendering the process more efficient. These include a number of approximation techniques. As mentioned in the chapter, Krylov subspace approximation method for computing the exponential of the matrix-vector product, or the Nyström method for computing the approximate eigenvectors for the large sparse matrix [134] can be used for improving the efficiency. The Krylov approach introduced in [89] also shows the possibility of running the computation in parallel.

Finally, the algorithm developed in this chapter has only been applied to two applications. There are potentially a much wider range of computer vision and pattern recognition tasks that it can be applied to. For example, it can be used to the tasks of image segmentation, with the different image segments as the data

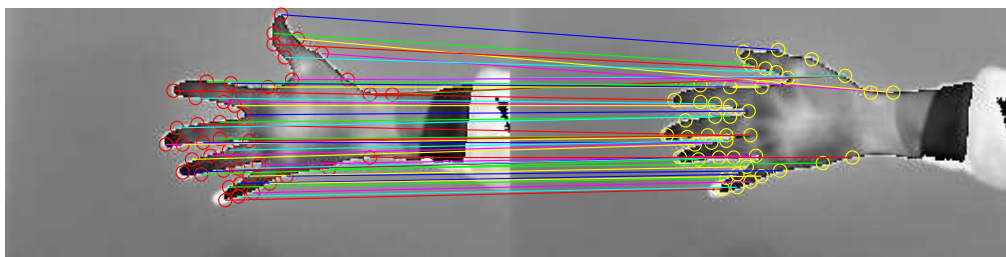
labels, and the rest runs similar to our data classification experiments.



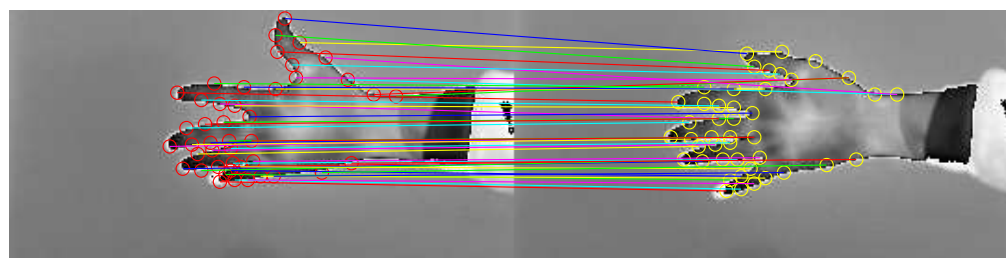
(a) Frames 08 and 11



(b) Frames 08 and 25



(c) Frames 09 and 25



(d) Frames 11 and 25

Figure 4.19: Point correspondence matching results of image frames from a hand sequence.

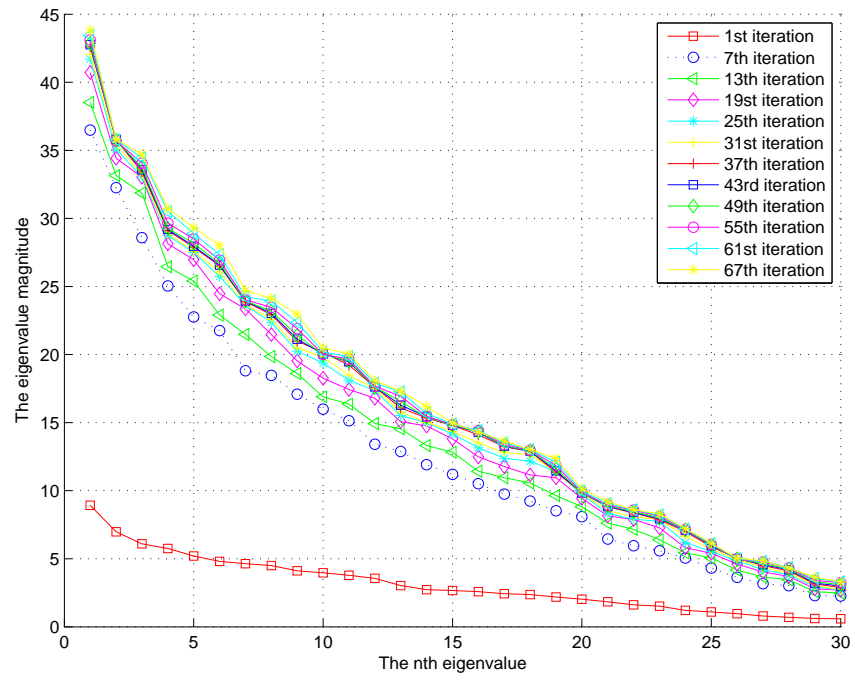


Figure 4.20: The distribution of the first fifty eigenvalues of the weighted adjacency matrix \mathcal{A}_S in a feature correspondence matching example at iterations 1, 7, 13, 19, 25, 31, 37, 43, 49, 55, and 61. The data-sets involved are the hand sequences 09 & 11.

Chapter 5

Conclusions

The general objective in this thesis is to obtain the behaviour information of moving objects in image frames. Owing to their interesting properties as reviewed in Chapter 2, kernel methods and graph spectral methods are of interest. For this purpose, first a kernelized spectral graph matching algorithm was developed for rigid and articulated feature point-sets. Then in a second effort, a new probabilistic relaxation labelling algorithm was developed, also in the spirit of incorporating the kernel advantages. As presented in previous chapters, these algorithms give encouraging results in feature correspondence matching and data classification. In this chapter we conclude the contributions and analyze their strengths and weaknesses. Further, we discuss possible improvements of the algorithms and suggest potential future extension for more challenging correspondence matching, consistent labelling, and data clustering tasks.

5.1 Correspondence matching

The novel parts of the correspondence matching algorithm we developed in this thesis were the use of kernel functions on graphs, and the combination of label consistency constraints into the proximity matrix for articulated motion. The development also showed the links between the kernel PCA and early graph spec-

tra based point matching approaches. Recently kernel methods are gaining an increasing interest in various areas. The definition of kernels make it flexible in choosing kernel functions for different purposes. This provides us with a broad range of revitalising the existing graph spectral feature correspondence matching approaches. For example, according to the problems in this thesis, the Gaussian and polynomial kernels are used to construct the Gram matrix. It would also be interesting to use other kernel functions when the data come from other resources. In fact, the applications of the algorithm are not restricted in the computer vision area.

The first weakness or limitation in the approach is that for the articulated case, the performance also depends on the initial label probabilities. Also it still remains heuristic in deciding the compatibility coefficients for each label pair, although the performance in our experiments is encouraging with current compatibility values. In the experiments, we have an assumption that the relative motion of two articulated components is not significant. This is represented in the initial label probabilities, and also it is assumed that we always have the confidence that the relative positions are not changed. Giving these assumptions, the labelling process may be adapted to a semi-supervised one, and a random walk process may be incorporated to improve the result. Another weakness of this algorithm is that when using the Gaussian kernel, the value of the width parameter σ is not easy to choose. In Chapter 3 we used an heuristic function to compute the parameter automatically. But this function only gave acceptable results. Better matching results were found using other parameter values. Thus a more sophisticated method is required for choosing this value.

5.2 Consistent labelling

In this development, a new formulation of probabilistic relaxation labelling is developed using the theory of diffusion processes on graphs. While many devel-

opments have used discrete time random walks for the task of data classification, many real world problems require a continuous setting. Currently not many developments and applications in the computer vision and pattern recognition area are based on the continuous time Markov processes. Although the diffusion process has been used in image smoothing and enhancing applications in terms of a discretized diffusion operator since the early 1990s (see, for example, refs. [97, 48]), it has not been used for finding feature correspondences between image frames. Our development of using a diffusion based probabilistic relaxation for feature correspondence matching is thus the first attempt. The application of our development to the problem of data classification is also a new one in the sense that a diffusion process model is used on a graph setting with continuous state space assumption, and multiple labels are assigned to the objects in the same time. Other data classification developments which use discrete or continuous time random walks mostly requires either at least one known label for each class, that is, they are in the semi-supervised manner, or assume that there are only two labels.

One weak point of our development is that it requires the computation of matrix exponentiation, and the graph construction makes our Gram matrix grows quickly when the number of data points increases. To improve the weakness, first of all more efficient method for computing the matrix exponentiation may be applied. In Chapter 4, we have mentioned using the Krylov subspace approximation methods. We also briefly introduced the potential use of the Nyström method to obtain the approximate eigenvectors of the matrix. It is of interest to implement these approximation methods in our algorithm. Another interesting attempt may be to adopt regular graphs to represent both the object-set and label-set. In this way, the computational demand can be dramatically reduced since now the algorithm requires the eigen-decomposition of only the small matrices, as is discussed by Chung and Yau in [26], and by Ellis in [41].

The discussion of the diffusion process running backwards in time in Chapter

4 is rather rough. It would be interesting to have a more sophisticated analysis of its behaviour and computation.

5.3 Future work

Based on the analysis above and our experiments in previous chapters, it is interesting to extend the current work in this thesis to a broader application in computer vision and pattern recognition. One intuitive extension would be to apply the newly developed method to the problem of image segmentation. Give that the number of segments of an image is known, and some initial confidence about which segment each image pixel belongs to is at hand, reasonable segmentation results shall be expected by using our probabilistic relaxation algorithm. To cope with the large number of data points in images, a coarse-to-fine hierarchical approach can be incorporated. Another interesting application would be to apply our method to the problem of speech recognition. One example is to apply the method for classifying different speakers in segments of speech data. In real life when we hear a sample of a speech or a song record, we will have some confidence of which person the speaker is. This initial confidence can be interpreted into our initial label probabilities. Relationship between these different speakers can be obtained either from labeled data, or from current data, or even from experience.

While it is interesting to use approximation methods for computing the matrix exponentiation discussed previously, it is also interesting to change the graph construction and to remain using the diffusion process theory to improve the efficiency. Rather than putting all the object-label assignments into the graph node-set, we may use only the object-set to build the graph. A q -state Potts model [138] may be also included, where q is the number of possible labels, to accommodate all the possible label assignments. A Markov process is then to be constructed from the graph, allowing the diffusion of initial label confidences and jumps be-

tween different states.

Gaussian processes are special cases of continuous time Markov processes in which the components of the state probability distribution vector have a joint multivariate Gaussian distribution [85, 135]. The process can be parameterized by a mean function and a covariance function, which are the counterparts of the drift and diffusion coefficients in the diffusion equation in Equation 4.7. Under mild conditions Gaussian processes are good approximations to real world random processes, and object tracking assuming Gaussian distribution has already had successful developments, e.g., the Kalman filter [76]. Research work on defining the drift and covariance coefficients from the given data exist in literature, as introduced in the work of Williams [135], and the work in economics by Hansen and Scheinkman [59]. As a standard diffusion process has a Gaussian nature, it is also of interest to use Gaussian process to model our problem. When the Gaussian assumption does not exist, more sophisticated models such as the particle filters [72, 7] are of interest. These provide us interesting directions to work with.

Another interest is to improve the underlying framework of the consistent labelling method using more general stochastic diffusion equations and the geometric properties of manifolds. Several researchers have already proved that under mild conditions graphs and the graph Laplacian are capable of representing the underlying Riemannian manifold, and the Laplace-Beltrami operator with a suitably defined Riemannian metric, respectively [11]. The geodesic distance on Riemannian manifold captures the structure of the data, and governs the diffusion. In [105] Rosenberg has given a thorough analysis of the short time and long time behaviour of diffusion processes on Riemannian manifold. Lafferty and Lebanon [82] have also proposed a method of learning from data by combining statistical theory with Riemannian manifold. Sochen has also developed diffusion based methods on Riemannian manifolds based on short time diffusion behaviour [119]. These provide us with sufficient theoretical basis for developing

improved probabilistic relaxation labelling methods for a wide range of data-sets and for broad applications. Thus it might be of interest to combine stochastic differential equations and Riemannian manifold concept together to develop new confidence collection and propagation algorithms.

Bibliography

- [1] A. Agarwal and B. Triggs. Tracking articulated motion using a mixture of autoregressive models. In *European Conference on Computer Vision*, pages III 54–65, May 2004.
- [2] J. K. Aggarwal, Q. Cai, W. Liao, and B. Sabata. Articulated and elastic non-rigid motion: A review. In *Proceeding of The Workshop on Motion Of Non-rigid And Articulated Objects*, Austin Texas, Nov 1994.
- [3] J. K. Aggarwal, Q. Cai, W. Liao, and B. Sabata. Nonrigid motion analysis: Articulated and elastic motion. *Computer Vision and Image Understanding*, 70(2):142–156, May 1998.
- [4] A. R. Ahmadyfard and J. Kittler. Using relaxation technique for region-based object recognition. *Image and Vision Computing*, 20:769–781, 2002.
- [5] Y. Aït-Sahalia, L. Hansen, and J. Scheinkman. Operator methods for continuous-time Markov processes, 2004.
- [6] C. J. Alpert and S.-Z. Yao. Spectral partitioning: The more eigenvectors, the better. In *Proc. ACM/IEEE Design Automation conf.*, 1995.
- [7] M. Arulampalam, S. Maskell, N. Gordon, and T. Clapp. A tutorial on particle filters for nonlinear/non-gaussian bayesian tracking. *IEEE Transactions on signal Processing*, 50(2):174–188, 2002.

- [8] F. R. Bach and M. I. Jordan. Learning spectral clustering. In *Advances in Neural Information Processing Systems*, volume 16. The MIT Press, 2004.
- [9] D. H. Ballard and C. M. Brown. *Computer Vision*. Prentice-Hall, Inc., Englewood Cliffs, 1982.
- [10] S. T. Barnard and M. A. Fischler. Computational stereo. *Computing Surveys*, 14(4):553–572, 1982.
- [11] M. Belkin and P. Niyogi. Laplacian eigenmaps and spectral techniques for embedding and clustering. In T. Dietterich, S. Becker, and Z. Ghahramani, editors, *Advances in Neural Information Processing Systems*, volume 14. MIT Press, 2002.
- [12] M. Belkin and P. Niyogi. Using manifold structure for partially labelled classification. In *Advances in Neural Information Processing Systems NIPS'2002*, volume 15. MIT Press, 2002.
- [13] A. Ben-Hur, D. Horn, H. T. Siegelmann, and V. Vapnik. Support vector clustering. *Journal of Machine Learning Research*, 2:125–137, 2001.
- [14] Y. Bengio, O. Delalleau, N. L. Roux, J.-F. Paiement, P. Vincent, and M. Ouimet. Learning eigenfunctions links spectral embedding and kernel PCA. *Neural Computation*, 16:2197–2219, 2004.
- [15] M. Blatt, S. Wiseman, and E. Domany. Data clustering using a model granular magnet. *Neural Computation*, 9:1805–1842, 1997.
- [16] M. E. Brand and K. Huang. A unifying theorem for spectral embedding and clustering. Technical Report TR2002-42, MERL, 2002.
- [17] N. Briggs. *Algebraic Graph Theory*. Cambridge university press, 1974.
- [18] T. Caelli and S. Kosinov. An eigenspace projection clustering method for inexact graph matching. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(4):515–519, 2004.

- [19] A. S. Carasso, J. G. Sanderson, and J. M. Hyman. Digital removal of random media image degradations by solving the diffusion equation backwards in time. *SIAM Journal on Numerical Analysis*, 15(2):344–367, 1978.
- [20] M. Carcassoni and E. R. Hancock. Point pattern matching with robust spectral correspondence. In *IEEE computer society conference on computer vision and pattern recognition*, volume I, pages 649–655. IEEE Computer Society Press, 2000.
- [21] M. Carcassoni and E. R. Hancock. Correspondence matching with modal clusters. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(12), 2003.
- [22] M. Carcassoni and E. R. Hancock. Spectral correspondence for point pattern matching. *Pattern Recognition*, 36:193–204, 2003.
- [23] M. A. Carreira-Perpiñán and R. S. Zemel. Proximity graphs for clustering and manifold learning. In L. K. Saul, Y. Weiss, and L. Bottou, editors, *Advances in Neural Information Processing Systems 17*, pages 225–232, Cambridge, MA, 2005. MIT Press.
- [24] O. Chapelle, J. Weston, and B. Schölkopf. Cluster kernels for semi-supervised learning. In S. Becker, S. Thrun, and K. Obermayer, editors, *Advances in Neural Information Processing Systems*, volume 15. The MIT Press, 2003.
- [25] W. Christmas, J. Kittler, and M. Petrou. Structural matching in computer vision using probabilistic relaxation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17:749–764, 1995.
- [26] F. Chung and S.-T. Yau. Discrete Green’s functions. *J. Combinatorial Theory*, 91:191–214, 2000.

- [27] F. R. K. Chung. *Spectral Graph Theory*. Number 92 in Regional Conference Series in Mathematics. American Mathematical Society, 1997.
- [28] R. Coifman, S. Lafon, A. Lee, M. Maggioni, B. Naddler, F. Warner, and S. Zucker. Geometric diffusions as a tool for harmonic analysis and structure definition of data: Diffusion maps. *PNAS*, 102(21):7426–7431, 2005.
- [29] T. F. Cootes, C. J. Taylor, D. H. Cooper, and J. Graham. Training models of shape from sets of examples. In *BMVC*, 1992.
- [30] T. F. Cootes, C. J. Taylor, D. H. Cooper, and J. Graham. Active shape models – their training and application. *Computer Vision and Image Understanding*, 61(1):38–59, January 1995.
- [31] J. Costeria and T. Kanade. A multi-body factorization method for motion analysis. In *Proc. Fifth Int. Conf. Computer Vision*, pages 1071–1076, MA,USA, 1995.
- [32] D. Cox and H. D. Miller. *The Theory of Stochastic Processes*. Methuen & Co. Ltd., London, 1965.
- [33] T. F. Cox and M. A. A. Cox. *Multidimensional Scaling*. Chapman and Hall, London, 1994.
- [34] N. Cristianini, J. Shawe-Taylor, and J. Kandola. Spectral kernel methods for clustering. In *Proceedings of the Neural Information Processing Systems*, pages 649–655, 2001.
- [35] A. D. J. Cross and E. R. Hancock. Graph matching with a dual-step EM algorithm. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(11):1236–1253, 1998.
- [36] F. Dellaert, S. M. Seitz, C. E. Thorpe, and S. Thrun. Structure from motion without correspondence. In *Proceedings of CVPR*, 2000.

- [37] C. B. D.J. Newman, S. Hettich and C. Merz. UCI repository of machine learning databases, 1998.
- [38] W. Donath and A. Hoffman. Algorithms for partitioning of graphs and computer logic based on eigenvectors of connection matrices. *IBM Technical Disclosure Bulletin*, 15:938–944, 1972.
- [39] W. Donath and A. Hoffman. Lower bounds for the partitioning of graphs. *IBM J. RES. DEVELOP.*, 17:420–425, 1973.
- [40] R. O. Duda and P. E. Hart. *Pattern classification and scene analysis*. A Wiley-Interscience Publication, New York, Wiley, 1973.
- [41] R. B. Ellis. Discrete Green’s functions for products of regular graphs. *ArXiv Mathematics e-prints*, Sept. 2003.
- [42] P. Faber. A theoretical framework for relaxation processes in pattern recognition: Application to robust nonparametric contour generalization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(8):1021–1027, 2003.
- [43] O. Faugeras and M. Berthod. Improving consistency and reducing ambiguity in stochastic labeling: An optimization approach. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 3(4):412–423, 1981.
- [44] M. Fiedler. Algebraic connectivity of graphs. *Czechoslovak Mathematical Journal*, 23(98):298–305, 1973.
- [45] A. Finch, R. C. Wilson, and E. R. Hancock. Matching Delaunay Graphs. *Pattern Recognition*, 30(1):123–140, 1997.
- [46] I. Fischer and J. Poland. New methods for spectral clustering. Technical Report IDSIA-12-04, IDSIA, 2004.

- [47] S. Geman and D. Geman. Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 6(6), 1984.
- [48] G. Gilboa, N. Sochen, and Y. Y. Zeevi. Forward-and-backward diffusion processes for adaptive image enhancement and denoising. *IEEE Transactions on Image Processing*, 11(7):689–703, 2002.
- [49] S. Gold and A. Rangarajan. A graduated assignment algorithm for graph matching. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(4):377–388, Apr. 1996.
- [50] G. H. Golub and C. F. van Loan. *Matrix Computations*. The John Hopkins University Press, 1996.
- [51] N. J. Gordon, D. J. Salmond, and A. F. M. Smith. Novel approach to non-linear/non-Gaussian Bayesian state estimation. *IEE-Proceedings-F*, 140(2), 1993.
- [52] L. Grady. Random walks for image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(11), 2006.
- [53] S. Guattery and G. L. Miller. On the quality of spectral separators. *SIAM Journal on Matrix Analysis and Applications*, 19(3):701–719, 1998.
- [54] L. Hagen and A. Kahng. A new approach to effective circuit clustering. In *Proceedings IEEE Conference on Computer-Aided Design*, pages 422–427, 1992.
- [55] K. M. Hall. An r-dimensional quadratic placement algorithm. *Management Science*, 17(3):219–229, Nov. 1970.
- [56] J. Ham, D. D. Lee, S. Mika, and B. Schölkopf. A kernel view of the dimensionality reduction of manifolds. In *Proceedings of the twenty-first international conference on machine learning*, page 47, 2004.

- [57] E. R. Hancock and J. Kittler. Discrete relaxation. *Pattern Recognition*, 23:711–733, 1990.
- [58] E. R. Hancock and J. Kittler. Edge-labeling using dictionary based relaxation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12:165–181, 1990.
- [59] L. P. Hansen and J. A. Scheinkman. Back to the future: Generating moment implications for continuous-time Markov processes. *Econometrica*, 63(4):767–804, 1995.
- [60] M. Hansen and W. Higgins. Watershed-driven relaxation labelling for image segmentation. In *IEEE International Conference on Image Processing*, volume 3, pages 460–464, 1994.
- [61] M. Hansen and W. Higgins. Relaxation methods for supervised image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(9):949–962, 1997.
- [62] R. M. Haralick. An interpretation for probabilistic relaxation (processing mechanisms in image analysis). *Computer Vision, Graphics, and Image Processing*, 22:388–395, 1983.
- [63] D. Haussler. Convolution kernels on discrete structures. Technical Report UCSC-CRL-99-10, Department of Computer Science, University of California at Santa Cruz, 1999.
- [64] M. Hein, J.-Y. Audibert, and U. von Luxburg. From graphs to manifolds — weak and strong pointwise consistency of graph Laplacians. In *Proceedings of the 18th Conference on Learning Theory (COLT)*, pages 470–485, 2005.
- [65] G. E. Hinton, C. K. I. Williams, and M. Revow. Adaptive elastic models for hand-printed character recognition. In S. H. J.E. Moody and R. Lipp-

man, editors, *Advances in Neural Information Processing*, volume 4. Morgan Kaufmann, 1992.

- [66] M. Hochbruck and C. Lubich. On krylov subspace approximations to the matrix exponential operator. *SIAM Journal on Numerical Analysis*, 34(5):1911–1925, 1997.
- [67] T. Hofmann and J. Buhmann. Pairwise data clustering by deterministic annealing. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(1), 1997.
- [68] C. C. Holmes and N. M. Adams. A probabilistic nearest neighbour method for statistical pattern recognition. *J. R. Statist. Soc.*, 64(Part 2):295–306, 2002.
- [69] W. Hu, T. Tan, L. Wang, and S. Maybank. A survey on visual surveillance of object motion and behaviors. *IEEE Transactions on Systems, Man, and Cybernetics – Part C: Applications and Reviews*, 34(3), 2004.
- [70] B. D. Hughes, M. F. Shiesinger, and E. W. Montroll. Random walks with self-similar clusters. *Proc. Natl. Acad. Sci. USA*, 78(6):3287–3291, 1981.
- [71] R. A. Hummel and S. W. Zucker. On the foundations of relaxation labeling processes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1:PAMI-5:267, 1983.
- [72] M. Isard and A. Blake. Contour tracking by stochastic propagation of conditional density. In *Proc. European conf. on Computer Vision*, volume 1, pages 343–356, Cambridge UK, 1996.
- [73] A. Jain and R. Dubes. *Algorithms for Clustering Data*. Prentice Hall, 1988.
- [74] A. K. Jain, M. Murty, and P. Flynn. Data clustering: A review. *ACM Computing Surveys*, 31(3):264–323, 1999.

- [75] S. X. Ju, M. J. Black, and Y. Yacoob. Cardboard people: A parameterized model of articulated image motion. In *2nd Int. Conf. on Automatic Face- and Gesture-Recognition*, pages 38–44, 1996.
- [76] R. E. Kalman. A new approach to linear filtering and prediction problems. *Transactions of the ASME – Journal of Basic Engineering*, 82(Series D):35–45, 1960.
- [77] R. Kannan, S. Vempala, and A. Vetta. On clusterings: Good, bad and spectral. *Journal of the ACM*, 51(3):497–515, 2004.
- [78] S. Karlin and H. M. Taylor. *A first course in stochastic processes*. Academic Press, 2nd edition, 1975.
- [79] M. Kass, A. Witkin, and D. Terzopoulos. Snakes: Active contour models. *International Journal of Computer Vision*, 1(4):321–331, 1988.
- [80] J. Kittler and E. R. Hancock. Combining evidence in probabilistic relaxation. *International Journal Of Pattern Recognition And Artificial Inteligence*, 3(1):29–51, 1989.
- [81] R. I. Kondor and J. Lafferty. Diffusion kernels on graphs and other discrete input spaces. In *Proceedings of the International Conference on Machine Learning (ICML)*, pages 315–322, 2002.
- [82] J. Lafferty and G. Lebanon. Diffusion kernels on statistical manifolds. Technical Report CMU-CS-04-101, School of Computer Science, Carnegie Mello University, Pittsburgh, PA 15213, USA, 2004.
- [83] B. Luo and E. R. Hancock. Matching point-sets using Procrustes alignment and the EM algorithm. In *British machine vision conference*, 1999.
- [84] B. Luo and E. R. Hancock. Structural matching using EM algorithm and singular value decomposition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23:1120–1136, 2001.

- [85] D. J. C. Mackay. Gaussian processes: A replacement for neural networks? *Tutorial at the Tenth Annual Conference on Neural Information Processing Systems*, 1997. Available from <http://wol.ra.phy.cam.ac.uk/pub/mackay/>.
- [86] M. Meilă and J. Shi. Learning segmentation by random walks. In *NIPS*, pages 873–879, 2000.
- [87] S. Mika, G. Ratsch, J. Weston, B. Schölkopf, and K. Müller. Fisher discriminant analysis with kernels. In *Proceedings of IEEE Neural Networks for Signal Processing*, pages 41–48, 1999.
- [88] S. Mika, B. Schölkopf, A. Smola, K. Müller, M. Scholz, and G. Rätsch. Kernel PCA and de-noising in feature spaces. *Advances in Neural Information Processing Systems 11*, 1998.
- [89] C. Moler and C. van Loan. Nineteen dubious ways to compute the exponential of a matrix, twenty-five years later. *SIAM Review*, 45(1):3–49, 2003.
- [90] A. Y. Ng, M. I. Jordan, and Y. Weiss. On spectral clustering: analysis and an algorithm. In *Advances in Neural Information Processing Systems*, volume 14, 2001.
- [91] K. Okuma, A. Taleghani, N. de Freitas, J. Little, and D. Lowe. A boosted particle filter: Multitarget detection and tracking. In *ECCV*, pages 28–39, 2004.
- [92] G. Pajares, J. M. de la Cruz, and J. A. López-Orozco. Relaxation labeling in stereo image matching. *Pattern Recognition*, 33:53–68, 2000.
- [93] J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann, 1988.
- [94] M. Pelillo. Relaxation labeling networks for the maximum clique problem. *Journal of Artificial Neural Networks*, 2(4):313–328, 1995.

- [95] M. Pelillo and M. Refice. Learning compatibility coefficients for relaxation labeling processes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16(9):933–945, 1994.
- [96] P. Perona and W. T. Freeman. A factorization approach to grouping. In *ECCV*, pages 655–670, 1998.
- [97] P. Perona and J. Malik. Scale-space and edge detection using anisotropic diffusion. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(7), Jul 1990.
- [98] C. Poelman and T. Kanade. A paraperspective factorization method for shape and motion recovery. *IEEE Trans. on Patt. Anal. and Mach. Intell.*, 19(3):206–218, 1997.
- [99] A. Pothen, H. Simon, and K.-P. Liou. Partitioning sparse matrices with eigenvectors of graphs. *SIAM Journal on Matrix Analysis and Applications*, 11(3):430–452, 1990.
- [100] H. Qiu and E. R. Hancock. Graph matching using commute time spanning trees. In *18th International Conference on Pattern Recognition*, pages 1224–1227, 2006.
- [101] A. Rangarajan. SelfAnnealing: Unifying deterministic annealing and relaxation labelling. In *First International Workshop on Energy Minimization Methods in Computer Vision and Pattern Recognition*, pages 229–244, 1997.
- [102] R.C.Wilson, A. Cross, and E. Hancock. Sensitivity analysis for structural matching. In *Proceedings of the 13th International Conference on Pattern Recognition*, pages 62–66, 1996.
- [103] B. D. Ripley. *Stochastic simulation*. John Wiley & Sons, Inc., 1987.
- [104] A. Robles-Kelly and E. R. Hancock. String edit distance, random walks and graph matching. In *S+SSPR*, pages 104–112, 2002.

- [105] S. Rosenberg. *The Laplacian on a Riemannian manifold :an introduction to analysis on manifolds*. Cambridge University Press, 1997.
- [106] A. Rosenfeld, R. Hummel, and S. Zucker. Scene labeling by relaxation operations. *IEEE Trans. Systems. Man and Cybernetics*, 6:420–433, June 1976.
- [107] F. Schaffalitzky and A. Zisserman. Viewpoint invariant texture matching and wide baseline stereo. In *International Conference on Computer Vision*, page 636, 2001.
- [108] B. Schölkopf and A. J. Smola. *Learning with Kernels*. MIT Press, 2001.
- [109] B. Schölkopf, A. J. Smola, and K. R. Müller. Nonlinear component analysis as a kernel eigenvalue problem. *Neural Computation*, 10:1299–1319, 1998.
- [110] S. Sclaroff and A. Pentland. Modal matching for correspondence and recognition. *IEEE Tran. Pattern Analysis and Machine Intelligence*, 17, 1995.
- [111] G. L. Scott and H. C. Longuet-Higgins. Feature grouping by relocalisation of eigenvectors of the proximity matrix. In *British Machine Vision Conference*, 1990.
- [112] G. L. Scott and H. C. Longuet-Higgins. An algorithm for associating the features of two images. *Proc. Royal Soc. London*, 244:21–26, 1991.
- [113] A. J. Seary and W. D. Richards. Partitioning networks by eigenvectors. In *Proceedings of the International Conference on Social Networks*, volume 1: Methodology, pages 47–58, 1995.
- [114] L. S. Shapiro and J. M. Brady. Feature-based correspondence – An eigenvector approach. *Image and Vision Computing*, 10:283–288, 1992.
- [115] J. Shawe-Taylor and N. Cristianini. *Kernel methods for pattern analysis*. Cambridge University Press, 2004.

- [116] J. Shi and J. Malik. Normalized cuts and image segmentation. In *Proc. Computer Vision and Pattern Recognition*, 1997.
- [117] A. Singer. From graph to manifold Laplacian: The convergence rate. *Applied and Computational Harmonic Analysis*, 21:128–134, 2006.
- [118] A. J. Smola and R. Kondor. Kernels and regularization on graphs. In M. Warmuth and B. Schölkopf, editors, *Proceedings of the Annual Conference on Computational Learning Theory*, 2003.
- [119] N. Sochen. Stochastic processes in vision: From langevin to beltrami. In *International Conference on Computer Vision*, 2001.
- [120] D. Spielman and S.-H. Teng. Spectral partitioning works: Planar graphs and finite element meshes. In *IEEE Symposium on Foundations of Computer Science*, pages 96–105. IEEE Computer Society Press, 1996.
- [121] D. W. Stroock. *An Introduction to Markov Processes*. Springer, Berlin, 2005.
- [122] E. B. Sudderth, A. T. Ihler, W. T. Freeman, and A. S. Willsky. Nonparametric belief propagation. In *IEEE International Conference Computer Vision and Pattern Recognition*, pages 605–612, 2003.
- [123] M. Szummer and T. Jaakkola. Partially labeled classification with Markov random walks. In *Advances in Neural Information Processing Systems*, volume 14, pages 945–952, 2002.
- [124] N. Tishby and N. Slonim. Data clustering by Markovian relaxation and the information bottleneck method. In *NIPS 13*, 2000.
- [125] C. Tomasi and T. Kanade. Shape and motion from image streams under orthography — A factorization method. Technical Report TR-92-1270, Cornell University, March 1992.

- [126] K. Tsuda and W. Noble. Learning kernels from biological networks by maximizing entropy. *Bioinformatics*, 20:326–333, 2004.
- [127] S. Umeyama. An eigendecomposition approach to weighted graph matching problem. *IEEE Tran. Pattern Analysis and Machine Intelligence*, 10(5), Sep 1988.
- [128] V. N. Vapnik. *Statistical learning theory*. John Wiley & Sons, Inc., 1998.
- [129] J.-P. Vert and M. Kanehisa. Graph-driven feature extraction from microarray data using diffusion kernels and kernel CCA. In S. T. S. Becker and K. Obermayer, editors, *Advances in Neural Information Processing Systems 15*, pages 1425–1432. MIT Press, Cambridge, MA, 2003.
- [130] D. L. Waltz. Generating semantic descriptions from drawings of scenes with shadows. Technical Report 271, MIT AI Lab, 1972.
- [131] Y. Weiss. Interpreting images by propagating Bayesian beliefs. In *ADVANCES IN NEURAL INFORMATION PROCESSING SYSTEMS*, pages 908–914, 1997.
- [132] Y. Weiss. Segmentation using eigenvectors: A unifying view. In *IEEE International Conference on Computer Vision*, pages 975–982, 1999.
- [133] J. Weston, C. Leslie, E. Ie, D. Zhou, A. Elisseeff, and W. Noble. Semi-supervised protein classification using cluster kernels. *Bioinformatics*, 21(15):3241–3247, 2005.
- [134] C. Williams and M. Seeger. Using the Nyström method to speed up kernel machines. In T. Leen, T. Dietterich, and V. Tresp, editors, *Advances in neural information processing systems 13*, pages 682–688. 2001.
- [135] C. K. Williams. Gaussian processes. In M. Arbib, editor, *The Handbook of Brain Theory and Neural Networks*. The MIT Press, second edition, 2002.

- [136] R. C. Wilson and E. R. Hancock. A bayesian compatibility model for graph matching. *Pattern Recognition Letters*, 17:263–276, 1996.
- [137] R. C. Wilson and E. R. Hancock. Structural matching by discrete relaxation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(6):634–648, Jun 1997.
- [138] F. Wu. The Potts model. *Review of Modern Physics*, 54(1):235–268, 1982.
- [139] Z. Wu and R. Leahy. An optimal graph theoretic approach to data clustering: Theory and its application to image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(11):1101–1113, 1993.
- [140] C.-H. Yeang and M. Szummer. Markov random walk representations with continuous distributions. In *Proceedings of the 19th Conference in Uncertainty in Artificial Intelligence*, pages 600–607, 2003.
- [141] L. Yen, D. Vanvyve, F. Wouters, F. Fouss, M. Verleysen, and M. Saerens. Clustering using a random walk based distance measure. In *ESANN*, pages 317–324, 2005.
- [142] K. Yoshida. *Functional Analysis*. Springer-Verlag, 1965.
- [143] S. Yu and J. Shi. Multiclass spectral clustering. In *9th IEEE International Conference on Computer Vision*, pages 313–319, 2003.
- [144] C. Zahn. Graph theoretical methods for detecting and describing gestalt clusters. *IEEE Transactions on Computers*, C-20:68, 1971.
- [145] L. Zelnik-Manor and P. Perona. Self-tuning spectral clustering. In L. K. Saul, Y. Weiss, and L. Bottou, editors, *Advances in Neural Information Processing Systems 17*, pages 1601–1608. MIT Press, Cambridge, MA, 2005.
- [146] Y. Zheng and D. Doermann. Robust point matching for two-dimensional nonrigid shapes. In *Proceedings IEEE Conf. on Computer Vision*, pages 1561–1566, 2005.

- [147] X. Zhu, Z. Ghahramani, and J. Lafferty. Semi-supervised learning using Gaussian fields and harmonic functions. In *ICML*, pages 1561–1566, 2003.
- [148] X. Zhu, J. Kandola, Z. Ghahramani, and J. Lafferty. Nonparametric transformations of graph kernels for semi-supervised learning. In L. K. Saul, Y. Weiss, and L. Bottou, editors, *Advances in Neural Information Processing Systems 17*, pages 1641–1648, Cambridge, MA, 2005. MIT Press.
- [149] S. W. Zucker. Relaxation labeling: 25 years and still iterating. In L. S. Davis, editor, *Foundations of Image Understanding*, pages 289–322. Kluwer Academic Publisher, Boston, 2001.