This is a repository copy of *Pattern vectors from algebraic graph theory*.

Article:

Pattern Vectors from Algebraic Graph Theory

Richard C. Wilson, *Member*, *IEEE Computer Society*, Edwin R. Hancock, and Bin Luo

Abstract—Graph structures have proven computationally cumbersome for pattern analysis. The reason for this is that, before graphs can be converted to pattern vectors, correspondences must be established between the nodes of structures which are potentially of different size. To overcome this problem, in this paper, we turn to the spectral decomposition of the Laplacian matrix. We show how the elements of the spectral matrix for the Laplacian can be used to construct symmetric polynomials that are permutation invariants. The coefficients of these polynomials can be used as graph features which can be encoded in a vectorial manner. We extend this representation to graphs in which there are unary attributes on the nodes and binary attributes on the edges by using the spectral decomposition of a Hermitian property matrix that can be viewed as a complex analogue of the Laplacian. To embed the graphs in a pattern space, we explore whether the vectors of invariants can be embedded in a low-dimensional space using a number of alternative strategies, including principal components analysis (PCA), multidimensional scaling (MDS), and locality preserving projection (LPP). Experimentally, we demonstrate that the embeddings result in well-defined graph clusters. Our experiments with the spectral representation involve both synthetic and real-world data. The experiments with synthetic data demonstrate that the distances between spectral feature vectors can be used to discriminate between graphs on the basis of their structure. The real-world experiments show that the method can be used to locate clusters of graphs.

Index Terms—Graph matching, graph features, spectral methods.

✦

1 INTRODUCTION

THE analysis of relational patterns or graphs has proven to be considerably more elusive than the analysis of vectorial patterns. Relational patterns arise naturally in the representation of data in which there is a structural arrangement and are encountered in computer vision, genomics, and network analysis. One of the challenges that arise in these domains is that of knowledge discovery from large graph data sets. The tools that are required in this endeavor are robust algorithms that can be used to organize, query, and navigate large sets of graphs. In particular, the graphs need to be embedded in a pattern space so that similar structures are close together and dissimilar ones are far apart. Moreover, if the graphs can be embedded on a manifold in a pattern space, then the modes of shape variation can be explored by traversing the manifold in a systematic way. The process of constructing low-dimensional spaces or manifolds is a routine procedure with pattern-vectors. A variety of well-established techniques, such as principal components analysis [12], multidimensional scaling [7], and independent components analysis, together with more recently developed ones, such as locally linear embedding [28], isomap [36], and the Laplacian eigenmap [2], exist for solving the problem. In addition to providing generic data analysis tools, these methods have been extensively exploited in computer vision to construct shape—spaces for 2D or 3D objects and, in particular, faces represented using coordinate and intensity data. Collectively, these methods are sometimes referred to as manifold learning theory. However, there are few analogous methods which can be used to construct low-dimensional pattern spaces or manifolds for sets of graphs.

There are two reasons why pattern-vectors are more easily manipulated than graphs. First, there is no natural ordering for the nodes in a graph, unlike the components of a vector. Hence, correspondences to a reference structure must be established as a prerequisite. The second problem is that the variation in the graphs of a particular class may manifest itself as subtle changes in structure, which may involve different numbers of nodes or different edge structure. Even if the nodes or the edges of a graph could be encoded in a vectorial manner, then the vectors would be of variable length. One way of circumventing these problems is to develop graph-clustering methods in which an explicit class archetype and correspondences with it are maintained [25], [31]. Graph clustering is an important process since it can be used to structure large databases of graphs [32] in a manner that renders retrieval efficient. One of the problems that hinders graph clustering is the need to define a class archetype that can capture both the salient structure of the class and the modes of variation contained within it. For instance, Munger et al. [22] have recently taken some important steps in this direction by developing a genetic algorithm for searching for median graphs. Variations in class structure can be captured and learned provided a suitable probability distribution can be defined over the nodes and edges of the archetype. For instance, the random graphs of Wong et al. [40] capture this distribution using a discretely defined probability distribution and Bagdanov and Worring [1] have overcome some of the computational difficulties associated with this method by using continuous Gaussian distributions. There is a considerable body of related literature in the graphical models community concerned with learning the structure of Bayesian networks from data [11]. Moreover, in recent work, we have shown that, if reliable correspondence information is to hand, then the edge structure of the graph can be encoded as a

- *R.C. Wilson and E.R. Hancock are with the Department of Computer Science, University of York, Heslington, York Y01 5DD, UK.*
  E-mail: {wilson, erh}@cs.york.ac.uk.
- *B. Luo is with the Key Lab of Intelligent Computing and Signal Processing of the Ministry of Education, Anhui University, Hefei 230039, P.R. China.*
  E-mail: luobin@ahu.edu.cn.

long-vector and embedded in a low-dimensional space using principal components analysis [20]. An alternative to using a probability distribution over a class archetype is to use pairwise clustering methods. Here, the starting point is a matrix of pairwise affinities between graphs. There are a variety of algorithms available for performing pairwise clustering, but one of the most popular approaches is to use spectral methods, which use the eigenvectors of an affinity matrix to define clusters [33], [13]. There are a number of examples of applying pairwise clustering methods to graph edit distances and similarity measures [19], [23].

However, one of the criticisms that can be aimed at these methods for learning the distribution of graphs is that they are, in a sense, brute force because of their need for correspondences either to establish an archetype or to compute graph similarity. For noisy graphs (those which are subject to structural differences), this problem is thought to be NP-hard. Although relatively robust approximate methods exist for computing correspondence [5], these can prove time consuming. In this paper, we are interested in the problem of constructing pattern vectors for graphs. These vectors represent the graph without the need to compute correspondences. These vectors are necessarily large, since they must represent the entire set of graphs of a particular size. For this reason, we are also interested in pattern spaces for families of graphs. In particular, we are interested in the problem of whether it is possible to construct a low-dimensional pattern space from our pattern vectors.

The adopted approach is based on spectral graph theory [6], [21], [3]. Although existing graph-spectral methods have proven effective for graph-matching and indexing [38], they have not made full use of the available spectral representation and are restricted to the use of either the spectrum of eigenvalues or a single eigenvector.

1.1 Related Literature

Spectral graph theory is concerned with understanding how the structural properties of graphs can be characterized using the eigenvectors of the adjacency matrix or the closely related Laplacian matrix (the degree matrix minus the adjacency matrix). There are good introductory texts on the subject by Biggs [3] and Cvetkovic et al. [8]. Comprehensive reviews of recent progress in the field can be found in the research monograph of Chung [6] and the survey papers of Lovasc [17] and Mohar [21]. The subject has acquired considerable topicality in the computer science literature since spectral methods can be used to develop efficient methods for path planning and data clustering. In fact, the Googlebot search engine uses ideas derived from spectral graph theory. Techniques from spectral graph theory have been applied in a number of areas in computer vision, too, and have led to the development of algorithms for grouping and segmentation [33], correspondence matching [38], [18], and shape indexing [34]. Among the first to apply spectral ideas to the grouping problem were Scott and Longuet-Higgins [30], who showed how to extract groups of points by relocating the eigenvectors of a point-proximity matrix. However, probably the best known work in this area is that of Shi and Malik [33], who showed how the Fiedler (second smallest) eigenvector of the Laplacian matrix could be used to locate groupings that optimize a normalized cut measure. In related work, both Perona and Freeman [24], and Sarkar and Boyer [29] have shown how the thresholded leading eigenvector of the weighted adjacency matrix can be used for grouping points

and line-segments. Robles-Kelly and Hancock [26] have developed a statistical variant of the method which avoids thresholding the leading eigenvector and instead employs the EM algorithm to iteratively locate clusters of objects.

The use of graph-spectral methods for correspondence matching has proven to be an altogether more elusive task. The idea underpinning spectral methods for correspondence analysis is to locate matches between nodes by comparing the eigenvectors of the adjacency matrix. However, although this method works well for graphs with the same number of nodes and small differences in edge structure, it does not work well when graphs of different size are being matched. The reason for this is that the eigenvectors of the adjacency matrix are unstable under changes in the size of the adjacency matrix. Turning to the literature, the algorithm of Umeyama [38] illustrates this point very clearly. The method commences by performing singular value decomposition on the adjacency matrices of the graphs under study. The permutation matrix which brings nodes of the graphs into correspondence is found by taking the sum of the outer products of the sets of corresponding singular vectors. The method cannot be operated when the adjacency matrices are of different size, i.e., the singular vectors are of different length. To overcome this problem, Luo and Hancock [18], have drawn on the apparatus of the EM algorithm and treat the correspondences as missing or hidden data. By introducing a correspondence probability matrix, they overcome problems associated with the different sizes of the adjacency matrices. An alternative solution to the problem of size difference is adopted by Kosinov and Caelli [16] who project the graph onto the eigenspace spanned by its leading eigenvectors. Provided that the eigenvectors are normalized, then the relative angles of the nodes are robust to size difference under the projection. In related work, Kesselman et al. have used a minimum distortion procedure to embed graphs in a low-dimensional space where correspondences can be located between groups of nodes [14]. Recent work by Robles-Kelly and Hancock [27] has shown how an eigenvector method can be used to sort the nodes of a graph into string order and how string matching methods can be used to overcome the size difference problem. Finally, Shokoufandeh et al. [34] have shown how to use the interleaving property of the eigenvalues to index shock trees.

1.2 Contribution

As noted earlier, one of the problems which hinders the pattern analysis of graphs is that they are neither vectorial in nature nor easily transformed into vectors. Although the spectral methods described above provide a means by which the structure of graphs can be characterized and the required correspondences can be located, they make only limited use of the available spectral information. Moreover, although graph-matching may provide a fine measure of distance between structures and this, in turn, may be used to cluster similar graphs, it does not result in an ordering of the graphs that has metrical significance under structural variations due to graded structural changes. Hence, we aim to use spectral methods to vectorize graphs and, hence, embed them in low-dimensional pattern spaces using manifold learning methods. We describe a method for constructing spectral features which are permutation invariants and which make use of the full spectral matrix. To construct these invariants, we use symmetric polynomials. The arguments of the polynomials are the elements of the spectral matrix. We use the values of the symmetric polynomials to construct graph pattern-vectors.

Size differences between graphs can be accommodated by padding the spectral matrix with trailing vectors of zeros.

The representation can be extended to attributed graphs using graph property matrices with complex entries instead of the real valued Laplacian. We show how weighted and attributed edges can be encoded by complex numbers with the weight as the magnitude or modulus and the attribute as the phase. If the property matrices are required to be Hermitian, then the eigenvalues are real and the eigenvectors are complex. The real and imaginary components of the eigenvectors can again be used to compute symmetric polynomials.

We explore how the spectral feature vectors may be used to construct pattern spaces for sets of graphs [14]. We investigate a number of different approaches. The first of these is to apply principal components analysis to the covariance matrix for the vectors. This locates a variance preserving embedding for the pattern vectors. The second approach is multidimensional scaling and this preserves the distance between vectors. The third approach is locally linear projection which aims to preserve both the variance of the data and the pattern of distances [10]. We demonstrate that this latter method gives the best graph-clusters.

The outline of this paper is as follows: Section 2 details our construction of the spectral matrix. In Section 3, we show how symmetric polynomials can be used to construct permutation invariants from the elements of the spectral matrix. Section 4 provides details of how the symmetric polynomials can be used to construct useful pattern vectors and how problems of variable node-set size can be accommodated. To accommodate attributes, in Section 5, we show how to extend the representation using complex numbers to construct a Hermitian property matrix. Section 6 briefly reviews how PCA, MDS, and locality preserving projection can be performed on the resulting pattern vectors. In Section 7, we present results on both synthetic and real-world data. Finally, Section 8 offers some conclusions and suggests directions for further investigation.

2   SPECTRAL GRAPH REPRESENTATION

Consider the undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{W})$ with node-set $\mathcal{V} = \{\nu_1, \nu_2, \ldots, \nu_n\}$, edge-set $\mathcal{E} = \{e_1, e_2, \ldots, e_m\} \subset \mathcal{V} \times \mathcal{V}$, and weight function $\mathcal{W} : \mathcal{E} \rightarrow (0, 1]$. The adjacency matrix $\mathbf{A}$ for the graph $\mathcal{G}$ is the $n \times n$ matrix with elements

$$A_{ab} = \begin{cases} 1 & \text{if } (\nu_a, \nu_b) \in \mathcal{E} \\ 0 & \text{otherwise.} \end{cases}$$

In other words, the matrix represents the edge structure of the graph. Clearly since the graph is undirected, the matrix $\mathbf{A}$ is symmetric. If the graph edges are weighted, the adjacency matrix is defined to be

$$A_{ab} = \begin{cases} \mathcal{W}(\nu_a, \nu_b) & \text{if } (\nu_a, \nu_b) \in \mathcal{E} \\ 0 & \text{otherwise.} \end{cases}$$

The Laplacian of the graph is given by $\mathbf{L} = \mathbf{D} - \mathbf{A}$, where $\mathbf{D}$ is the diagonal node degree matrix whose elements $D_{aa} = \sum_{b=1}^{n} A_{ab}$ are the number of edges which exit the individual nodes. The Laplacian is more suitable for spectral analysis than the adjacency matrix since it is positive semidefinite.

In general, the task of comparing two such graphs, $\mathcal{G}_1$ and $\mathcal{G}_2$, involves finding a correspondence mapping between the nodes of the two graphs, $f : \mathcal{V}_1 \cup \phi \leftrightarrow \mathcal{V}_2 \cup \phi$.

The recovery of the correspondence map f can be posed as that of minimizing an error criterion. The minimum value of the criterion can be taken as a measure of the similarity of the two graphs. The additional node "ϕ" represents a null match or dummy node. Extraneous or additional nodes are matched to the dummy node. A number of search and optimization methods have been developed to solve this problem [5], [9]. We may also consider the correspondence mapping problem as one of finding the permutation of nodes in the second graph which places them in the same order as that of the first graph. This permutation can be used to map the Laplacian matrix of the second graph onto that of the first. If the graphs are *isomorphic*, then this permutation matrix satisfies the condition $\mathbf{L}_1 = \mathbf{P}\mathbf{L}_2\mathbf{P}^T$. When the graphs are not isomorphic, then this condition no longer holds. However, the Frobenius distance $\|\mathbf{L}_1 - \mathbf{P}\mathbf{L}_2\mathbf{P}^T\|$ between the matrices can be used to gauge the degree of similarity between the two graphs. Spectral techniques have been used to solve this problem. For instance, working with adjacency matrices, Umeyama [38] seeks the permutation matrix $\mathbf{P}_U$ that minimizes the Frobenius norm $J(\mathbf{P}_U) = \|\mathbf{P}_U\mathbf{A}_1\mathbf{P}_U^T - \mathbf{A}_2\|$. The method performs the singular value decompositions $\mathbf{A}_1 = \mathbf{U}_1\mathbf{\Delta}_1\mathbf{U}_1^T$ and $\mathbf{A}_2 = \mathbf{U}_2\mathbf{\Delta}_2\mathbf{U}_2^T$, where the $\mathbf{U}$s are orthogonal matrices and the $\mathbf{\Delta}$s are diagonal matrices. Once these factorizations have been performed, the required permutation matrix is approximated by $\mathbf{U}_2\mathbf{U}_1^T$.

In some applications, especially structural chemistry, eigenvalues have also been used to compare the structural similarity of different graphs. However, although the eigenvalue spectrum is a permutation invariant, it represents only a fraction of the information residing in the eigensystem of the adjacency matrix.

Since the matrix $\mathbf{L}$ is positive semidefinite, it has eigenvalues which are all either positive or zero. The spectral matrix is found by performing the eigenvector expansion for the Laplacian matrix $\mathbf{L}$, i.e.,

$$\mathbf{L} = \sum_{i=1}^{n} \lambda_i \mathbf{e}_i \mathbf{e}_i^T,$$

where λ_i and $\mathbf{e}_i$ are the n eigenvectors and eigenvalues of the symmetric matrix $\mathbf{L}$. The spectral matrix then has the scaled eigenvectors as columns and is given by

$$\mathbf{\Phi} = \left(\sqrt{\lambda_1}\mathbf{e}_1, \sqrt{\lambda_2}\mathbf{e}_2, \ldots \sqrt{\lambda_n}\mathbf{e}_n \right). \tag{1}$$

The matrix $\mathbf{\Phi}$ is a complete representation of the graph in the sense that we can use it to reconstruct the original Laplacian matrix using the relation $\mathbf{L} = \mathbf{\Phi}\mathbf{\Phi}^T$.

The matrix $\mathbf{\Phi}$ is a unique representation of $\mathbf{L}$ iff all n eigenvalues are distinct or zero. This follows directly from the fact that there are n distinct eigenvectors when the eigenvalues are also all distinct. When an eigenvalue is repeated, then there exists a subspace spanned by the eigenvectors of the degenerate eigenvalues in which all vectors are also eigenvectors of $\mathbf{L}$. In this situation, if the repeated eigenvalue is nonzero, there is continuum of spectral matrices representing the same graph. However, this is rare for moderately large graphs. Those graphs for which the eigenvalues are distinct or zero are referred to as simple.

3 NODE PERMUTATIONS AND INVARIANTS

The topology of a graph is invariant under permutations of the node labels. However, the adjacency matrix and, hence, the Laplacian matrix, is modified by the node order since the rows and columns are indexed by the node order. If we relabel the nodes, the Laplacian matrix undergoes a permutation of both rows and columns. Let the matrix $\mathbf{P}$ be the permutation matrix representing the change in node order. The permuted matrix is $\mathbf{L}' = \mathbf{PLP}^T$. There is a family of Laplacian matrices which are can be transformed into one another using a permutation matrix. The spectral matrix is also modified by permutations, but the permutation only reorders the rows of the matrix $\mathbf{\Phi}$ To show this, let $\mathbf{L}$ be the Laplacian matrix of a graph $\mathcal{G}$ and let $\mathbf{L}' = \mathbf{PLP}^T$ be the Laplacian matrix obtained by relabeling the nodes using the permutation $\mathbf{P}$. Further, let $\mathbf{e}$ be a normalized eigenvector of $\mathbf{L}$ with associated eigenvalue λ and let $\mathbf{e}' = \mathbf{Pe}$. With these ingredients, we have that

$$\mathbf{L}'\mathbf{e}' = \mathbf{PLP}^T\mathbf{Pe} = \mathbf{PLe} = \lambda\mathbf{e}'.$$

Hence, $\mathbf{e}'$ is an eigenvector of $\mathbf{L}'$ with associated eigenvalue λ. As a result, we can write the spectral matrix $\mathbf{\Phi}'$ of the permuted Laplacian matrix $\mathbf{L}' = \mathbf{\Phi}'\mathbf{\Phi}'^T$ as $\mathbf{\Phi}' = \mathbf{P\Phi}$. Direct comparison of the spectral matrices for different graphs is, hence, not possible because of the unknown permutation.

The eigenvalues of the adjacency matrix have been used as a compact spectral representation for comparing graphs because they are not changed by the application of a permutation matrix. The eigenvalues can be recovered from the spectral matrix using the identity

$$\lambda_j = \sum_{i=1}^{n} \Phi_{ij}^2.$$

The expression $\sum_{i=1}^{n} \Phi_{ij}^2$ is, in fact, a *symmetric polynomial* in the components of eigenvector $\mathbf{e}_i$. A symmetric polynomial is invariant under permutation of the variable indices. In this case, the polynomial is invariant under permutation of the row index i.

In fact, the eigenvalue is just one example of a family of symmetric polynomials which can be defined on the components of the spectral matrix. However, there is a special set of these polynomials, referred to as the *elementary symmetric polynomials* ($\mathcal{S}$) that form a basis set for symmetric polynomials. In other words, any symmetric polynomial can itself be expressed as a polynomial function of the elementary symmetric polynomials belonging to the set $\mathcal{S}$.

We therefore turn our attention to the set of elementary symmetric polynomials. For a set of variables $\{v_1, v_2 \ldots v_n\}$, they can be defined as

$$
\begin{aligned}
S_1(v_1, \ldots v_n) &= \sum_{i=1}^{n} v_i, \\
S_2(v_1, \ldots v_n) &= \sum_{i=1}^{n}\sum_{j=i+1}^{n} v_i v_j, \\
&\vdots \\
S_r(v_1, \ldots v_n) &= \sum_{i_1 < i_2 < \ldots < i_r} v_{i_1} v_{i_2} \ldots v_{i_r}, \\
&\vdots \\
S_n(v_1, \ldots v_n) &= \prod_{i=1}^{n} v_i.
\end{aligned}
$$

The power symmetric polynomial functions,

$$
\begin{aligned}
P_1(v_1, \ldots v_n) &= \sum_{i=1}^{n} v_i, \\
P_2(v_1, \ldots v_n) &= \sum_{i=1}^{n} v_i^2, \\
&\vdots \\
P_r(v_1, \ldots v_n) &= \sum_{i=1}^{n} v_i^r, \\
&\vdots \\
P_n(v_1, \ldots v_n) &= \sum_{i=1}^{n} v_i^n,
\end{aligned}
$$

also form a basis set over the set of symmetric polynomials. As a consequence, any function which is invariant to permutation of the variable indices and that can be expanded as a Taylor series can be expressed in terms of one of these sets of polynomials. The two sets of polynomials are related to one another by the Newton-Girard formula:

$$S_r = \frac{(-1)^{r+1}}{r} \sum_{k=1}^{r} (-1)^{k+r} P_k S_{r-k}, \qquad (2)$$

where we have used the shorthand S_r for $S_r(v_1, \ldots\ldots, v_n)$ and P_r for $P_r(v_1, \ldots, v_n)$. As a consequence, the elementary symmetric polynomials can be efficiently computed using the power symmetric polynomials.

In this paper, we intend to use the polynomials to construct invariants from the elements of the spectral matrix. The polynomials can provide spectral "features" which are invariant under node permutations of the nodes in a graph and utilize the full spectral matrix. These features are constructed as follows: Each column of the spectral matrix $\mathbf{\Phi}$ forms the input to the set of spectral polynomials. For example, the column $\{\Phi_{1,i}, \Phi_{2,i}, \ldots, \Phi_{n,i}\}$ produces the polynomials $S_1(\Phi_{1,i}, \ldots, \Phi_{n,i})$, $S_2(\Phi_{1,i}, \ldots, \Phi_{n,i}), \ldots, S_n(\Phi_{1,i}, \ldots, \Phi_{n,i})$. The values of each of these polynomials is invariant to the node order of the Laplacian. We can construct a set of n^2 spectral features using the n columns of the spectral matrix in combination with the n symmetric polynomials.

Each set of n features for each spectral mode contains all the information about that mode up to a permutation of components. This means that it is possible to reconstruct the original components of the mode given the values of the features only. This is achieved using the relationship between the roots of a polynomial in x and the elementary symmetric polynomials. The polynomial

$$\prod_i (x - v_i) = 0 \qquad (3)$$

has roots $v_1, v_2, \ldots, v_n$. Multiplying out (3) gives

$$x^n - S_1 x^{n-1} + S_2 x^{n-2} + \ldots + (-1)^n S_n = 0, \qquad (4)$$

where we have again used the shorthand S_r for $S_r(v_1, \ldots\ldots, v_n)$. By substituting the feature values into (4) and finding the roots, we can recover the values of the original components. The root order is undetermined, so, as expected, the values are recovered up to a permutation.

4 PATTERN VECTORS FROM SYMMETRIC POLYNOMIALS

In this section, we detail how the symmetric polynomials can be used to construct graph feature vectors which can be used for the purposes of pattern analysis. There are two practical issues that need to be addressed. The first of these is how to deal with graphs which have node-sets of different size. The second is how to transform the vectors into an informative representation.

4.1 Graphs of Different Sizes

In order to accommodate graphs with different numbers of nodes, we need to be able to compare spectral representations of different sizes. This is achieved by expanding the representation. Consider two graphs with m and n nodes, respectively, where $m < n$. If we add $n - m$ nodes with no connections to the smaller graph, we obtain two graphs of the same size. The edit cost in terms of edge insertions and deletions between these two graphs is identical to the original pair. The effect on the spectral representation is merely to add trailing zeros to each eigenvector and also additional zero eigenmodes. As a consequence, the first m elementary symmetric polynomials are unchanged and the subsequent $n - m$ polynomials are zero. The new representation in the symmetric polynomials S can therefore be easily calculated. The corresponding power symmetric polynomials can be calculated using the Newton-Girard formula. This new representation can be directly compared with that for the larger graph.

4.2 Feature Distributions

While the elementary symmetric polynomials provide spectral features which are invariant to permutations, they are not suitable as a representation for gauging the difference between graphs. The distribution of S_r for large r is highly non-Gaussian with a dominant tail because of the product terms appearing in the higher order polynomials. In order to make the distribution more tractable, it is convenient to take logarithms. If we wish to take logarithms, the condition $S_r > 0 \; \forall r$ must hold. We therefore perform a component transform and construct the following matrix from the symmetric polynomials

$$F_{ij} = \mathrm{signum}\big(S_j(\Phi_{1,i}, \Phi_{2,i}, \ldots, \Phi_{n,i})\big) \\ \ln\big(1 + |S_j(\Phi_{1,i}, \Phi_{2,i}, \ldots, \Phi_{n,i})|\big), \quad (5)$$

where $1 \leq i \leq n, 1 \leq j \leq n$. The rows of this matrix are stacked to form a long-vector

$$\mathbf{B} = (F_{1,1}, \ldots, F_{1,n}, F_{2,1}, \ldots, F_{2,n}, \ldots, F_{n,1}, \ldots, F_{n,n})^T. \quad (6)$$

4.3 Complexity

The computational complexity of generating pattern vectors is governed by the eigendecomposition of the Laplacian matrix. This requires $O(n^3)$ operations, where n is the number of vertices in the graph. Computation of the symmetric polynomials and subsequent feature representations only requires $O(n^2)$ operations. These operations are effectively preprocessing of the graph structures. Matching of two graphs simply involves comparing two pattern vectors of length n^2 and, so, takes $O(n^2)$ operations. Standard inexact matching techniques are generally iterative and, for comparison, both the graduated assignment algorithm of

Gold and Rangarajan [9] and probabilistic relaxation [5] require $O(n^4)$ operations per iteration.

5 UNARY AND BINARY ATTRIBUTES

Attributed graphs are an important class of representations and need to be accommodated in graph matching. Van Wyk and van Wyk [39] have suggested the use of an augmented matrix to capture additional measurement information. While it is possible to make use of a such an approach in conjunction with symmetric polynomials, another approach from the spectral theory of Hermitian matrices suggests itself.

A Hermitian matrix $\mathbf{H}$ is a square matrix with complex elements that remains unchanged under the joint operation of transposition and complex conjugation of the elements (denoted by the dagger operator †), i.e., $\mathbf{H}^\dagger = \mathbf{H}$. Hermitian matrices can be viewed as the complex number counterpart of the symmetric matrix for real numbers. Each off-diagonal element is a complex number which has two components and can therefore represent a 2-component measurement vector on a graph edge. The on-diagonal elements are, necessarily, real quantities, so the node measurements are restricted to be scalar quantities.

There are some constraints on how the measurements may be represented in order to produce a positive semidefinite Hermitian matrix. Let $\{x_1, x_2, \ldots, x_n\}$ be a set of measurements for the node-set $\mathcal{V}$. Further, let $\{y_{1,2}, y_{1,3}, \ldots, y_{n,n}\}$ be the set of measurements associated with the edges of the graph in addition to the graph weights. Each edge then has a pair of observations $(\mathcal{W}_{a,b}, y_{a,b})$ associated with it. There are a number of ways in which the complex number $H_{a,b}$ could represent this information, for example, with the real part as $\mathcal{W}$ and the imaginary part as y. However, we would like our complex property matrix to reflect the Laplacian. As a result, the off-diagonal elements of $\mathbf{H}$ are chosen to be

$$H_{a,b} = -\mathcal{W}_{a,b} e^{iy_{ab}}.$$

In other words, the connection weights are encoded by the magnitude of the complex number $H_{a,b}$ and the additional measurement by its phase. By using this encoding, the magnitude of the numbers is the same as in the original Laplacian matrix. Clearly, this encoding is most suitable when the measurements are angles. If the measurements are easily bounded, they can be mapped onto an angular interval and phase wrapping can be avoided. If the measurements are not easily bounded, then this encoding is not suitable.

The measurements must satisfy the conditions $-\pi \leq y_{a,b} < \pi$ and $y_{a,b} = -y_{b,a}$ to produce a Hermitian matrix. To ensure a positive definite matrix, we require $H_{aa} > -\sum_{b \neq a} |H_{ab}|$. This condition is satisfied if

$$H_{aa} = x_a + \sum_{b \neq a} \mathcal{W}_{a,b}$$

$$\text{and } x_a \geq 0.$$

When defined in this way, the property matrix is a complex analogue of the weighted Laplacian for the graph.

For a Hermitian matrix, there is an orthogonal complete basis set of eigenvectors and eigenvalues obeying the eigenvalue equation $\mathbf{He} = \lambda\mathbf{e}$. In the Hermitian case, the eigenvalues λ are real and the components of the eigenvectors $\mathbf{e}$ are complex. There is a potential ambiguity in the eigenvectors in that any multiple of an eigenvector is

also a solution, i.e., $\mathbf{H}\alpha\mathbf{e} = \lambda\alpha\mathbf{e}$. In the real case, we choose α such that $\mathbf{e}$ is of unit length. In the complex case, α itself may be complex and needs to determined by two constraints. We set the vector length to $|\mathbf{e}_i| = 1$ and, in addition, we impose the condition $\arg \sum_{i=1}^{n} \mathbf{e}_i = 0$, which specifies both real and imaginary parts.

This representation can be extended further by using the four-component complex numbers known as quaternions. As with real and complex numbers, there is an appropriate eigendecomposition which allows the spectral matrix to be found. In this case, an edge weight and three additional binary measurements may be encoded on an edge. It is not possible to encode more than one unary measurement using this approach. However, for the experiments in this paper, we have concentrated on the complex representation.

When the eigenvectors are constructed in this way, the spectral matrix is found by performing the eigenvector expansion

$$\mathbf{H} = \sum_{i=1}^{n} \lambda_i \mathbf{e}_i \mathbf{e}_i^{\dagger},$$

where λ_i and $\mathbf{e}_i$ are the n eigenvectors and eigenvalues of the Hermitian matrix $\mathbf{H}$. We construct the complex spectral matrix for the graph $\mathcal{G}$ using the eigenvectors as columns, i.e.,

$$\mathbf{\Phi} = \left(\sqrt{\lambda_1}\mathbf{e}_1, \sqrt{\lambda_2}\mathbf{e}_2, \ldots \sqrt{\lambda_n}\mathbf{e}_n \right). \tag{7}$$

We can again reconstruct the original Hermitian property matrix using the relation $\mathbf{H} = \mathbf{\Phi}\mathbf{\Phi}^{\dagger}$.

Since the components of the eigenvectors are complex numbers, therefore, each symmetric polynomial is complex. Hence, the symmetric polynomials must be evaluated with complex arithmetic and also evaluate to complex numbers. Each S_r therefore has both real and imaginary components. The real and imaginary components of the symmetric polynomials are interleaved and stacked to form a feature-vector $\mathbf{B}$ for the graph. This feature-vector is real.

6 GRAPH EMBEDDING METHODS

We explore three different methods for embedding the graph feature vectors in a pattern space. Two of these are classical methods. Principal components analysis (PCA) finds the projection that accounts for the variance or scatter of the data. Multidimensional scaling (MDS), on the other hand, preserves the relative distances between objects. The remaining method is a newly reported one that offers a compromise between preserving variance and the relational arrangement of the data and is referred to as locality preserving projection [10].

In this paper, we are concerned with the set of graphs $\mathcal{G}_1, \mathcal{G}_2, \ldots, \mathcal{G}_k, \ldots, \mathcal{G}_N$. The kth graph is denoted by $\mathcal{G}_k = (\mathcal{V}_k, \mathcal{E}_k)$ and the associated vector of symmetric polynomials is denoted by $\mathbf{B}_k$.

6.1 Principal Component Analysis

We commence by constructing the matrix $\mathbf{X} = [\mathbf{B}_1 - \bar{\mathbf{B}} | \mathbf{B}_2 - \bar{\mathbf{B}} | \ldots | \mathbf{B}_k - \bar{\mathbf{B}} | \ldots | \mathbf{B}_N - \bar{\mathbf{B}}]$, with the graph feature vectors as columns. Here, $\bar{\mathbf{B}}$ is the mean feature vector for the data set. Next, we compute the covariance matrix for the elements of the feature vectors by taking the matrix product $\mathbf{C} = \mathbf{X}\mathbf{X}^T$. We extract the principal components directions by performing the eigendecomposition $\mathbf{C} =$

$\sum_{i=1}^{N} l_i \mathbf{u}_i \mathbf{u}_i^T$ on the covariance matrix $\mathbf{C}$, where the l_i are the eigenvalues and the $\mathbf{u}_i$ are the eigenvectors. We use the first s leading eigenvectors (2 or 3, in practice, for visualization purposes) to represent the graphs extracted from the images. The coordinate system of the eigenspace is spanned by the s orthogonal vectors $\mathbf{U} = (\mathbf{u}_1, \mathbf{u}_2, \ldots, \mathbf{u}_s)$. The individual graphs represented by the long vectors $\mathbf{B}_k, k = 1, 2, \ldots, N$ can be projected onto this eigenspace using the formula $\mathbf{y}_k = \mathbf{U}^T(\mathbf{B}_k - \bar{\mathbf{B}})$. Hence, each graph G_k is represented by an s-component vector $\mathbf{y}_k$ in the eigenspace.

Linear discriminant analysis (LDA) is an extension of PCA to the multiclass problem. We commence by constructing separate data matrices $\mathbf{X}_1, \mathbf{X}_2, \ldots \mathbf{X}_{N_c}$ for each of the N_c classes. These may be used to compute the corresponding class covariance matrices $\mathbf{C}_i = \mathbf{X}_i \mathbf{X}_i^T$. The average class covariance matrix $\tilde{\mathbf{C}} = \frac{1}{N_c}\sum_{i=1}^{N_c} \mathbf{C}_i$ is found. This matrix is used as a sphering transform. We commence by computing the eigendecomposition

$$\tilde{\mathbf{C}} = \sum_{i=1}^{N} l_i \mathbf{u}_i \mathbf{u}_i^T = \mathbf{U}\Lambda\mathbf{U}^T,$$

where $\mathbf{U}$ is the matrix with the eigenvectors of $\tilde{\mathbf{C}}$ as columns and $\Lambda = \text{diag}(l_1, l_2, \ldots, l_n)$ is the diagonal eigenvalue matrix. The sphered representation of the data is $\mathbf{X}' = \Lambda^{-\frac{1}{2}}\mathbf{U}^T\mathbf{X}$. Standard PCA is then applied to the resulting data matrix $\mathbf{X}'$. The purpose of this technique is to find a linear projection which describes the class differences rather than the overall variance of the data.

6.2 Multidimensional Scaling

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. To commence, we require pairwise distances between graphs. We do this by computing the norms between the spectral pattern vectors for the graphs. For the graphs indexed i_1 and i_2, the distance is $d_{i_1,i_2} = (\mathbf{B}_{i_1} - \mathbf{B}_{i_2})^T (\mathbf{B}_{i_1} - \mathbf{B}_{i_2})$. The pairwise distances d_{i_1,i_2} are used as the elements of an $N \times N$ dissimilarity matrix $\mathbf{R}$, whose elements are defined as follows:

$$R_{i_1,i_2} = \begin{cases} d_{i_1,i_2} & \text{if } i_1 \neq i_2 \\ 0 & \text{if } i_1 = i_2. \end{cases} \tag{8}$$

In this paper, we use the classical multidimensional scaling method [7] 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 rth row, $\hat{d}_{.c}$ is the dissimilarity average value over the cth 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 nonzero eigenvalues. We arrange these k nonzero eigenvalues in descending order, i.e., $l_1 \geq l_2 \geq \ldots \geq l_k > 0$. The corresponding ordered
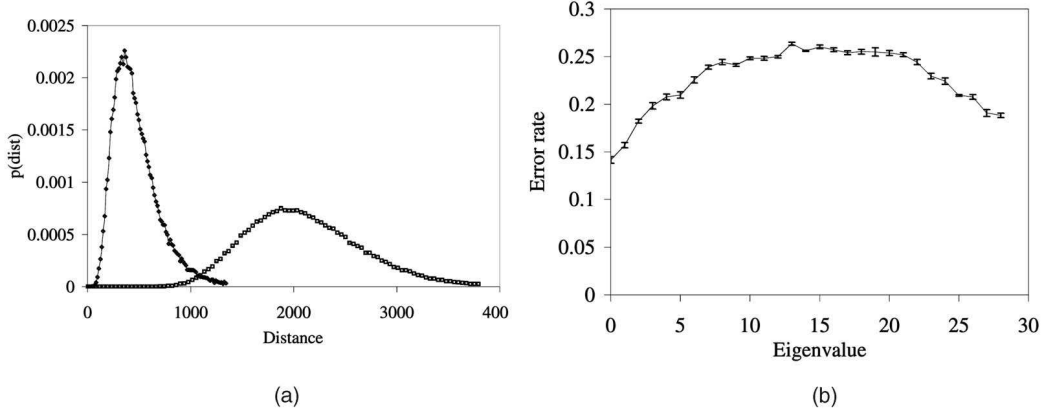
Fig. 1. (a) Distributions of distance to edited graphs and (b) eigenvectors.

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 $\mathbf{y}_j = (Y_{j,1}, Y_{j,2}, \ldots, Y_{j,s})^T$.

6.3 Locality Preserving Projection

Our next pattern space embedding method is He and Niyogi's Locality Preserving Projections(LPP) [10]. LPP is a linear dimensionality reduction method which attempts to project high-dimensional data to a low-dimensional manifold, while preserving the neighborhood structure of the data set. The method is relatively insensitive to outliers and noise. This is an important feature in our graph clustering task since the outliers are usually introduced by imperfect segmentation processes. The linearity of the method makes it computationally efficient.

The graph feature vectors are used as the columns of a data-matrix $\mathbf{X} = (\mathbf{B}_1|\mathbf{B}_2|\ldots|\mathbf{B}_N)$. The relational structure of the data is represented by a proximity weight matrix $\mathbf{W}$ with elements $W_{i_1,i_2} = \exp[-kd_{i_1,i_2}]$, where k is a constant. If $\mathbf{Q}$ is the diagonal degree matrix with the row weights $Q_{k,k} = \sum_{j=1}^{N} W_{k,j}$ as elements, then the relational structure of the data is represented using the Laplacian matrix $\mathbf{J} = \mathbf{Q} - \mathbf{W}$. The idea behind LPP is to analyze the structure of the weighted covariance matrix $\mathbf{X}\mathbf{W}\mathbf{X}^T$. The optimal projection of the data is found by solving the generalized eigenvector problem

$$\mathbf{X}\mathbf{J}\mathbf{X}^T\mathbf{u} = l\mathbf{X}\mathbf{Q}\mathbf{X}^T\mathbf{u}. \tag{9}$$

We project the data onto the space spanned by the eigenvectors corresponding to the s smallest eigenvalues. Let $\mathbf{U} = (\mathbf{u}_1, \mathbf{u}_2, \ldots, \mathbf{u}_s)$ be the matrix with the corresponding eigenvectors as columns, then projection of the kth feature vector is $\mathbf{y}_k = \mathbf{U}^T\mathbf{B}_k$.

7 Experiments

There are three aspects to the experimental evaluation of the techniques reported in this paper. We commence with a study on synthetic data aimed at evaluating the ability of the spectral features to distinguish between graphs under controlled structural error. The second part of the study focuses on real-world data and assesses whether the spectral feature vectors can be embedded in a pattern space that

reveals cluster-structure. Here, we explore the two different applications of view-based polyhedral object recognition and shock graph recognition. The latter application focuses on the use of the complex variant of the property matrix.

7.1 Synthetic Data

We commence by examining the ability of the spectral feature set and distance measure to separate both structurally related and structurally unrelated graphs. This study utilizes random graphs which consist of 30 nodes and 140 edges. The edges are generated by connecting random pairs of nodes. From each seed graph constructed in this way, we generate structurally related graphs by applying random edit operations to simulate the effect of noise. The structural variants are generated from a seed graph by either deleting an edge or inserting a new random edge. Each of these operations is assigned unit cost and, therefore, a graph with a deleted edge has unit edit distance to the original seed graph. In Fig. 1a, we have plotted the distribution of feature vector distance $d_{i_1,i_2} = (\mathbf{B}_{i_1} - \mathbf{B}_{i_2})^T(\mathbf{B}_{i_1} - \mathbf{B}_{i_2})$ for two sets of graphs. The leftmost curve shows the distribution of distances when we take individual seed graphs and remove a single edge at random. The rightmost curve is the distribution of distance obtained when we compare the distinct seed graphs. In the case of the random edge-edits, the modal distance between feature vectors is much less than that for the structurally distinct seed graphs. Hence, the distance between feature vectors appears to provide scope for distinguishing between distinct graphs when there are small variations in edge structure due to noise.

The results presented in Table 1 take this study one step further and demonstrate the performance under different levels of corruption. Here, we have computed the confusion probability, i.e., the overlap between distribution of feature-vector distance for the edited graphs and the seed graphs, as a

TABLE 1
Performance of Feature Set for Edited Graphs

Edit distance	1	2	3	4	5
Confusion prob.	0.015	0.047	0.098	0.144	0.179
Edit distance	6	7	8	9	10
Confusion prob.	0.209	0.234	0.262	0.283	0.289

TABLE 2
Confusion Probabilities of Different
Eigenvector Sets and Edit Distances

Eigenvectors	Edit distance			
	1	2	3	4
First only	0.003	0.100	0.144	0.211
Start 5 and end 5	0.043	0.088	0.132	0.171
All	0.015	0.047	0.098	0.144

function of the number of randomly deleted edges. The confusion probability increases with increasing edit distance.

In Fig. 1b, we investigate the effect of the different eigenmodes used in computing symmetric polynomials. We again compute the confusion probability due to the overlap of the two distance distributions. Here, we confine our attention to the removal of a single edge. Only one eigenvector is used in the distance measure. Eigenvector-n denotes the eigenvector with the nth largest eigenvalue. The plot demonstrates that the leading and the last few eigenvectors give the lowest confusion probabilities and are, hence, the most important in differentiating graph structure.

At first sight, this suggests that only a few of the eigenvectors need be used to characterize the graph. However, as Table 2 reveals, this is only the case for small edit distances. As the edit distance increases, then more of the eigenvectors become important.

In Fig. 2a, we plot the distance in feature space, i.e., $d_{i_1,i_2} = (\mathbf{B}_{i_1} - \mathbf{B}_{i_2})^T (\mathbf{B}_{i_1} - \mathbf{B}_{i_2})$ as a function of the graph edit distance. Although the relationship is not linear, it is monotonic and approximately linear for small edit distances. Hence, the distance between the feature vectors does reflect in a direct way the changes in structure of the graphs.

To provide an illustration, we have created three clusters of graphs from three seed or reference graphs by perturbing them by edit operations. The distances between the feature-vectors for these graphs are then used to embed the graphs in a two-dimensional space using multidimensional scaling. The procedure is as follows: Three seed graphs are generated

at random. These seed graphs are used to generate samples by deleting an edge at random from the graph. The distance matrix is then constructed, which has elements d_{i_1,i_2}, the distance between graphs i_1 and i_2. Finally, the MDS technique is used to visualize the distribution of graphs in a two-dimensional space. The results are shown in Fig. 2b. The graphs form good clusters and are well-separated from one another.

To take this study on synthetic data one step further, we have performed a classification experiment. We have generated 100 graphs of 25 nodes each. For each graph, the edge-structure is randomly generated. Associated with each edge is a weight randomly and uniformly drawn from the interval $[0, 1]$. We have investigated the effect of adding random noise to the edge-weights. The weight noise is drawn from a Gaussian distribution of zero mean and known standard deviation.

We have investigated the effect of this noise on three different vector representations of the weighted graphs. The first of these is a vector with the first four polynomial features as components. The second is a vector whose components are the bin-contents of the normalized edge-weight histogram. Here, the edge weights are allocated to eight uniformly spaced bins. The final vector has the leading four eigenvalues of the Laplacian matrix as components. We have computed the distances between the feature vectors for the uncorrupted and noise corrupted graphs. To compute a classification error rate, we have recorded the fraction of times that the uncorrupted graphs do not have the smallest distance to the corresponding noise corrupted graph. In Fig. 3, we show the error-rate as a function of the edge-weight noise standard deviation. The main features to note from this plot are that the lowest error rate is returned by the polynomial features and the highest error rate results from the use of the edge-weight histogram.

7.2 View-Based Object Recognition

Our first real-world experimental vehicle is provided by 2D views of 3D objects. 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 Fig. 4,
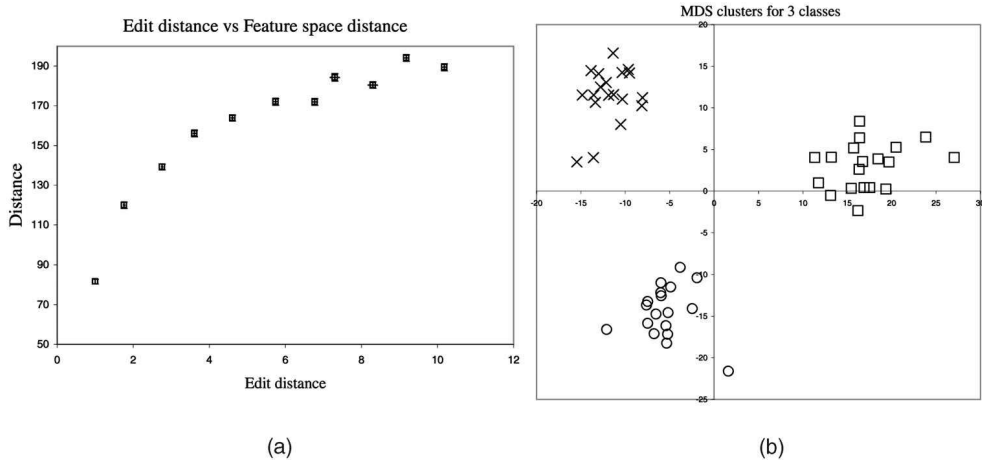

(a)   (b)

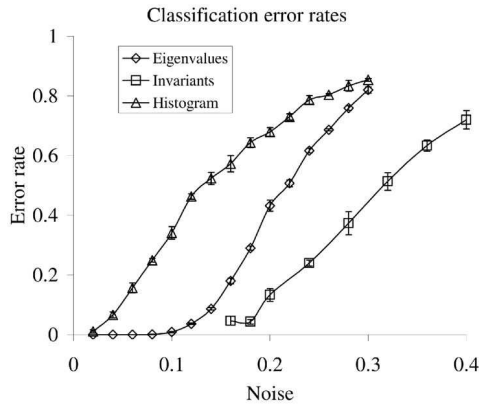Fig. 2. (a) Distance between feature vectors versus graph edit distance and (b) MDS applied to three random graph-sets.

Fig. 3. Classification error rates.

Fig. 4. Example images from the CMU, MOVI, and chalet sequences and their corresponding graphs.

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 Fig. 5, we compare the results obtained with the different embedding strategies and the different graph features. In Column 1, we show the results obtained when PCA is used, Column 2 when MDS is used, and Column 3 when LPP is used. The top row shows the results obtained using a standard spectral feature, namely, the spectrum ordered eigenvalues of the Laplacian, i.e., $\mathbf{B}_k = (\lambda_1^k, \lambda_2^k, \ldots)^T$. The second row shows the results obtained when the symmetric polynomials are computed using the spectral matrix for the Laplacian. The different image sequences are displayed in various shades of gray and black.

There are a number of conclusions that can be drawn from this plot. First, the most distinct clusters are produced when either MDS or LPP are used. Second, the best spectral features seem to be the symmetric polynomials computed from the Laplacian spectral matrix. To display the cluster-structure obtained, in Fig. 6, we visualize the results obtained using LPP and the Laplacian polynomials by placing thumbnails of the original images in the space
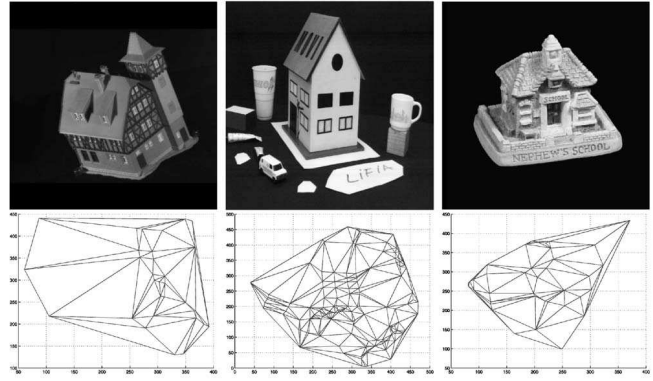
spanned by the leading two eigenvectors. The different objects form well-separated and compact clusters.

To take this study further, we investigate the effects of applying the embedding methods to two of the sequences separately. Fig. 7 shows the results obtained with the Laplacian polynomials for MOVI and Chalet sequences. Column 1 is for PCA, Column 2 is for MDS, and Column 3 is for LPP. In the case of both image sequences, LPP results in a smooth trajectory as the different views in the sequence are traversed.

7.3 Shock Graphs

The final example focuses on the use of the complex property matrix representation and is furnished by shock trees which are a compact representation of curved boundary shape. There are a number of ways in which a shape graph can be computed [15], [35]. One of the most effective recent methods has been to use the Hamilton-Jacobi equations from classical mechanics to solve the eikonal equation for inward boundary motion [35]. The skeleton is the set of singularities in the boundary motion where opposing fronts collide and these can be characterized as locations where the divergence of the
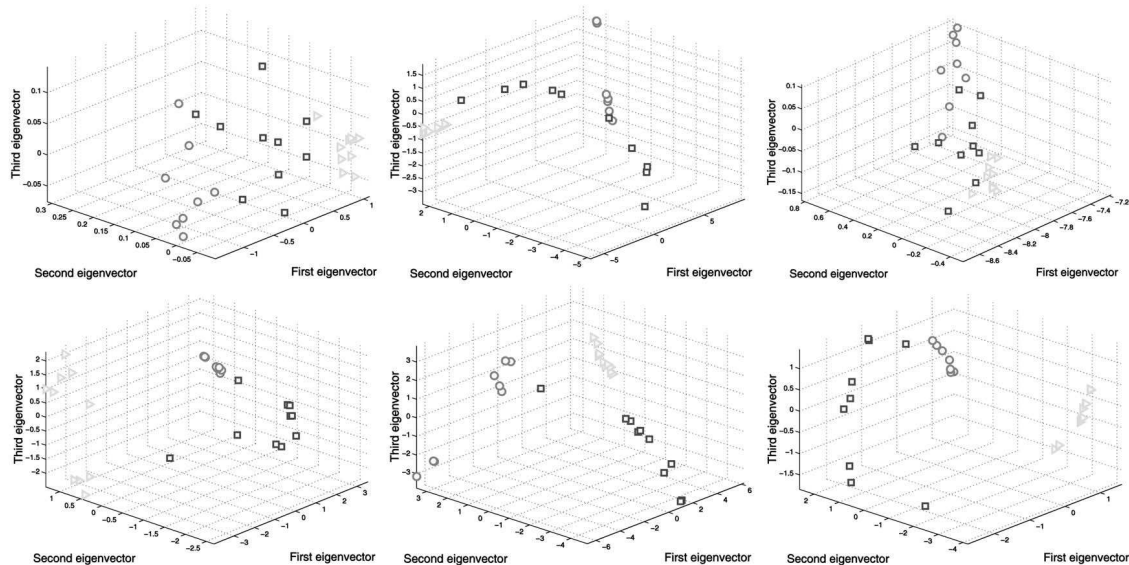

Fig. 5. Clustering CMU, MOVI, and Chalet sequences, Row 1: Eigenvalues. Row 2: Laplacian Matrix Polynomials. Column 1: PCA, Column 2: MDS, and Column 3: LPP.
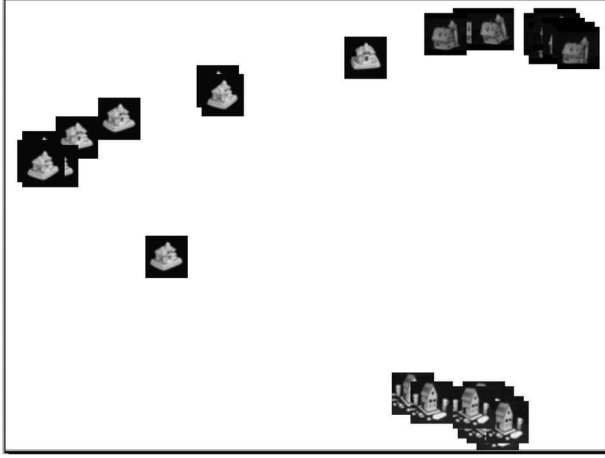
Fig. 6. LPP space clustering using the graph polynomials.

distance map to the boundary is nonzero. Once detected, then the skeleton can be converted into a tree. There are several ways in which this can be affected. For instance, Kimia et al. use the branches and junctions of the skeleton as the nodes and edges of a tree [15]. Siddiqi et al. use the time of formation of singularities under the eikonal equation to order the points on the skeleton [35]. According to this representation, the root of the tree is located at the deepest point within the shape, i.e., the final skeleton point to be formed and the terminal nodes are the closest to the boundary. Once the tree is to hand, then it may be attributed with a number of properties of the boundaries. These might include the radius of the bitangent circle or the rate of change of skeleton length with boundary length. The tree may also be attributed with shock labels. These capture the differential structure of the boundary since they describe whether the radius of the bitangent circle is constant, increasing, decreasing, locally minimum, or locally maximum. The labels hence indicate whether the boundary is of constant width, tapering-out, tapering-in, locally constricted, or locally bulging.

7.3.1 Tree Attributes

Before processing, the shapes are firstly normalized in area and aligned along the axis of largest moment. After performing these transforms, the distances and angles associated with the different shapes can be directly compared and can therefore be used as attributes.

To abstract the shape skeleton using a tree, we place a node at each junction point in the skeleton. The edges of the tree represent the existence of a connecting skeletal branch between pairs of junctions. The nodes of the tree are characterized using the radius $r(a)$ of the smallest bitangent circle from the junction to the boundary. Hence, for the node a, $x_a = r(a)$. The edges are characterized by two measurements. For the edge (a, b), the first of these, $y_{a,b}$ is the angle between the nodes a and b, i.e., $y_{a,b} = \theta(a, b)$. Since most skeleton branches are relatively straight, this is an approximation to the angle of the corresponding skeletal branch. Furthermore, since $-\pi \leq \theta(a, b) < \pi$ and $\theta(a, b) = -\theta(b, a)$, the measurement is already suitable for use in the Hermitian property matrix.

In order to compute edge weights, we note that the importance of a skeletal branch may be determined by the rate of change of boundary length l with skeleton length s [37], which we denote by dl/ds. This quantity is related to the rate of change of the bitangent circle radius along the skeleton, i.e., dr/ds, by the formula

$$\frac{dl}{ds} = \sqrt{1 - \left(\frac{dr}{ds}\right)^2}.$$

The edge weight $\mathcal{W}_{a,b}$ is given by the average value of dl/ds along the relevant skeletal branch.

It must be noted that the tree representation adopted here is relatively simplistic and more sophisticated alternatives exist elsewhere in the literature [15], [35]. It must be stressed that our main aim here is to furnish a simple attributed tree representation on which to test our spectral analysis.

It is a well-known fact that trees tend to be cospectral [4] which, as noted in Section 3, could be problematic for the spectral representation. However, because we are using
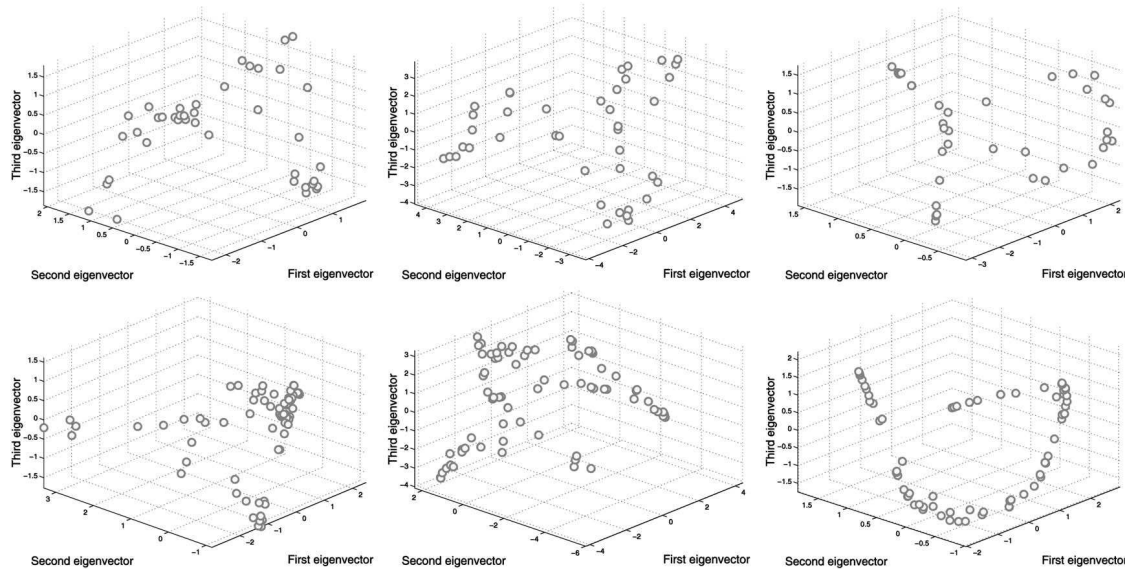

Fig. 7. Comparison of the MOVI and Chalet sequences with various projection methods. Column 1: PCA, Column 2: MDS, and Column 3: LPP.
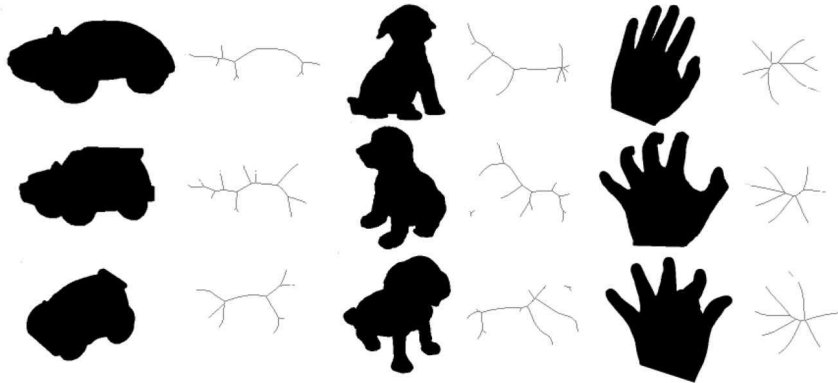
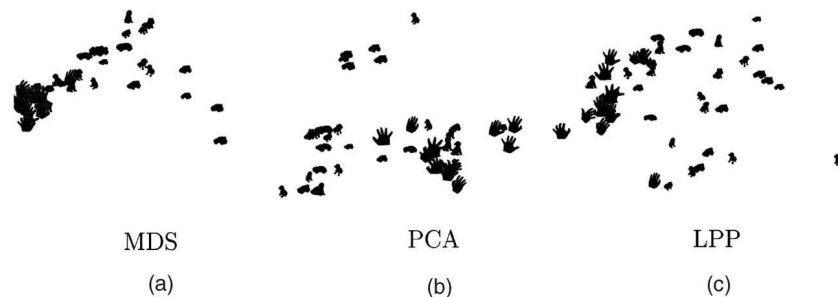Fig. 8. Examples from the shape database with their associated skeletons.

Fig. 9. (a) MDS, (b) PCA, and (c) LPP applied to the shock graphs.

attributed trees, any ambiguity is removed by the measurement information.

7.3.2 Experiments with Shock Trees

Our experiments are performed using a database of 42 binary shapes. Each binary shape is extracted from a 2D view of a 3D object. There are three classes in the database and, for each object, there are a number of views acquired from different viewing directions and a number of different examples of the class. We extract the skeleton from each binary shape and attribute the resulting tree in the manner outlined in Section 7.3.1. Fig. 8 shows some examples of the types of shape present in the database along with their skeletons.

We commence by showing some results for the three shapes shown in Fig. 8. The objects studied are a hand, some puppies, and some cars. The dog and car shapes consist of a number of different objects and different views of each object. The hand category contains different hand configurations. We apply the three embedding strategies outlined in Section 6 to the vectors of permutation invariants extracted from the Hermitian variant of the Laplacian. We commence in Fig. 9a by showing the result of applying the MDS procedure to the three shape categories. The "hand" shapes form a compact cluster in the MDS space. There are other local clusters consisting of three or four members of the remaining two classes. This reflects the fact that, while the hand shapes have very similar shock graphs, the remaining two categories have rather variable shock graphs because of the different objects.

Fig. 9b shows the result of using PCA. Here, the distributions of shapes are much less compact. While a distinct cluster of hand shapes still occurs, they are generally more dispersed over the feature space. There are some distinct clusters of the car shape, but the distributions overlap more in the PCA projection when compared to the MDS space.

Fig. 9c shows the result of the LPP procedure on the data set. The results are similar to the PCA method.

One of the motivations for the work presented here was the potential ambiguities that are encountered when using the spectral features of trees. To demonstrate the effect of using attributed trees rather than simply weighting the edges, we have compared the LDA projections using both types of data. Fig. 10 illustrates the result of this comparison. Fig. 10b shows the result obtained using the symmetric polynomials from the eigenvectors of the Laplacian matrix $\mathbf{L} = D - \mathbf{W}$, associated with the edge weight matrix. Fig. 10a shows the result of using the using the Hermitian property matrix. The Hermitian property matrix for the attributed trees produces a better class separation than the Laplacian matrix for the weighted trees. The separation can be measured by the Fisher discriminant between the classes, which is the squared distance between class centers divided by the variance along the line joining the centers. For the Hermitian property matrix, the separations are 1.12 for the car/dog classes, 0.97 for the car/hand, and 0.92 for the dog/hand. For the weighted matrix, the separations are 0.77 for the car/dog, 1.02 for the car/hand, and 0.88 for the dog/hand.

8   CONCLUSIONS

In this paper, we have shown how graphs can be converted into pattern vectors by utilizing the spectral decomposition of the Laplacian matrix and basis sets of symmetric polynomials. These feature vectors are complete, unique, and continuous. However, and most important of all, they are permutation invariants. We investigate how to embed the vectors in a pattern space, suitable for clustering the graphs. Here, we explore a number of alternatives including
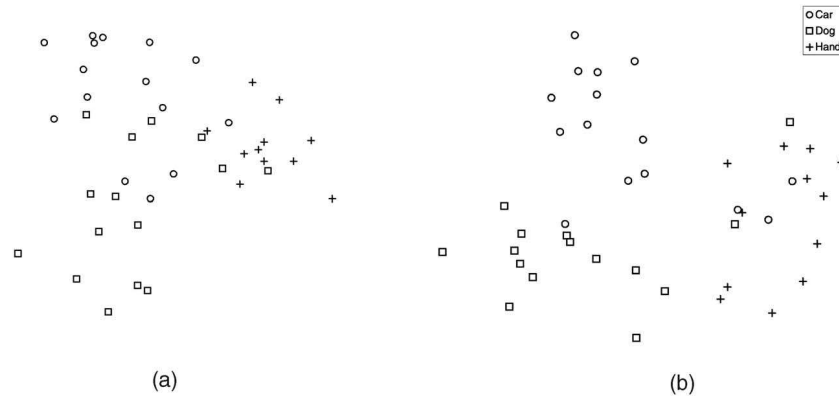
Fig. 10. A comparison of attributed trees with weighted trees. (a) Trees with edge weights based on boundary lengths. (b) Attributed trees with additional edge angle information.

PCA, MDS, and LPP. In an experimental study, we show that the feature vectors derived from the symmetric polynomials of the Laplacian spectral decomposition yield good clusters when MDS or LPP are used.

There are clearly a number of ways in which the work presented in this paper can be developed. For instance, since the representation based on the symmetric polynomials is complete, they may form the means by which a generative model of variations in graph structure can be developed. This model could be learned in the space spanned by the permutation invariants and the mean graph and its modes of variation reconstructed by inverting the system of equations associated with the symmetric polynomials.

ACKNOWLEDGMENTS

REFERENCES

[1] A.D. Bagdanov and M. Worring, "First Order Gaussian Graphs for Efficient Structure Classification," *Pattern Recogntion,* vol. 36, pp. 1311-1324, 2003.

[2] M. Belkin and P. Niyogi, "Laplacian Eigenmaps for Dimensionality Reduction and Data Representation," *Neural Computation,* vol. 15, no. 6, pp. 1373-1396, 2003.

[3] N. Biggs, *Algebraic Graph Theory.* Cambridge Univ. Press, 1993.

[4] P. Botti and R. Morris, "Almost All Trees Share a Complete Set of Inmanantal Polynomials," *J. Graph Theory,* vol. 17, pp. 467-476, 1993.

[5] W.J. Christmas, J. Kittler, and M. Petrou, "Structural Matching in Computer Vision Using Probabilistic Relaxation," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 17, pp. 749-764, 1995.

[6] F.R.K. Chung, "Spectral Graph Theory," *Am. Math. Soc. Edition.,* CBMS series 92, 1997.

[7] T. Cox and M. Cox, *Multidimensional Scaling.* Chapman-Hall, 1994.

[8] D. Cvetkovic, P. Rowlinson, and S. Simic, *Eigenspaces of Graphs.* Cambridge Univ. Press, 1997.

[9] S. Gold and A. Rangarajan, "A Graduated Assignment Algorithm for Graph Matching," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 18, pp. 377-388, 1996.

[10] X. He and P. Niyogi, "Locality Preserving Projections," *Advances in Neural Information Processing Systems 16,* MIT Press, 2003.

[11] D. Heckerman, D. Geiger, and D.M. Chickering, "Learning Bayesian Networks: The Combination of Knowledge and Statistical Data," *Machine Learning,* vol. 20, pp. 197-243, 1995.

[12] I.T. Jolliffe, *Principal Components Analysis.* Springer-Verlag, 1986.

[13] R. Kannan et al., "On Clusterings: Good, Bad and Spectral," *Proc. 41st Symp. Foundation of Computer Science,* pp. 367-377, 2000.

[14] Y. Kesselman, A. Shokoufandeh, M. Demerici, and S. Dickinson, "Many-to-Many Graph Matching via Metric Embedding," *Proc. IEEE Conf. Computer Vision and Pattern Recognition,* pp. 850-857, 2003.

[15] B.B. Kimia, A.R. Tannenbaum, and S.W. Zucker, "Shapes, Shocks, and Deformations I. The Components of 2-Dimensional Shape and the Reaction-Diffusion Space," *Int'l J. Computer Vision,* vol. 15, pp. 189-224, 1995.

[16] S. Kosinov and T. Caelli, "Inexact Multisubgraph Matching Using Graph Eigenspace and Clustering Models," *Proc. Joint IAPR Int'l Workshops Structural, Syntactic, and Statistical Pattern Recognition, SSPR 2002 and SPR 2002,* pp. 133-142, 2002.

[17] L. Lovasz, "Random Walks on Graphs: A Survey," *Bolyai Soc. Math. Studies,* vol. 2, pp. 1-46, 1993.

[18] B. Luo and E.R. Hancock, "Structural Matching Using the EM Algorithm and Singular Value Decomposition," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 23, pp. 1120-1136, 2001.

[19] B. Luo, A. Torsello, A. Robles-Kelly, R.C. Wilson, and E.R. Hancock, "A Probabilistic Framework for Graph Clustering," *Proc. IEEE Conf. Computer Vision and Pattern Recognition,* pp. 912-919, 2001.

[20] B. Luo, R.C. Wilson, and E.R. Hancock, "Eigenspaces for Graphs," *Int'l J. Image and Graphics,* vol. 2, pp. 247-268, 2002.

[21] B. Mohar, "Laplace Eigenvalues of Graphs—A Survey," *Discrete Math.,* vol. 109, pp. 171-183, 1992.

[22] A. Munger, H. Bunke, and X. Jiang, "Combinatorial Search vs. Genetic Algorithms: A Case Study Based on the Generalized Median Graph Problem," *Pattern Recognition Letters,* vol. 20, pp. 1271-1279, 1999.

[23] M. Pavan and M. Pelillo, "Dominant Sets and Hierarchical Clustering," *Proc. Ninth IEEE Int'l Conf. Computer Vision,* vol. I, pp. 362-369, 2003.

[24] P. Perona and W.T. Freeman, "A Factorization Approach to Grouping," *Proc. European Conf. Computer Vision,* pp. 655-670, 1998.

[25] S. Rizzi, "Genetic Operators for Hierarchical Graph Clustering," *Pattern Recognition Letters,* vol. 19, pp. 1293-1300, 1998.

[26] A. Robles-Kelly and E.R. Hancock, "An Expectation-Maximisation Framework for Segmentation and Grouping," *Image and Vision Computing,* vol. 20, pp. 725-738, 2002.

[27] A. Robles-Kelly and E.R. Hancock, "Edit Distance from Graph Spectra," *Proc. Ninth IEEE Int'l Conf. Computer Vision,* vol. I, pp. 127-135, 2003.

[28] S. Roweis and L. Saul, "Non-Linear Dimensionality Reduction by Locally Linear Embedding," *Science,* vol. 299, pp. 2323-2326, 2002.

[29] S. Sarkar and K.L. Boyer, "Quantitative Measures of Change Based on Feature Organization: Eigenvalues and Eigenvectors," *Computer Vision and Image Understanding,* vol. 71, pp. 110-136, 1998.

[30] G.L. Scott and H.C. Longuet-Higgins, "Feature Grouping by Relocalisation of Eigenvectors of the Proximity Matrix," *Proc. British Machine Vision Conf.,* pp. 103-108, 1990.

[31] J. Segen, "Learning Graph Models of Shape," *Proc. Fifth Int'l Conf. Machine Learning,* J. Laird, ed., pp. 29-25, 1988.

[32] K. Sengupta and K.L. Boyer, "Organizing Large Structural Modelbases," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 17, 1995.

[33] J. Shi and J. Malik, "Normalized Cuts and Image Segmentation," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 22, pp. 888-905, 2000.

[34] A. Shokoufandeh, S. Dickinson, K. Siddiqi, and S. Zucker, "Indexing Using a Spectral Coding of Topological Structure," *Proc. IEEE Conf. Computer Vision and Pattern Recognition,* pp. 491-497, 1999.

[35] K. Siddiqi, A. Shokoufandeh, S.J. Dickinson, and S.W. Zucker, "Shock Graphs and Shape Matching," *Int'l J. Computer Vision,* vol. 35, pp. 13-32, 1999.

[36] J.B. Tenenbaum, V.D. Silva, and J.C. Langford, "A Global Geometric Framework for Non-Linear Dimensionality Reduction," *Science,* vol. 290, pp. 586-591, 2000.

[37] A. Torsello and E.R. Hancock, "A Skeletal Measure of 2D Shape Similarity," *Proc. Fourth Int'l Workshop Visual Form,* pp. 594-605, 2001.

[38] S. Umeyama, "An Eigen Decomposition Approach to Weighted Graph Matching Problems," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 10, pp. 695-703, 1988.

[39] B.J. van Wyk and M.A. van Wyk, "Kronecker Product Graph Matching," *Pattern Recognition,* vol. 39, no. 9, pp. 2019-2030, 2003.

[40] A.K.C. Wong, J. Constant, and M.L. You, "Random Graphs," *Syntactic and Structural Pattern Recognition,* World Scientific, 1990.

Richard C. Wilson received the BA degree in physics from the University of Oxford in 1992. In 1996, he completed the PhD degree at the University of York in the area of pattern recognition. From 1996 to 1998, he worked as a research associate at the University of York. After a period of postdoctoral research, he was awarded an Advanced Research Fellowship in 1998. In 2003, he took up a lecturing post and he is now a reader in the Department of Computer Science at the University of York. He has published more than 100 papers in journals, edited books, and refereed conferences. He received an outstanding paper award in the 1997 Pattern Recognition Society awards and has won the best paper prize in ACCV 2002. He is currently an associate editor of the journal *Pattern Recognition*. His research interests are in statistical and structural pattern recognition, graph methods for computer vision, high-level vision, and scene understanding. He is a member of the IEEE Computer Society.

Edwin R. Hancock studied physics as an undergraduate at the University of Durham and graduated with honors in 1977. He remained at Durham to complete the PhD degree in the area of high-energy physics in 1981. Following this, he worked for 10 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). 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 optimization 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 databases, and statistical learning theory. He has published more than 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. He 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*.

Bin Luo received the BEng degree in electronics and the MEng degree in computer science from Anhui university of China in 1984 and 1991, respectively. From 1996 to 1997, he worked as a British Council visiting scholar at the University of York under the Sino-British Friendship Scholarship Scheme (SBFSS). In 2002, he was awarded the PhD degree in computer science from the University of York, United Kingdom. He is at present a professor at Anhui University of China. He has published more than 60 papers in journals, edited books, and refereed conferences. His current research interests include graph spectral analysis, large image database retrieval, image and graph matching, statistical pattern recognition, and image feature extraction.

▷ **For more information on this or any other computing topic, please visit our Digital Library at** www.computer.org/publications/dlib.