Graph Matching and Clustering using Spectral Partitions

Huaijun Qiu * Edwin R. Hancock

Department of Computer Science,University of York, York Y010 5DD, UK.

Abstract

Although inexact graph-matching is a problem of potentially exponential complexity, the problem may be simplified by decomposing the graphs to be matched into smaller subgraphs. If this is done, then the process may cast into a hierarchical framework and hence rendered suitable for parallel computation. In this paper we describe a spectral method which can be used to partition graphs into non-overlapping subgraphs. In particular, we demonstrate how the Fiedler-vector of the Laplacian matrix can be used to decompose graphs into non-overlapping neighbourhoods that can be used for the purposes of both matching and clustering.

Key words: Inexact graph matching; graph simplification; graph partition; graph spectral clustering;Fiedler vector.

1 Introduction

Graph partitioning is concerned with grouping the vertices of a connected graph into subsets so as to minimize the total cut weight [1]. The process is of central importance in electronic circuit design, map coloring and scheduling [2]. However, in this paper we are interested in the process since it provides a means by which the inexact graph-matching problem may be decomposed into a series of simpler subgraph matching problems. As demonstrated by Messmer and Bunke [3], error-tolerant graph matching can be simplified using decomposition methods and reduced to a problem of subgraph indexing. Our aim

* Corresponding author: Tel +44 1904 43 3381; Fax +44 1904 432767.
Email addresses: jun@cs.york.ac.uk (Huaijun Qiu), erh@cs.york.ac.uk (Edwin R. Hancock).
URL: http://www-users.cs.york.ac.uk/ erh/ (Edwin R. Hancock).

in this paper is to explore whether spectral methods can be used to partition graphs in a stable manner for the purposes of matching by decomposition.

Recently, there has been increased interest in the use of spectral graph theory for characterising the global structural properties of graphs. Spectral graph theory aims to summarise the structural properties of graphs using the eigenvectors of the adjacency matrix or the Laplacian matrix [4]. There are several examples of the application of spectral matching methods for grouping and matching in the computer vision literature. For instance, Umeyama has shown how graphs of the same size can be matched by performing singular value decomposition on the adjacency matrices [5]. Here the permutation matrix that brings the nodes of the graphs into correspondence is found by taking the outer product of the matrices of left eigenvectors for the two graphs. In related work Shapiro and Brady [6] have shown how to locate feature correspondences using the eigenvectors of a point-proximity weight matrix. These two methods fail when the graphs being matched contain different numbers of nodes. However, this problem can be overcome by using the apparatus of the EM algorithm [7,8]. More recently, Shokoufandeh, Dickinson, Siddiqi and Zucker [9] have shown how graphs can be retrieved efficiently using an indexing mechanism that maps the topological structure of shock-trees into a low-dimensional vector space. Here the topological structure is encoded by exploiting the interleaving property of the eigenvalues.

One of the most important spectral attributes of a graph is the Fiedler vector, i.e. the eigenvector associated with the second smallest eigenvalue of the Laplacian matrix. In a useful review, Mohar [10] has summarized some important applications of Laplace eigenvalues such as the max-cut problem, semidefinite programming and steady state random walks on Markov chains. More recently, Haemers [11] has explored the use of interlacing properties for the eigenvalues and has shown how these relate to the chromatic number, the diameter and the bandwidth of graphs. In the computer vision literature, Shi and Malik [12] have used the Fiedler vector to develop a recursive partition scheme and have applied this to image grouping and segmentation. The Fiedler vector may also be used for the Minimum Linear Arrangement problem(MinLA) which involves placing the nodes of a graph in a serial order which is suitable for the purposes of visualisation [13].

An extension of the minimum linear arrangement problem is the seriation problem which involves finding a serial ordering of the nodes, which maximally preserves the edge connectivity. This is clearly a problem of exponential complexity. As a result approximate solution methods have been employed. These involve casting the problem in an optimization setting. Hence techniques such as simulated annealing and mean field annealing have been applied to the problem. It may also be formulated using semidefinite programming, which is a technique closely akin to spectral graph theory since it relies on eigenvec-

tor methods. However, recently a graph-spectral solution has been found to the problem. Atkins, Boman and Hendrikson [14] have shown how to use the Fiedler eigenvector of the Laplacian matrix to sequence relational data. The method has been successfully applied to the consecutive ones problem and a number of DNA sequencing tasks. There is an obvious parallel between this method and steady state random walks on graphs, which can be located using the leading eigenvector of the Markov chain transition probability matrix. However, in the case of a random walk the path is not guaranteed to encourage edge connectivity. The spectral seriation method, on the other hand, does impose edge connectivity constraints on the recovered path.

The aim in this paper is to consider whether the partitions delivered by the Fiedler vector can be used to simplify the graph-matching problem. We focus on two problems. The first of these is to use the Fiedler vector to decompose graphs by partitioning them into structural units. Our aim is to explore whether the partitions are stable under structural error, and in particular whether they can be used for the purposes of graph-matching. The second problem studied is whether the partitions can be used to simplify the graphs in a hierarchical manner. Here we construct a graph in which the nodes are the partitions and the edges indicate whether the partitions are connected by edges in the original graph. This spectral construction can be applied recursively to provide a hierarchy of simplified graphs. We show that the simplified graphs can be used for efficient and reliable clustering.

The outline of the remainder of the paper is as follows. Section 2 defines the graph Laplacian and explains the meaning of the Fiedler vector. In Section 3 we describe how the Fiedler vector can be used to partition graphs into supercliques. Section 4 outlines how the supercliques can be used for graph-matching using ideas drawn from string edit distance. In Section 5 we explain how the supercliques can be used for the hierarchical simplification of graphs. Experiments are presented in Section 6. Finally, Section 7 offers some conclusions.

2 Laplacian Matrix and Fiedler Vector

Consider the unweighted graph $G = (V, E)$ where V is the set of nodes and E is the set of edges. The adjacency matrix of the graph is A, and has elements

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

The weighted adjacency matrix is denoted by W.

The degree matrix of the graph is the diagonal matrix $D = diag(deg(i); i \in V)$ where the degree is the row-sum of the adjacency matrix $deg(i) = \sum_{j \in V} A(i,j)$. With these ingredients the Laplacian matrix $L = D - A$ has elements

$$L(i,j) = \begin{cases} \sum_{\langle i,k \rangle \in E} A(i,k) & \text{if } i = j \\ -A(i,j) & \text{if } i \neq j \text{ and } (i,j) \in E \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

The Laplacian matrix has a number of important properties. It is symmetric and positive semidefinite. The eigenvector $(1, 1, \ldots, 1)^T$ corresponds to the trivial zero eigenvalue. If the graph is connected then all other eigenvalues are positive and the smallest eigenvalue is a simple one, which means that the number of connected components of the graph is equal to the multiplicity of the smallest eigenvalue. If we arrange all the eigenvalues from the smallest to the largest i.e. $0 \leq \lambda_1 \leq \lambda_2 \ldots \leq \lambda_n$, the most important are the largest eigenvalue λ_{max} and the second smallest eigenvalue λ_2, whose corresponding eigenvector is referred to as the *Fiedler Vector* [15].

Our aim is to decompose the graph into non-overlapping neighbourhoods using a path-based seriation method. The aim is to find a path sequence for the nodes in the graph using a permutation π. The permutation gives the order of the nodes in the sequence. The sequence is such that the elements of the edge weight matrix W decrease as the path is traversed. Hence, if $\pi(i) < \pi(j) < \pi(k)$, then $W(i,j) > W(i,k)$ and $W(j,k) > W(i,k)$. This behaviour can be captured using the penalty function

$$g(\pi) = \sum_{i=1}^{|V|} \sum_{j=1}^{|V|} W(i,j)(\pi(i) - \pi(j))^2$$

By minimizing $g(\pi)$ it is possible to find the permutation that minimizes the difference in edge weight between adjacent nodes in the path, and this in turn sorts the edge weights into magnitude order. Unfortunately, minimizing $g(\pi)$ is potentially NP complete due to the combinatorial nature of the discrete permutation π. To overcome this problem, a relaxed solution is sought that approximates the structure of $g(\pi)$ using a vector $\vec{x} = (x_1, x_2, ..., x_{|V|})$ of continuous variables x_i. Hence, the penalty function considered is

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

The value of $\hat{g}(\vec{x})$ does not change if a constant amount is added to each of the components x_i. Hence, the minimization problem must be subject to

constraints on the components of the vector $\vec{x}$. The constraints are that

$$\sum_{i=1}^{|V|} x_i^2 = 1 \quad \text{and} \quad \sum_{i=1}^{|V|} x_i = 0 \tag{3}$$

The solution to this relaxed problem may be obtained from the Laplacian matrix. If $\vec{e} = (1, 1, 1..., 1)^T$ is the all-ones vector, then the solution to the minimization problem is the vector

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

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

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

3 Graph Partition

The aim in this paper is to use the Fiedler vector to partition graphs into non-overlapping structural units and to use the structural units generated by this decomposition for the purposes of graph-matching and graph-simplification.

3.1 Decomposition

The neighbourhood of the node i consists of its center node, together with its immediate neighbors connected by edges in the graph, i.e., $\hat{N}_i = \{i\} \cup \{u; (i,u) \in E\}$. An illustration is provided in Figure 1, which shows a graph with two of its neighbourhoods highlighted. Hence, each neighbourhood consists of a *center node* and *immediate neighbors* of the center node, i.e. $N_i = \hat{N}_i \setminus \{i\}$.

The problem addressed here is how to partition the graph into a set of non-overlapping neighbourhoods using the node order defined by the Fiedler vector. Our idea is to assign to each node a measure of significance as the centre of a neighbourhood. We then traverse the path defined by the Fielder vector selecting the centre-nodes on the basis of this measure.

We commence by assigning weights to the nodes on the basis of the rank-order of their component in the Fiedler vector. Let $\Gamma =< j_1, j_2, j_3, ..., j_{|V|} >$ be the rank-order of the nodes as defined by the Fiedler vector so that the permutation satisfies the condition $\pi(j_1) < \pi(j_2) < \pi(j_3) < ..... < \pi(j_{|V|}$ and the components of the Fiedler vector follow the condition $x_{j_1} > x_{j_2} > .. > x_{j_{|V|}}$. We assign weights to the nodes based on their rank order in the permutation. The weight assigned to the node $i \in V$ is $w_i = Rank\,(\pi(i))$. With this weighted graph in hand, we can gauge the significance of each node using the following score function:

$$\mathcal{F}_i = \alpha\,(deg(i) + |N_i \cap P|) + \frac{\beta}{w_i} \tag{4}$$

where P is the set of nodes on the perimeter of the graph, and α and β are heuristically set thresholds. The first term depends on the degree of the node and its proximity to the perimeter. Hence, it will sort nodes according to their distance from the perimeter. This will allow us to partition nodes from the outer layer first and then work inwards. The second term ensures that the first ranked nodes in the Fielder vector are visited first.

We use the score function to locate the non-overlapping neighbourhoods of the graph G. We traverse this list until we find a node k_1 which is neither in the perimeter, i.e. $k_1 \notin P$ nor whose score exceeds those of its neighbours, i.e. $\mathcal{F}_{k_1} = \arg\max_{i \in k_1 \cup N_{k_1}} \mathcal{F}_i$. When this condition is satisfied, then the node k_1 together with its neighbours N_{k_1} represent the first neighbourhood. The set of nodes $\hat{N}_{k_1} = k_1 \cup N_{k_1}$ are appended to a list T that tracks the set of nodes assigned to the neighbourhoods. This process is repeated for all the nodes which have not yet been assigned to a neighbourhood i.e. $R = \Gamma - T$. The procedure terminates when all the nodes of the graph have been assigned to non-overlapping neighbourhood. An example is shown at Figure 2. Here the different neighbourhoods are shown in different colours.

4 Matching

Our aim here is to match the graphs using the non-overlapping neighbourhoods detected using the Fiedler vector. To perform the matching we use the edit-distance method of Myers, Wilson and Hancock [16]. In this section for completeness, we review the elements of this method and explain how it may be applied to the neighbourhoods delivered by the Fiedler vector.

Consider a data graph $G_D = (V_D, E_D)$ which is to be matched onto a model graph $G_M = (V_M, E_M)$. The state of correspondence match can be represented by the function $f : V_D \mapsto V_M \cup \{\epsilon\}$ from the node-set of the data graph onto

the node-set of the model graph, where the node-set of the model graph is augmented by adding a NULL label, ϵ, to allow for unmatchable nodes in the data graph. Our score function for the match is the average over the matching probabilities for the set of neighbourhoods of the graph

$$Q_{D,M}(f) = \frac{1}{|V_D|} \sum_{i \in V_D} P(F_i) \tag{5}$$

where $F_i = (f(u_0), f(u_1), ..., f(u_{|N_i^D|})$ denotes the relational image of the neighbourhood N_i^D in G_D under the matching function f.

We use the Bayes rule to compute the matching probability over a set of legal structure-preserving mappings between the data and model graphs. The set of mappings is compiled by considering the cyclic permutations of the peripheral nodes about the centre node of the neighbourhood. The set of feasible mappings generated in this way is denoted by $\Theta_i = \{S\}$ which consists of structure-preserving mapping of the form $S = (s_0, s_1, .., s_u, ..., s_{|N_i^D|})$, where $s_u \in j \cup \{v; (j, v) \in N_j^M\} \cup \epsilon$ is either one of the node-labels drawn from the model graph neighbourhood or the null-label ϵ, and $u \in N_i^D$ is one of the node-labels drawn from the data graph neighbourhood N_i^D.

With the structure preserving mappings to hand we use the Bayes formula to compute the matching probability, $P(F_i)$. This is done by expanding over the set of structure preserving mappings Θ_i in the following manner

$$P(F_i) = \sum_{S \in \Theta_i} P(F_i|S) \cdot P(S) \tag{6}$$

We assume a uniform distribution of probability over the structure preserving mappings and write $P(S) = \frac{1}{|\Theta_i|}$ The conditional matching probability $P(F_i|S)$ is determined by comparing every assigned match $f(u)$ in the configuration F_i with the corresponding item s_u in the structure preserving mapping S.

4.1 Edit Distance

To model the structural differences in the neighbourhoods, we use the Levenshtein or string edit distance [17,18,16]. This models structural error by considering insertions and deletions, in addition to relabelling. In what follows, we work with the set of structure preserving mappings Θ_i^c which contains only cyclic permutations and whose size is therefore equal to $|N_i^D| - 1$.

Let X and Y be two strings of symbols drawn from an alphabet Σ. We wish to convert X to Y via an ordered sequence of operations such that the cost as-

sociated with the sequence is minimal. The original string to string correction algorithm defined *elementary edit operations*, $(a, b) \neq (\epsilon, \epsilon)$ where a and b are symbols from the two strings or the NULL symbol, ϵ. Thus, changing symbol x to y is denoted by (x, y), inserting y is denoted (ϵ, y), and deleting x is denoted (x, ϵ). A sequence of such operations which transforms X into Y is known as an *edit transformation* and denoted $\Delta =< \delta_1, ..., \delta_{|\Delta|} >$. Elementary costs are assigned by an elementary weighting function $\gamma : \Sigma \cup \{\epsilon\} \times \Sigma \cup \{\epsilon\} \mapsto \Re$; the cost of an edit transformation, $C(\Delta)$, is the sum of its elementary costs. The edit distance between X and Y is defined as

$$\mathbf{d}(X, Y) = \min\{C(\Delta) | \Delta \text{ transforms } X \text{ to } Y\} \tag{7}$$

In [19], Marzal and Vidal introduced the notion of an *edit path* which is a sequence of ordered pairs of positions in X and Y such that the path monotonically traverses the edit matrix of x and y from $(0, 0)$ to $(|X|, |Y|)$.

Essentially, the transition from one point in the path to the next is equivalent to an elementary edit operation:$(a, b) \rightarrow (a + 1, b)$ corresponds to deletion of the symbol in X at position a. Similarly, $(a, b) \rightarrow (a, b + 1)$ corresponds to insertion of the symbol at position b in Y. The transition $(a, b) \rightarrow (a + 1, b + 1)$ corresponds to a change from $X(a)$ to $Y(b)$. Thus, the cost of an edit path, $C(P)$, can be determined by summing the elementary weights of the edit operations implied by the path.

$$\mathbf{d}(X, Y) = \min\{C(P | P \textit{ is an edit path from } X \textit{ to } Y)\} \tag{8}$$

4.2 Matching Probabilities

If we replace X and Y by a structure preserving mapping, S_i, and the image of a data graph neighbourhood under the match, F_j, we can see that F_j could have arisen from S through the action of a memoryless error process, statistically independent of position (since the errors that "transformed" S to F_j could have occurred in any order). So we can factorize (6) over the elementary operations implied by the edit path P^*

$$P(F_j | S_i) = \prod_{(f(u),v) \Leftarrow P^*_{F_j, S_i}} P(f(u) | v) \tag{9}$$

where $(f(u), v)$ is an insertion, a deletion, a change or an identity operation implied by the edit path $P^*_{F_j, S_i}$ between the neighbourhood F_j and the unpadded structure preserving mapping S_i. For simplicity, we assume that different edit operations have identical cost. This does not influence the probability because it is the probabilities of the transitions in the path which

contribute to the matching prior not the edit weights themselves, although they will determine the magnitude of the minimum cost.

$$\gamma\left(f\left(u\right),v\right)=\begin{cases}0 \text{ if } \left(f\left(u\right),v\right) \text{ is an identity}\\ 1 \text{ otherwise}\end{cases} \tag{10}$$

So, the probability for the edit operation given to each pair is:

$$P\left(f\left(u\right)|v\right)=\begin{cases}\left(1-P_e\right) \text{ if } \left(f\left(u\right),v\right) \text{ is an identity}\\ P_e \qquad \text{otherwise}\end{cases} \tag{11}$$

If we define the number of nonidentity transformations in the edit path to be $\Psi\left(P^*_{F_j,S_i}\right)$, the matching probability of F_j can be given:

$$P\left(F_j\right)=\frac{K_{N_j^D}}{|\Theta|}\sum_{S_i\in\ \Theta}\exp\left[-K_w\Psi\left(P^*_{F_j,S_i}\right)\right] \tag{12}$$

where $K_{N_j^D}=\left(1-P_e\right)^{\left|N_j^D\right|}$ and $K_w=\ln\frac{\left(1-P_e\right)}{P_e}$.

5 Hierarchical Simplification

The neighbourhoods extracted using the Fiedler vector may also be used to perform hierarchical graph simplification.

5.1 Partition Arrangements

Our simplification process proceeds as follows. We create a new graph in which each neighbourhood $\hat{N}_i=\{i\}\cup\{u;(i,u)\in E\}$ is represented by a node. In practice this is done by eliminating those nodes, which are not the center nodes of the neighbourhoods $N_i=\hat{N}_i\setminus\{i\}$. In other words, we select the center node of each neighbourhood to be the node-set for the next level representation. The node set is given by $\hat{V}=\left\{\hat{N}_1\setminus N_1,\hat{N}_2\setminus N_2,\ldots,\hat{N}_n\setminus N_n\right\}$. Our next step is to construct the edge-set for the simplified graph. We construct an edge between two nodes if there is a common edge contained within their associated neighbourhoods. The condition for the nodes $i\in\hat{V}$ and $j\in\hat{V}$ to form an edge in the simplified graph $\hat{G}=(\hat{V},\hat{E})$ is $(i,j)\in\hat{E}\Rightarrow|\hat{N}_i\cap\hat{N}_j|\geq 2$.

To provide an illustration of the usefulness of the simplifications provided by the Fiedler vector, we focus on the problem of graph clustering. The aim here is to investigate whether the simplified graphs preserve the pattern space distribution of the original graphs. There are a number of ways in which we could undertake this study. However, here we use a simple graph-spectral clustering method, which is in keeping with the overall philosophy of this paper.

Suppose that we aim to cluster the set of M graphs $\{G_1, ...G_k, ...G_M\}$. We commence by performing the spectral decomposition $L_k = \Phi_k \Lambda \Phi_T$ on the Laplacian matrix L_k for the graph indexed k, where $\Lambda_k = diag(\lambda_k^1, \lambda_k^2, ...)$ is the diagonal matrix of eigenvalues and Φ_k is a matrix with eigenvectors as columns. For the graph G_k, we construct a vector $B_k = (\lambda_k^1, \lambda_k^2, \ldots, \lambda_k^m)^T$ from the leading m eigenvalues. We can visualize the distribution of graphs by performing multidimensional scaling (MDS) on the matrix of distances $d_{k1,k2}$ between graphs. This distribution can be computed using either the edit distance technique used in the previous section where $d_{k1,k2} = -\ln Q_{k1,k2}$ or by using the spectral features where $d_{k1,k2} = (B_{k1} - B_{k2})^T (B_{k1} - B_{k2})$.

Multidimensional scaling (MDS) is a procedure which allows data specified in terms of a matrix of pairwise distances to be embedded in a Euclidean space. Here we intend to use the method to embed the graphs extracted from different viewpoints in a low-dimensional space. The pairwise distances d_{k_1,k_2} are used as the elements of an $N \times N$ dissimilarity matrix $\mathbf{R}$, whose elements are defined as follows

$$R_{k_1,k_2} = \begin{cases} d_{k_1,k_2} & \text{if } k_1 \neq k_2 \\ 0 & \text{if } k_1 = k_2 \end{cases} \tag{13}$$

In this paper, we use the classical multidimensional scaling method to embed the graphs in a Euclidean space using the matrix of pairwise dissimilarities $\mathbf{R}$. The first step of MDS is to calculate a matrix $\mathbf{T}$ whose element with row r and column c is given by $T_{rc} = -\frac{1}{2}[d_{rc}^2 - \hat{d}_{r.}^2 - \hat{d}_{.c}^2 + \hat{d}_{..}^2]$, where $\hat{d}_{r.} = \frac{1}{N}\sum_{c=1}^{N} d_{rc}$ is the average dissimilarity value over the r^{th} row, $\hat{d}_{.c}$ is the dissimilarity average value over the c^{th} column and $\hat{d}_{..} = \frac{1}{N^2}\sum_{r=1}^{N}\sum_{c=1}^{N} d_{r,c}$ is the average dissimilarity value over all rows and columns of the dissimilarity matrix $\mathbf{R}$.

We subject the matrix $\mathbf{T}$ to an eigenvector analysis to obtain a matrix of embedding coordinates $\mathbf{Y}$. If the rank of $\mathbf{T}$ is $k, k \leq N$, then we will have k non-zero eigenvalues. We arrange these k non-zero eigenvalues in descending order, i.e. $l_1 \geq l_2 \geq \ldots \geq l_k > 0$. The corresponding ordered eigenvectors are denoted by $\mathbf{u}_i$ where l_i is the ith eigenvalue. The embedding coordinate system for the

graphs is $\mathbf{Y} = [\sqrt{(}l_1)\mathbf{u}_1, \sqrt{l_2}\mathbf{u}_2, \ldots, \sqrt{l_s}\mathbf{u}_s]$, For the graph indexed j, the embedded vector of coordinates is a row of matrix $\mathbf{Y}$, so $\mathbf{y}_j = (Y_{j,1}, Y_{j,2}, ..., Y_{j,s})^T$.

6 Experiments

The aims in this section are threefold. First, we perform a sensitivity study to illustrate that the neighbourhoods delivered by the Fiedler vector form stable structural units for computing edit distance. Second, we show that the neighbourhood partition scheme leads to accurate matches on real world data. Third, we aim to illustrate that the simplification procedure results in a stable distribution of graphs in pattern-space.

6.1 Sensitivity Study

The aim in this part of the experiments is to measure the sensitivity of our new partition-based matching method to structural error. The synthetic graphs we used here are the Delaunay triangulations of randomly generated point-sets. The effects of structural errors are simulated by randomly deleting nodes and re-triangulating the remaining nodes. We have commenced by generating a random graph which has 40 nodes. We have randomly deleted nodes until only 12 nodes remain, i.e. there is 70% corruption. Figure 3 shows a sequence with one node deleted at a time. Coded in different colours are the different supercliques which result from the partitioning of nodes. These remain relatively stable as the nodes are deleted.

We have matched the corrupted graphs to the original graph using the discrete relaxation and edit distance algorithms. We also compare the results with four alternative algorithms. These are the original discrete relaxation method of Wilson and Hancock [8] which is applied to overlapping supercliques, the quadratic assignment method of Gold and Rangarajan [20], the non-quadratic graduated assignment method of Finch, Wilson and Hancock [21], and, the singular value decomposition method of Luo and Hancock [7]. In Figure 4 we show the fraction of correct correspondences as a function of the fraction of nodes deleted.

From this comparison it is clear that our method is robust to structural error. However, it does not perform as well as the original Wilson and Hancock method. One reason for this is that the supercliques delivered by our partitioning method do become unstable under significant corruption.

When used in conjunction with the edit-distance method, the partitions lead

to better results than when used with the dictionary-based discrete relaxation method. This is important since the former method is more computationally efficient than the latter, since the overheads associated with dictionary construction can grow exponentially if dummy nodes need to be inserted.

An example of the set of matches used in this experiment is shown in Figure 5. Here the different colours in the two graphs again encode the supercliques. The thin black lines between the two graphs show the correspondence matches. Here the results were obtained using edit-distance method described earlier. The graphs are of very different size. The set of roughly parallel lines correspond to the correct correspondences, and the remaining lines are the correspondence errors.

6.2 Real-Word Data

The real-world data used is taken from the CMU model-house sequence. This sequence is made up of a series of images which have been captured from different viewpoints. In order to convert the images into abstract graphs for matching, we extract point features using corner detector by Luo, Cross and Hancock [22]. Our graphs are the Delaunay triangulations of the corner-features. The supercliques obtained by graph partition are shown in different colors in Figure 6.

We have matched the first image to each of the subsequent images in the sequence by using discrete relaxation and edit distance. The results of those two methods are compared with those obtained using the method of Luo, Cross and Hancock [22] in Table 1. This table contains the number of detected corners to be matched, the number of correct correspondence, the number of missed corners and the number of miss-matched corners.

Figure 7 shows us the correct correspondence rate as a function of view difference for the two methods based on the data in Table 1. From the results, it is clear that our new method degrades gradually and out performs the Luo and Hancock's EM method when the difference in viewing angle is large. Figures 8 to 11 show the results of each pair of graph matching. There are clearly significant structural differences in the graphs including rotation, scaling and perspective distortion. But even in the worst case, our method has a correct correspondence rate of 36%.

6.3 Graph clustering

We have collected sequences of views for three toy houses. For each object the image sequences are obtained under slowly varying changes in viewer direction. From each image in each view sequence, we extract corner features. We use the extracted corner points to construct Delaunay graphs. In our experiments we use three different sequences. Each sequence contains images with equally spaced viewing directions. In Figure 12 we show examples of the raw image data and the associated graphs for the three toy houses, which we refer to as CMU/VASC, MOVI and Swiss Chalet.

In Figure 13 the two panels show the distances $d(k_1, k_2) = (B_{k_1} - B_{k_2})^T (B_{k_1} - B_{k_2})$ between the vectors of eigenvalues for the graphs indexed k_1 and k_2. The left panel is for the original graph and the right panel is for the simplified graph. It is clear that the simplification process has preserved much of the structure in the distance plot. For instance, the three sequences are clearly visible as blocks in the panels. Figure 14 shows a scatter plot of the distance between the simplified graphs (y-axis) as a function of the distance between the original graphs. Although there is considerable dispersion, there is an underlying linear trend.

Figures 15 and 16 repeat the distance matrices and the scatter plot using edit distance rather than the L2 norm for the spectral feature vectors. Again, there is a clear block structure. However, the dispersion in the scatter plot is greater. To take this study one step further, in Figures 17 and 18 we show the result of performing MDS on the distances for both the edit distance and the spectral feature vector. Here the images from which the graphs are extracted are shown as thumbnails embedded in the space spanned by the leading eigenvectors of the MDS analysis. In both cases the views of the different houses fall into distinct regions of the plot. Moreover, the hierarchical simplification of the graphs does not destroy the cluster structure.

7 Conclusions

In this paper, we have used the Fiedler vector of the Laplacian matrix to partition the nodes of a graph into structural units for the purposes of matching. This allows us to decompose the problem of matching the graphs into that of matching structural subunits. We investigate the matching of the structural subunits using a edit distance method. The partitioning method is sufficiently stable under structural error that accuracy of match is not sacrificed. Our motivation in undertaking this study is to use the partitions to develop a hierarchical matching method. The aim is to construct a graph that represents

the arrangement of the partitions. By first matching the partition arrangement graphs, we provide constraints on the matching of the individual partitions.

References

[1] B.W. Kernighan and S. Lin. An efficient heuristic procedure for partitioning graphs. *The Bell System Technical Journal*, pages 291–307, 1970.

[2] R.J. Wilson and J. J. Watkins. *Graphs: an introductory approach: a first course in discrete mathematics*. Wiley international edition. New York, etc., Wiley, 1990.

[3] B.T. M. and H. Bunke. A new algorithm for error-tolerant subgraph isomorphism detection. *IEEE PAMI*, 20:493–504, 1998.

[4] F.R.K. Chung. *Spectral Graph Theory*. CBMS series 92. American Mathmatical Society Ed., 1997.

[5] S. Umeyama. An eigendecomposition approach to weighted graph matching problems. *IEEE PAMI*, 10:695–703, 1988.

[6] L. Shapiro and J. Brady. Feature-based correspondence: an eigenvector approach. *Image and Vision Computing*, 10(2):283–288, June 1992.

[7] B. Luo and E. R. Hancock. Structural graph matching using the em algorithm and singular value decomposition. *IEEE PAMI*, 23(10):1120–1136, 2001.

[8] R. C. Wilson and E. R. Hancock. Structural matching by discrete relaxation. *IEEE PAMI*, 19(6):634–648, 1997.

[9] A. Shokoufandeh, S.J. Dickinson, K. Siddiqi, and S.W. Zucker. Indexing using a spectral encoding of topological structure. *IEEE-ICPR*, pages 491–497, 1999.

[10] B. Mohar. Some applications of laplace eigenvalues of graphs. *Graph Symmetry: Algebraic Methods and Applications*, 497 NATO ASI Series C:227–275, 1997.

[11] W.H. Haemers. Interlacing eigenvalues and graphs. *Linear Algebra and its Applications*, pages 593–616, 1995.

[12] J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE PAMI*, 22(8):888–905, 2000.

[13] J. Diaz, J. Petit, and M. Serna. A survey on graph layout problems. Technical report LSI-00-61-R, Universitat Polit'ecnica de Catalunya, 2000.

[14] J. E. Atkins, E. G. Boman, and B. Hendrickson. A spectral algorithm for seriation and the consecutive ones problem. *SIAM Journal on Computing*, 1998.

[15] M. Fiedler. A property of eigenvectors of non-negative symmetric matrices and its application to graph theory. *Czechoslovak Mathematics Journal*, 25:619–633, 1975.

[16] R. Myers, R. C. Wilson, and E. R. Hancock. Bayesian graph edit distance. *IEEE PAMI*, 22(6):628–635, 2000.

[17] V. I. Levenshtein. Binary codes capable of correcting deletions insertions and reversals. *Soviet Physics-Doklandy*, 10(8):707–710, 1966.

[18] R. A. Wagner and M. Fischer. The string-to-string correction problem. *J. ACM*, 21(1):168–173, 1974.

[19] A. Marzal and E. Vidal. Computation of normalized edit distance and applications. *IEEE Trans. Systems, Man, and Cybernetics*, 25:202–206, 1995.

[20] S. Gold and A. Rangarajan. A graduated assignment algorithm for graph matching. *IEEE PAMI*, 18(4):377–388, 1996.

[21] A.M. Finch, R.C. Wilson, and E.R. Hancock. An energy function and continuous edit process for graph matching. *Neural Computation*, 10(7):1873–1894, 1998.

[22] B. Luo, A. D. Cross, and E. R. Hancock. Corner detection via topographic analysis of vector potential. *Proceedings of the 9 th British Machine Vision Conference*, 1998.

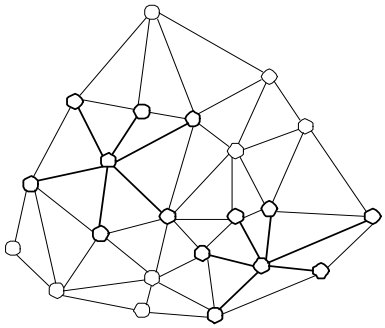

Fig. 1. Neighbourhoods.

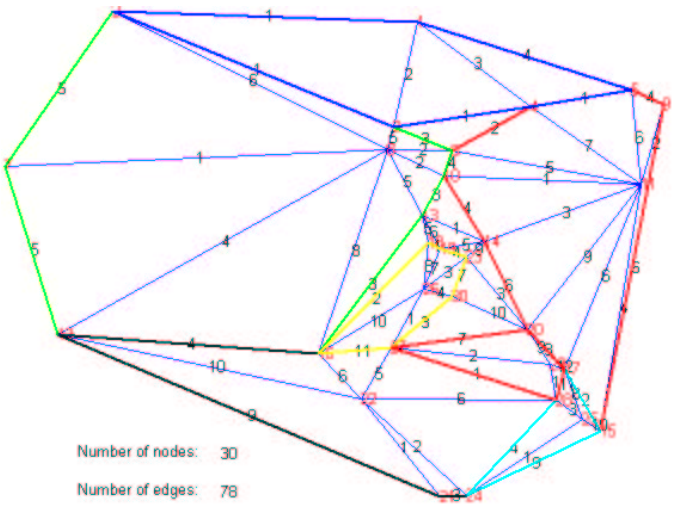

Fig. 2. Graph partition.

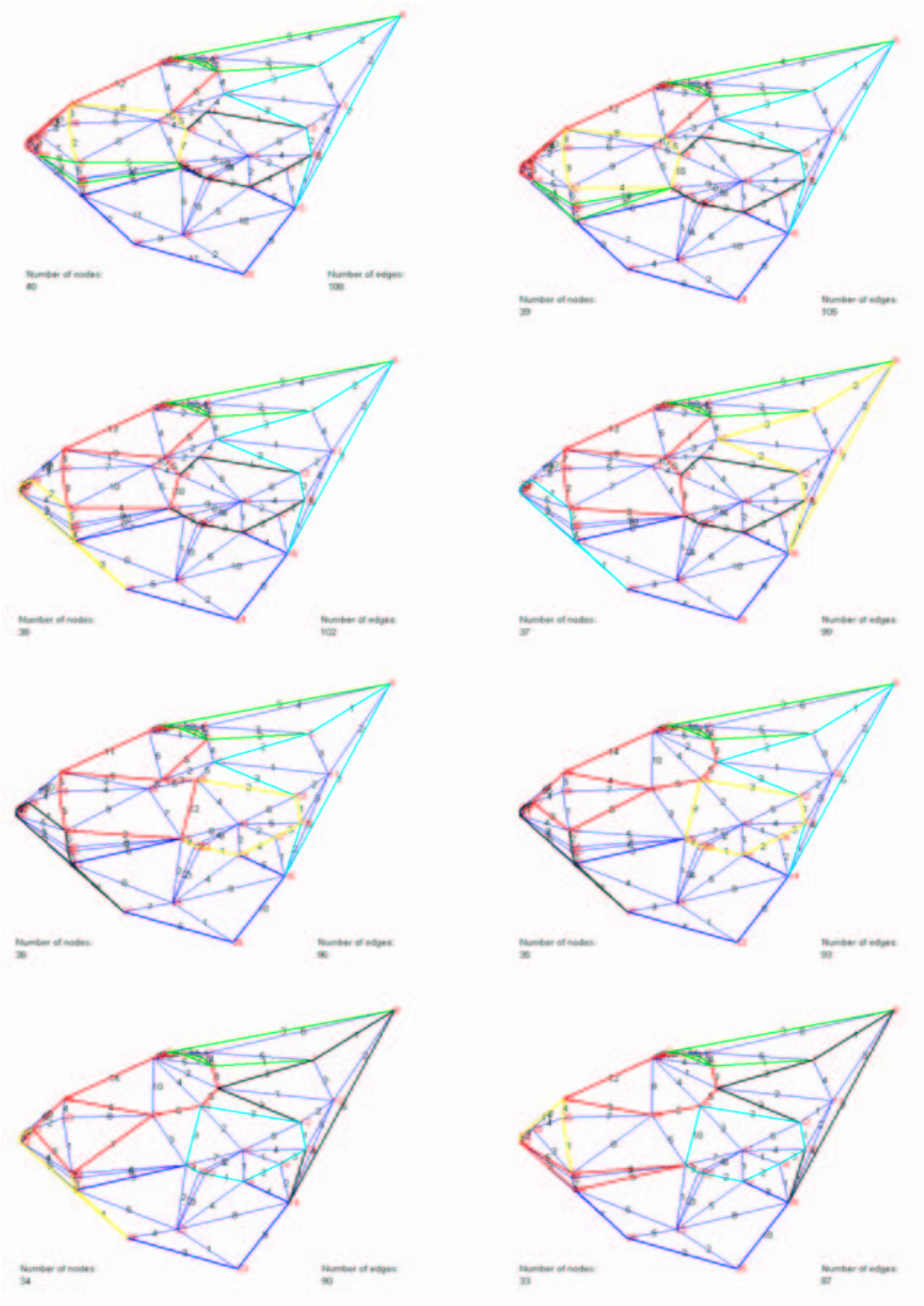

Fig. 3. A sequence of synthetic graphs showing the effect of controlled node deletion on the stability of the supercliques.

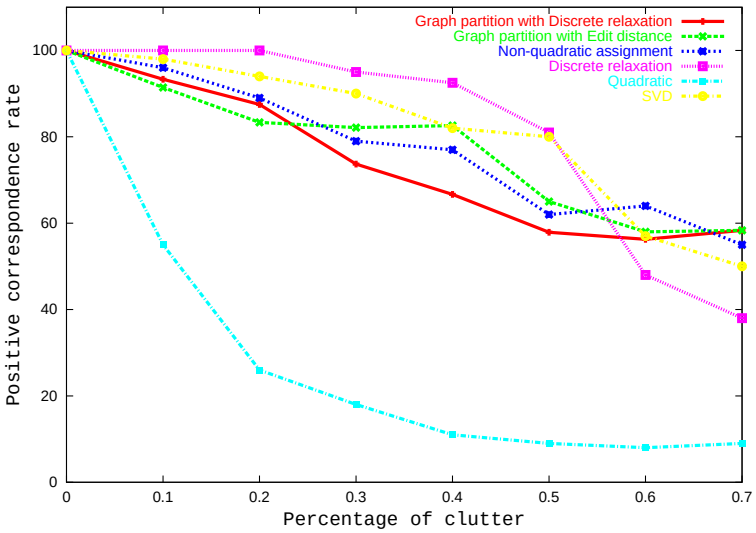

Fig. 4. Sensitivity study for graphs of different size.

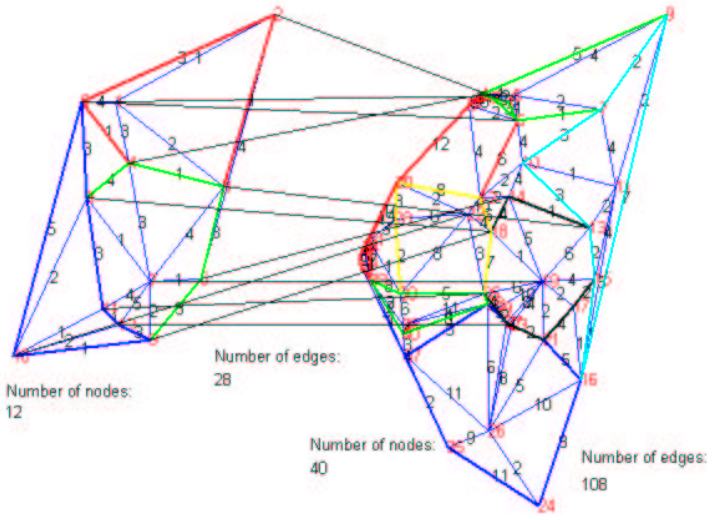

Fig. 5. An example in inexact graph matching.

Fig. 6. Graph partition on Delaunay triangulations.

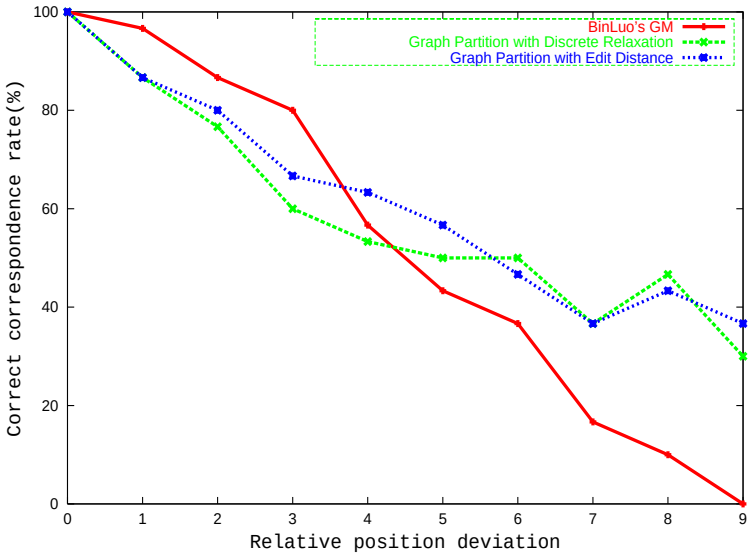

Fig. 7. Comparison of results.

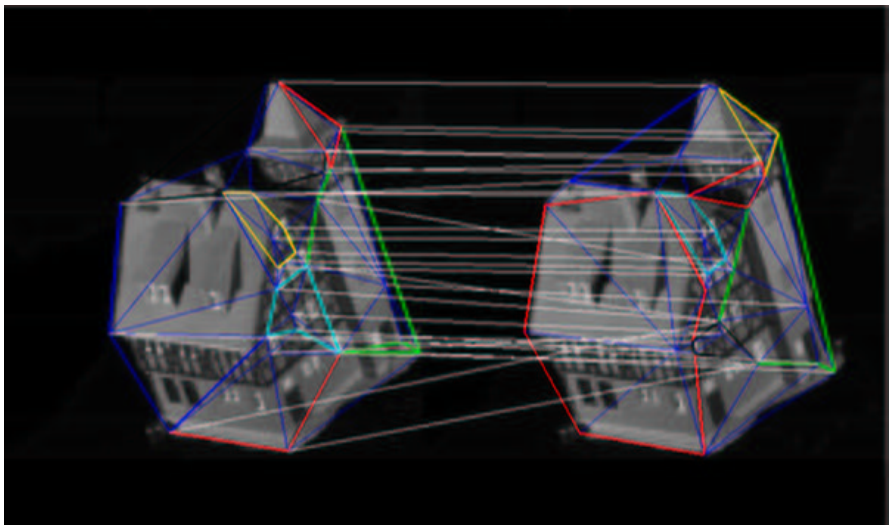

Fig. 8. Correspondences between the first and the third images.

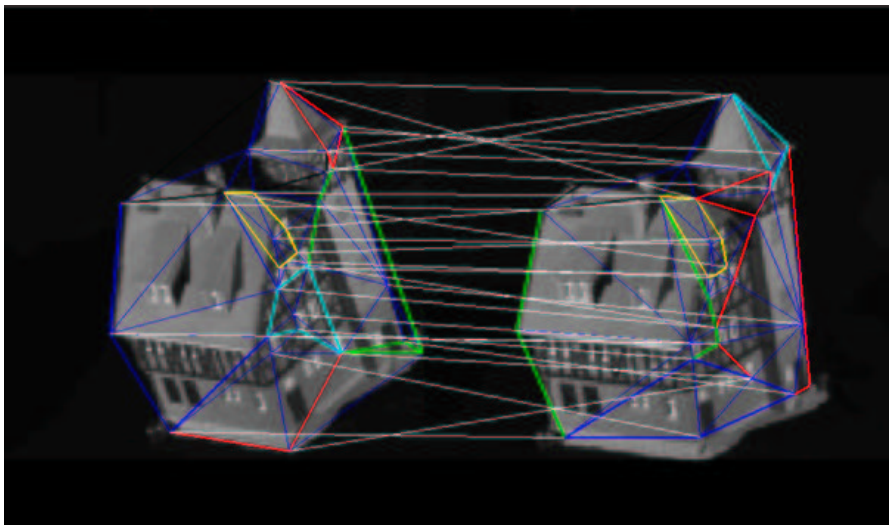

Fig. 9. Correspondences between the first and the fifth images.

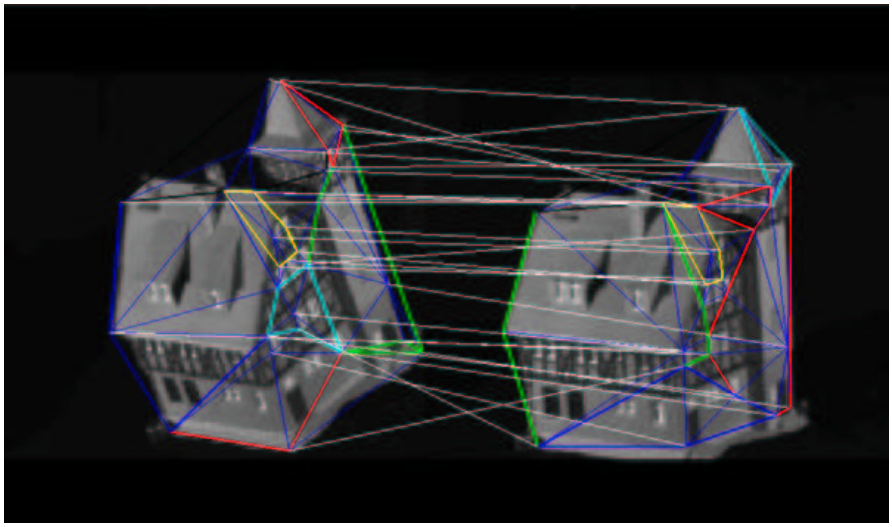

Fig. 10. Correspondences between the first and the seventh images.

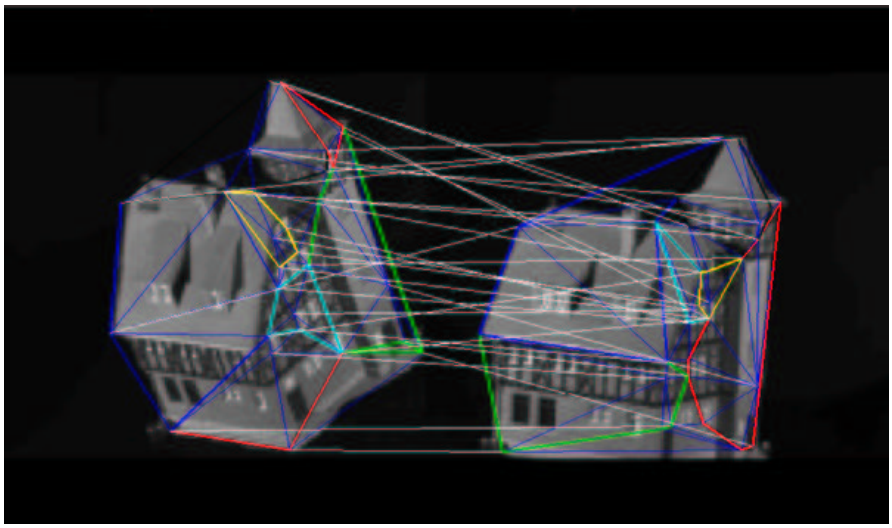

Fig. 11. Correspondences between the first and the tenth images.

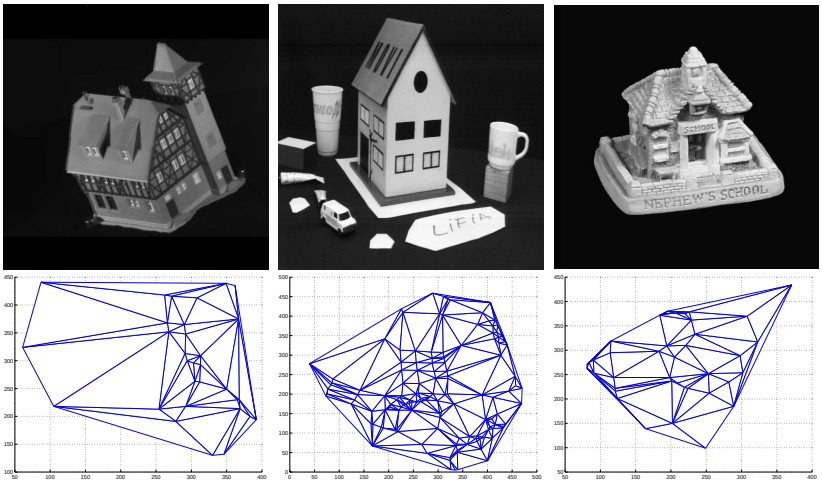

Fig. 12. Example images from the CMU, MOVI and chalet sequences and their corresponding graphs.

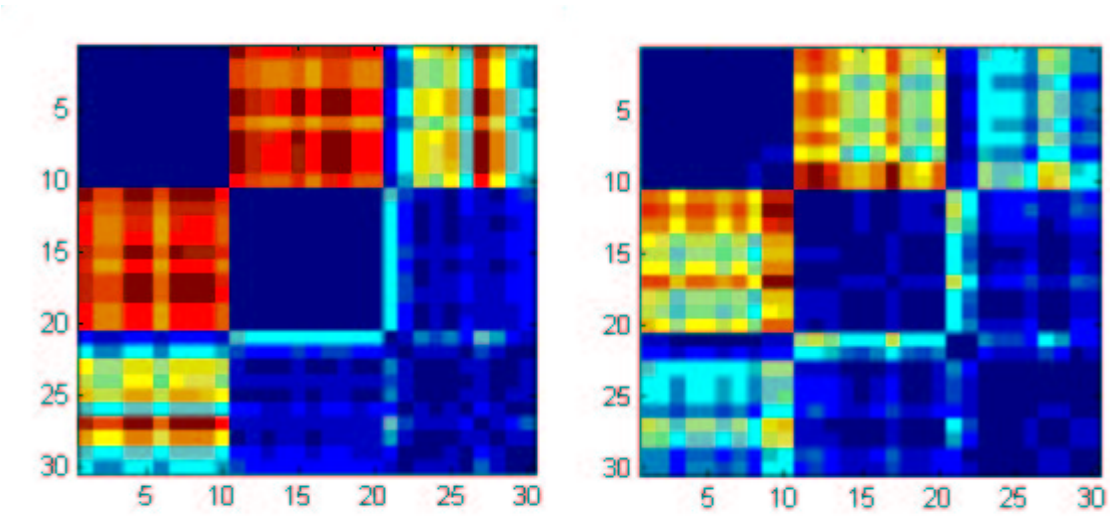

Fig. 13. Pairwise spectral graph distance; (left) original graph, (right) reduced graph.

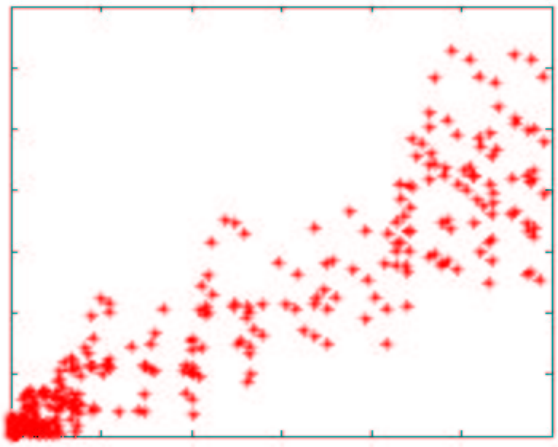

Fig. 14. Scatter plot for the original graph and reduced graph pairwise distance.

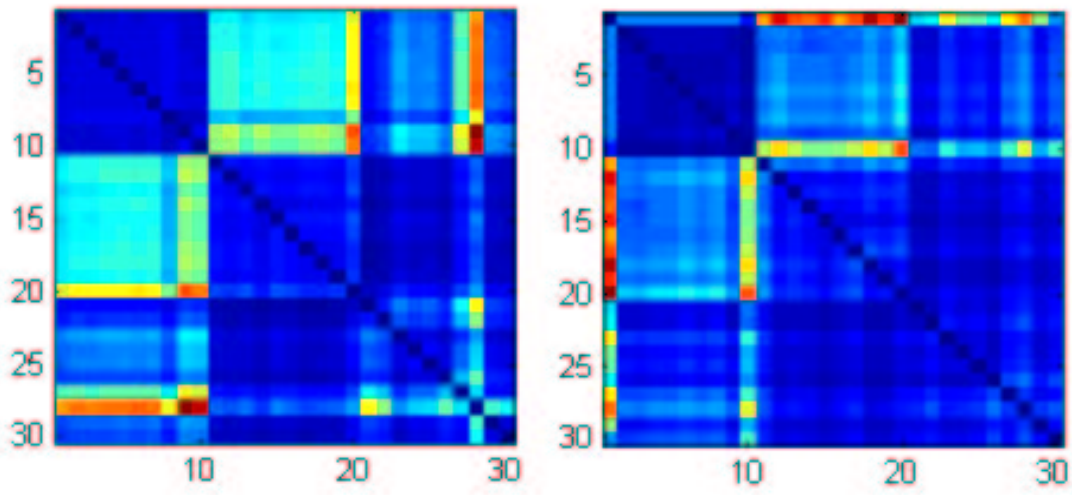

Fig. 15. Graph edit distance; (left) original graph, (right) reduced graph

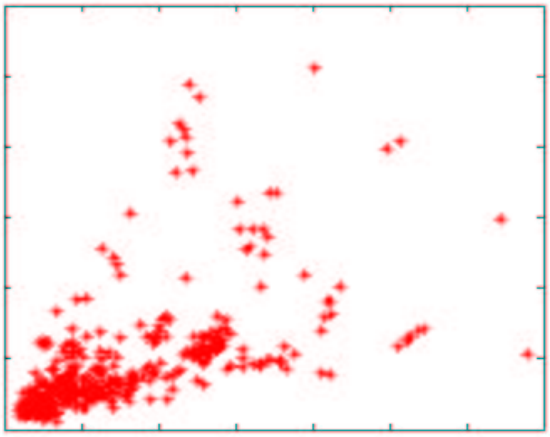

Fig. 16. Scatter plots for the original graph and reduced graph edit distance

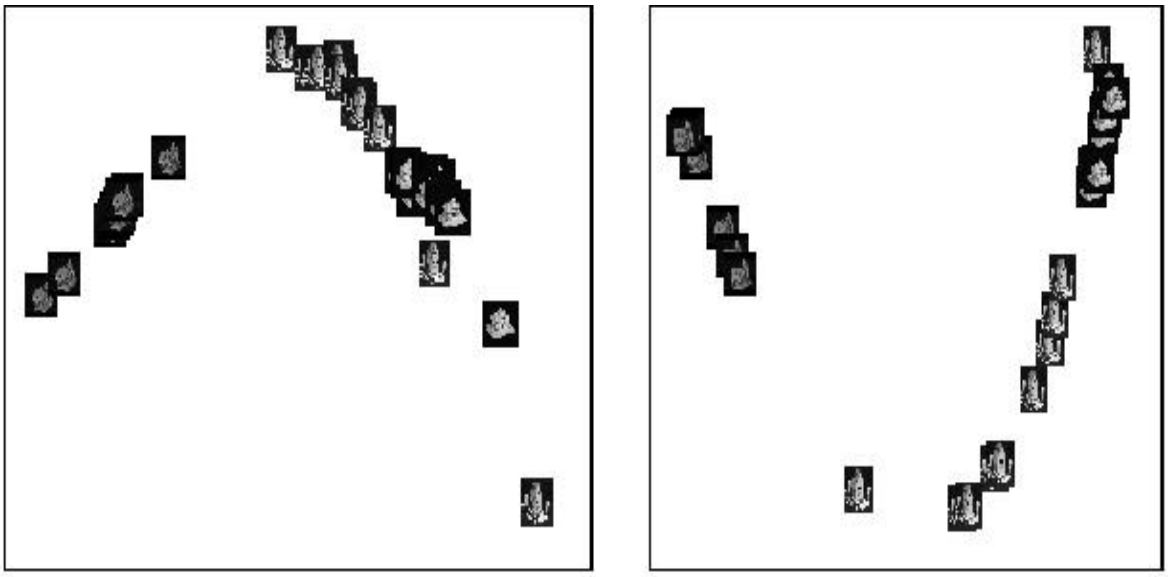

Fig. 17. MDS for the original graph (left) edit distance, (right)spectral feature vector

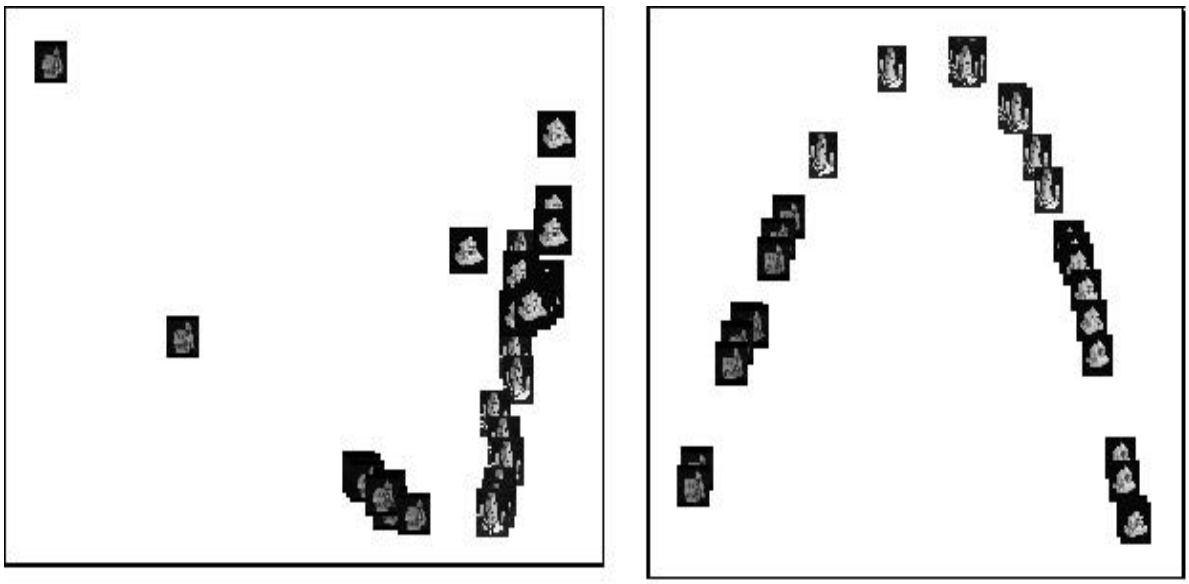

Fig. 18. MDS for the reduced graph (left) edit distance, (right)spectral feature vector

Table 1
Correspondence allocation results for three methods.

Method	House index	0	1	2	3	4	5	6	7	8	9
	Corners	30	32	32	30	30	32	30	30	30	31
EM	Correct	-	29	26	24	17	13	11	5	3	0
	False	-	0	2	3	8	11	12	15	19	24
	Missed	-	1	2	3	5	6	7	10	8	6
Discrete Relaxation	Correct	-	26	23	18	16	15	15	11	14	9
	False	-	4	6	9	12	14	13	17	16	20
	Missed	-	0	1	3	2	1	2	2	0	1
Edit Distance	Correct	-	26	24	20	19	17	14	11	13	11
	False	-	3	5	8	11	12	16	15	17	19
	Missed	-	1	1	2	0	1	0	4	0	0

Biography

HUAIJUN QIU is currently undertaking research towards a D.Phil. degree in computer vision in the Department of Computer Science at the University of York. Prior to this he gained his B.Sc in Computer Science from the East China University of Science and Technology-China(2001). His research interests are in graph matching and clustering; image skeletonization and segmentation. He has published some 10 papers on these topics.

EDWIN HANCOCK studied physics as an undergraduate at the University of Durham and graduated with honours in 1977. He remained at Durham to complete a PhD in the area of high energy physics in 1981. Following this he worked for ten years as a researcher in the fields of high-energy nuclear physics and pattern recognition at the Rutherford-Appleton Laboratory (now the Central Research Laboratory of the Research Councils). During this period he also held adjunct teaching posts at the University of Surrey and the Open University. In 1991 he moved to the University of York as a lecturer in the Department of Computer Science. He was promoted to Senior Lecturer in 1997 and to Reader in 1998. In 1998 he was appointed to a Chair in Computer Vision.

Professor Hancock now leads a group of some 15 faculty, research staff and PhD students working in the areas of computer vision and pattern recognition. His main research interests are in the use of optimisation and probabilistic methods for high and intermediate level vision. He is also interested in the methodology of structural and statistical pattern recognition. He is currently working on graph-matching, shape-from-X, image data-bases and statistical learning theory. His work has found applications in areas such as radar terrain analysis, seismic section analysis, remote sensing and medical imaging. Professor Hancock has published some 90 journal papers and 350 refereed conference publications. He was awarded the Pattern Recognition Society medal in 1991 and an outstanding paper award in 1997 by the journal Pattern Recognition. In 1998 he became a fellow of the International Association for Pattern Recognition.

Professor Hancock has been a member of the Editorial Boards of the journals IEEE Transactions on Pattern Analysis and Machine Intelligence, and, Pattern Recognition. He has also been a guest editor for special editions of the journals Image and Vision Computing and Pattern Recognition. He has been on the programme committees for numerous national and international meetings. In 1997 with Marcello Pelillo, he established a new series of international meetings on energy minimisation methods in computer vision and pattern recognition.