Eigenspaces for Graphs

Bin Luo
University of York,York, UK
and Anhui university, PR China.

Richard Wilson
University of York,York, UK.

Edwin Hancock
University of York,York, UK.
erh@cs.york.ac.uk

Abstract

In this paper, we investigate the feasibility of using graph-based descriptions to learn the view structure of 3D objects. The graphs used in our study are constructed from the Delaunay triangulations of corner features. The investigation is divided into two parts. We commence by considering how relational structures can be encoded in a way which can be used to generate parametric eigenspaces. Here we investigate four different relational representations derived from the graphs. The first three of these are vector encodings of the adjacency graph, the weighted adjacency graph, and the point proximity matrix; the fourth representation is the edge weight histogram. We study the eigenspaces which result from these different representations. In addition, we investigate how multidimensional scaling may be used to generate eigenspaces from a set of pairwise distances between graphs.

1 Introduction

View based object recognition has been studied in the computer vision literature for over three decades [29, 9]. Stated simply, the idea is to compile a series of images of an object as the set of possible viewing directions is spanned. The images are then subjected to some form of dimensionality reduction [11] or information abstraction [9]. This is a process of learning [14] that may involve either feature extraction, principal components analysis or the abstraction of the main structures using a relational description [21]. Once a condensed image representation is to hand, then the aim is to embed the different images in a low-dimensional representation which can be traversed with viewing direction. Recognition and pose recovery may be effected by finding the closest representative view. In other words, the aim is to embed high-dimensional view based image data in a low dimensional structure which is suitable for view indexing.

Broadly speaking there are two different approaches to this problem. The first of these is to construct an eigenspace [11]. This approach was first introduced by Murase and Nayar [11], and has since been refined in a number of different ways [16, 20]. The idea is to perform principal components analysis on the images collected as the viewing direction and illumination direction [1] are varied. This is achieved by first storing each image as a long-vector. Next the covariance matrix for the long-vectors is found. The eigenvectors of the covariance matrix define the directions of principal components in the space spanned by the long-vectors. Dimensionality reduction is achieved by projecting the original images onto the principal component directions and selecting the components corresponding to the leading eigenvectors. The method has mainly been applied to pixel based image representations.

The second approach to the problem is older and involves constructing a relational abstraction of the features present in the raw images [15, 29]. The aim here is to extract surfaces or boundary groupings from 2.5D range data or 2D image data. From this data the view occurrence of the different image structures is noted. Hence a group of images which all yield the same feature configuration are deemed to belong to a common view [7]. View indexing can be achieved by matching a relational arrangement of image structures to the set of corresponding representative view graphs. This approach to the problem has its origins in the work of Freeman on characteristic views. It has also stimulated the study of aspect graphs [9, 19, 27]. The topic draws heavily on work from psychology [2, 3] and differential topology [25, 15].

In this paper, we aim to explore a synthesis of these two methodologies. Our aim is to investigate whether it is possible to generate view-based eigenspaces using relational graphs. We study 3D polyhedral objects viewed from different directions. The features used in our study are corners. Our graphs are obtained by locating the Delaunay triangulations of these point-features. We explore two different approaches to constructing eigenspaces. The first of these involves encoding the adjacency matrix as a long-vector and repeating the Murase and Nayer[11] analysis. Here we investigate various representations of the adjacency structure of the graphs. These include vectors of weighted and unweighted adjacency indicators, a vector of point proximity weights and a normalised histogram of proximity weights. Our second approach to the problem is to use multidimensional scaling to embed a set of pairwise graph-distances into a low-dimensional space.

We compare the results from the different embedding strategies and the different adjacency representations on both synthetic and real-world data. Our aim here is to determine whether the view-trajectories in the resulting eigenspaces are well-ordered and can hence be used to index pose.

2 Image representation

We are interested in learning the view structure from a set of images $I_1, I_2, .., I_i, ..., I_N$ whose point-features have been abstracted using Delaunay graphs. Suppose that the features in the image I_i have been abstracted using the graph $G_i = (V_i, E_i)$. Here V_i is denotes the of nodes, i.e. the index-set for the point-features and $E_i \subseteq V_i \times V_i$ is the edge-set for the Delaunay graph. The graph index $i = 1, ...N$ runs over the set of images in their view-order.

We have experimented in four different representations of the structure of the graph G_i.

2.1 Adjacency matrices

Our first representation is based on the adjacency matrix A_i for the graph G_i. This is a $|V_i| \times |V_i|$ matrix whose element with row index a and column index b is

$$A_i(a,b) = \begin{cases} 1 & \text{if } (a,b) \in E_i \\ 0 & \text{otherwise} \end{cases} \quad (1)$$

We adopt a long-vector representation for the adjacency matrix. This is obtained by stacking the columns of the matrix A_i in order. The resulting vector is $B_i = (A_i(1,1), A_i(1,2), ..., A_i(1,|V_i|), A_i(2,1)....., A_i(|V_i|,1), ... A_i(|V_i|,|V_i|)^T$. Hence, each entry in the long-vector corresponds to a different edge in the graph. This representation is only meaningful provided that the order of the nodes in the different graphs is identical and that they contain the same numbers of nodes.

2.2 Weighted adjacency matrices

Our second representation involves weighting the edges. We do this by associating a weight with each edge which is determined by the distance between the pair of corresponding corner features in the image I_i. If $r(a,b)$ is the distance between the corner features a and b, then the weight associated with the edge (a,b) is

$$A_i(a,b) = \begin{cases} exp(-r(a,b)^2/\sigma^2) & \text{if } (a,b) \in E_i \\ 0 & \text{otherwise} \end{cases} \quad (2)$$

The columns of the resulting weighted adjacency matrix are again stacked to form a long-vector B_i.

2.3 Point proximity matrices

The third relational representation of the corner feature uses a point proximity matrix. Here we use the weighting function described above, but apply it to all pairs of corner features irrespective of whether they are connected by an edge of the Delaunay graph. The entry in the proximity matrix for the pair of corners a and b is

$$A_i(a,b) = exp(-r(a,b)^2/\sigma^2) \quad (3)$$

2.4 Weight histogram

Our aim is to use the long-vectors extracted from the adjacency representations for the corner features from different views to generate an eigenspace. This involves computing the covariance matrix for the different entries in the long-vectors. However, in order to be statistically meaningful, the entries in the long-vectors must be in correspondence with one another.

To overcome this problem, if correspondence information is not available, we have investigated the use of weight histograms. The weight-function $W : E_i \rightarrow [0,1]$ assigns to each edge a weight from the interval $[0,1]$. This interval can be divided into a number of contiguous but non-overlapping intervals R_α, $\alpha = 1, ..., K$. Hence, $[0,1] = \cup_{\alpha=1}^{K} R_\alpha$ and $R_\alpha \cap R_\beta = \emptyset$ if $\alpha \neq \beta$. We can associate with each interval R_α a bin $H(\alpha)$. For the graph G_i, the histogram bin-contents is incremented as follows:

$$H(\alpha) = \begin{cases} H(\alpha) + A_i(a,b) & \text{if } A_i(a,b) \in R_\alpha \\ H(\alpha) & \text{otherwise} \end{cases} \quad (4)$$

From the bin-contents, we compute a normalised histogram whose bin-contents is

$$h_i(\alpha) = \frac{H_i(\alpha)}{\sum_{\alpha=1}^{K} H_i(\alpha)} \quad (5)$$

We convert this histogram into a vector $B_i = (H_i(1), ..., H_i(K))^T$.

This procedure can be applied to both the weighted adjacency matrix or the proximity matrix.

3 Learning view structure

In this section we describe two methods from embedding graphs in eigenspaces. The first of these involves performing principal components analysis on the covariance matrices for the long-vectors of the adjacency matrix or the weight histograms. The second method involves performing multidimensional scaling on a set of pairwise distance between graphs.

3.1 Eigendecomposition of the image representation matrices

Our first method makes use of the parametric eigenspace idea of Murase and Nayar [11, 22, 24]. Specifically, we aim to generate parametric eigenspaces from the representations outlined in Section 2.

The relational data for each image is vectorised in the way outlined in Section 2. The N different image vectors are arranged in view order as the columns of the matrix $S = [B_1|B_2|\ldots|B_i|\ldots|B_N]$

Next, we compute the covariance matrix for the elements in the different rows of the matrix S. This is found by taking the matrix product $C = SS^T$ We extract the principal components directions for the relational data by performing an eigendecomposition on the covariance matrix C. The eigenvalues λ_i are found by solving the eigenvalue equation $|C - \lambda I| = 0$ and the corresponding eigenvalues $\vec{e}_i$ are found by solving the eigenvector equation $C\vec{e}_i = \lambda_i \vec{e}_i$

We use the first 3 leading eigenvectors to represent the graphs extracted from the images. The co-ordinate system of the eigenspace is spanned by the three orthogonal vectors by $E = (\vec{e}_1, \vec{e}_2, \vec{e}_3)$. The individual graphs represented by the long vectors $B_i, i = 1, 2, \ldots, N$ can be projected onto

this eigenspace using the formula $\vec{x}_i = \vec{e}^T B_i$ Hence each graph G_i is represented by a 3-component vector $\vec{x}_i$ in the eigenspace.

At this point, it is worth pausing to consider the meaning of the covariance matrix used to construct the different eigenspaces. When the vector B_i represents the adjacency structure of the graph, then the element k, l of the covariance matrix is given by

$$C(k,l) = \sum_{i=1}^{N} B_i(k) B_i(l) \tag{6}$$

Which is simply the co-occurrence frequency of the edges indexed k and l in the set of N view graphs.

In the case of the normalised weight histogram, the covariance matrix element is

$$C(k,l) = \sum_{i=1}^{N} h_i(k) h_i(l) \tag{7}$$

Which is correlation co-efficient for the bins k and l of the weight-histogram over the set of view graphs.

3.2 Multidimensional Scaling

Multidimensional scaling(MDS)[5] is a procedure which allows data specified in terms of a matrix of pairwise distances to be embedded in a Euclidean space. The classical multidimensional scaling method was proposed by Torgenson[31] and Gower[13]. Shepard and Kruskal developed another kind of scaling technique called ordinal scaling[12]. 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. There are many ways in which graph similarity can be measured. The alternatives include graph edit distance [8, 26, 4, 23], probabilistic similarity measures [6, 30], and quadratic distance measures [28, 10].Here we use two different methods. The first of these uses a simple matrix method to compute the distance between pairs of graphs corresponding to different viewpoints [17] For the graphs G_{i1} and G_{i2} the similarity measure is

$$d_{i1,i2} = Tr[A_{i1}^T (I - Q) A_{i2} Q^T] \tag{8}$$

where Q is a $|V_{i1}| \times |V_{i2}|$ matrix of correspondence indicators and I is a matrix whose entries are all unity.

The second method used to compute the distances between the graphs makes use of the weight-histogram. Here the distance between the graphs is simply the Euclidean distance between the normalised histogram bin-contents for the two graphs, i.e.

$$d_{i1,i2} = \sum_{\alpha=1}^{K} \Big[h_{i1}(\alpha) - h_{i2}(\alpha) \Big]^2 \tag{9}$$

The pairwise similarities $d_{1i,i2}$ are used as the elements of an $N \times N$ disimilarity matrix D, whose elements are defined as follows

$$D_{i1,i2} = \begin{cases} d_{i1,i2} & \text{if } i_1 \neq i2 \\ 0 & \text{if } i1 = i2 \end{cases} \tag{10}$$

In this paper, we use the classical multidimensional scaling method to embed our the view-graphs in a Euclidean space using the matrix of pairwise dissimilarities D. The first step of MDS is to calculate a matrix T whose element with row r and column c is given by

$$T_{rc} = -\frac{1}{2}[d_{rc}^2 - \hat{d}_{r.}^2 - \hat{d}_{.c}^2 + \hat{d}_{..}^2] \tag{11}$$

where

$$\hat{d}_{r.} = \frac{1}{N} \sum_{c=1}^{N} d_{rc} \tag{12}$$

is the average dissimilarity value over the rth row, $\hat{d}_{.c}$ is the similarly defined average value over the cth column and

$$\hat{d}_{..} = \frac{1}{N^2} \sum_{r=1}^{N} \sum_{c=1}^{N} d_{r,c} \tag{13}$$

is the average similarity value over all rows and columns of the similarity matrix T.

We subject the matrix T to an eigenvector analysis to obtain a matrix of embedding co-ordinates X. If the rank of 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. $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_k > 0$. The corresponding ordered eigenvectors are denoted by $\vec{e}_i$ where λ_i is the ith eigenvalue. The embedding co-ordinate system for the graphs obtained from different views is $X = [\vec{f}_1, \vec{f}_2, \ldots, \vec{f}_k]$ where $\vec{f}_i = \sqrt{\lambda_i}\vec{e}_i$ are the scaled eigenvectors. For the graph indexed i, the embedded vector of co-ordinates is $\vec{x}_i = (X_{i,1}, X_{i,2}, X_{i,3})^T$

3.3 Contrasting the Methods

Before concluding this section, we pause to compare the two eigendecomposition methods. Both aim to embed the graphs in a low-dimensional space. In the case of the projection method described in Section 3.1, the eigenspace is generated from the covariance matrix for the individual features contained within the graphs over the different views. These features may be either individual edges, in the case of the long-vector representation of the adjacency matrix, or weight frequencies, in the case of the histograms. Hence, the axes of the representation reflect the most salient distinguishing features of the different graphs. The original data is then projected onto this axis system for the purposes of constructing the view-space.

In the case of the multidimensional scaling method outlined in Section 3.2, the measure used to construct the eigenspace is one of graph similarity, rather than feature co-occurrence. The graphs are embedded in the eigen-space in such a way as to reflect the pattern of pairwise similarities.

4 Experiments

We have used two image sequences in our study. The first of these is the set of synthetic images shown in Figure 1. This is a set of perspective views of a house as it rotates. The associated graphs are shown in Figure 2. It is important

to note that although the number of feature-points in this sequence remains the same, there are significant structural differences in the graphs in the different views. In addition, because we have synthesised these images, the correspondences between feature points in the different views is known. In Figure 3, we show a second real world sequence of images. This sequence is taken from the CMU/VASC data-base. Figure 4 shows the Delaunay graphs for the second sequence. Here the feature points have been detected using the corner detector of [18]. In this sequence there are different numbers of feature points in the different views. In addition, we do not know the correspondences between feature points. Moreover, there are again significant structural differences between the edge-sets of the graphs.

4.1 Synthetic Images

We commence our experimental study using the synthetic image sequence, where we have access to the correspondences between nodes. In Figure 5(row 1), we show the result of projecting the unweighted adjacency graph for the points onto the parametric eigenspace. The trajectory is well behaved and does not exhibit kinks, or fold back on itself. In addition, the points corresponding to neighbouring views are always closer to one-another than views that are not adjacent. This feature is underlined by the interpoint distance function which is shown in the right panel of the figure.

Figure 5(row 2) repeats this analysis for the weighted adjacency matrix. Here, with the exception of the first and last views, the trajectory is almost linear. This may prove an advantage for view indexing since the trajectory can be interpolated in a linear fashion. From the right panel of the figure its is also clear that the interpoint distance function is also smoother than in the case of the unweighted adjacency matrix.

In Figure 5(row 3) we show the results obtained with the point-proximity matrix. This provides by far the most uniform spacing of points with view number. The trajectory is also considerably smoother than in the previous two cases. The interpoint distance function is also very continuous.

Next we turn our attention to the weight histogram representation of the points. In Figure 5(row 4), we show the results with the weight histogram for the edges of Delaunay graph. Here the trajectory appears well-ordered, but it is somewhat erratic and is not smooth. This feature is supported by the interpoint distance function, which exhibits local maxima and minima. A slightly better picture emerges when the point proximity histogram is used. In Figure 5(row 5), the trajectory is slightly less erratic and the distance function contains less local structure.

Finally, we turn our attention to multidimensional scaling. Figure 6(row 1) shows the result obtained when multidimensional scaling is applied to the graph-distances. Here the trajectory is well-ordered and relatively smooth. In addition, the interpoint distance function is also quite smooth. Figures 6(row 2) and 6(row 3) repeat the MDS analysis for the adjacency weight histogram and the proximity weight histogram. The results are comparable to those obtained with the matrix-based distance measure, and do not require correspondence information.

4.2 Real World Sequence

Since the numbers of corners in the real world images vary, and, in addition, we do not have correspondence data for the detected points, we can only use the histogram-based method and multidimensional scaling.

Figures 7(row 1) and 7(row 2) show the trajectories and distance plots for the edge weight and point proximity weight histograms. From Figure 7(row 1) it is clear that the edge weight histogram is not suitable for view-based object recognition. The trajectory is erratic and folds over on itself several times. Moreover, the associated inter-point distance plot is very noisy. In the case of the proximity weight histogram, the trajectory is still quite noisy, but does not fold-over upon itself.

Finally, we show the results obtained by applying multidimensional scaling. Figures 7(row 3), 7(row 4) and 7(row 5) show the results obtained with the matrix-based graph distance, the weight histogram distances and the proximity weight histogram distances. to the graph-distances. Here the best trajectory is obtained when the matrix-based graph distance is used. This is considerably less erratic than those obtained with the two weight histograms.

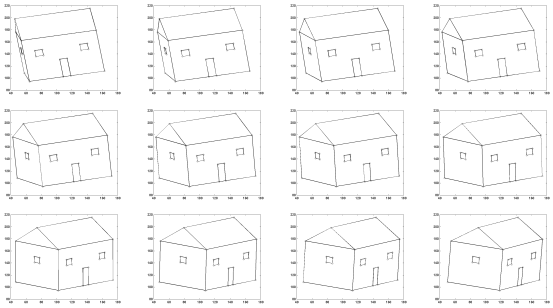

Figure 1. Model images with feature points

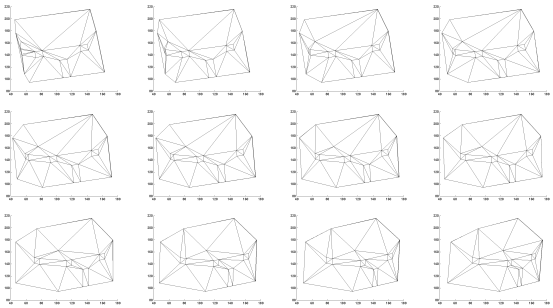

Figure 2. Graph representation of the model images

5 Conclusions

In this paper, we have investigated various ways for constructing view manifolds from graph structures extracted

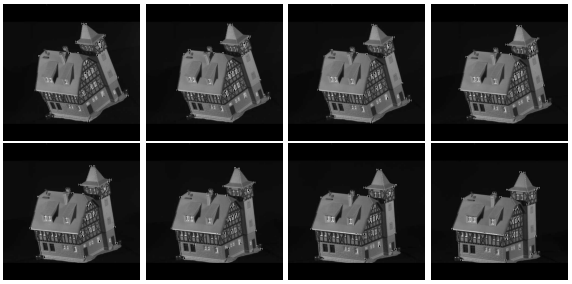

Figure 3. House images with feature points

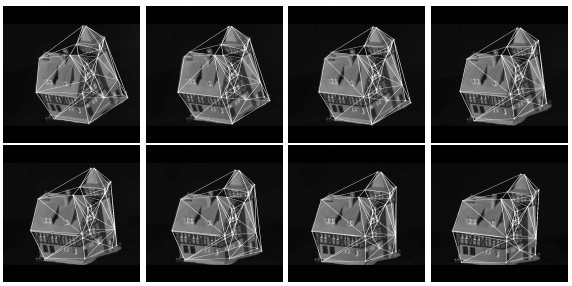

Figure 4. Graph representation of the house images

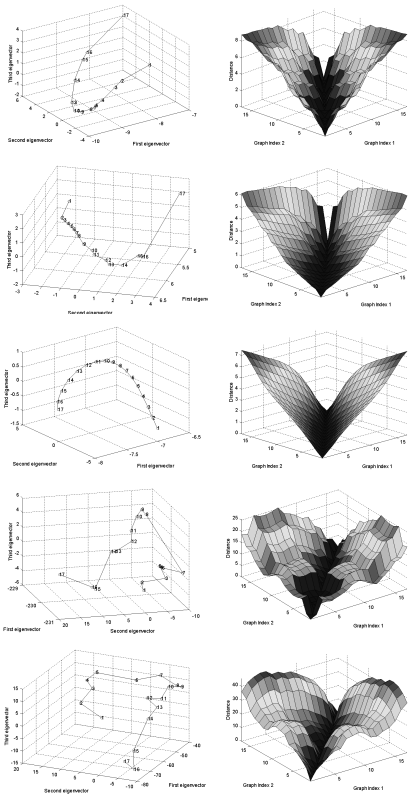

Figure 5. Eigenspace analysis of unweighted adjacency matrix, weighted adjacency matrix,point proximity matrix,weights histogram and proximity histogram. Left:Parametric 3D eigenspace curve by projecting the model graph set. Right:Distance between each pair of graphs.

from images collected when an object is in different poses. We have explored two different approaches to the problem. The first of these is an application of the parametric eigenspace method of Murase and Nayar. This method requires correspondence information and constructs the eigenspace by measuring the structural correlation of the graphs for the different views. The second method uses multidimensional scaling to embed the graphs in a Euclidean space using a matrix of pairwise similarities for the graphs.

Although the first method gives very promising results using both weighted and unweighted adjacency matrix representations, it requires exact correspondences to be provided and does not accommodate graphs of different size. The second method, although it does not return good trajectories, is more flexible in the sense that it can accommodate graphs of different size.

References

[1] P.N. Belhumeur and D.J. Kriegman. What is the set of images of an object under all possible illumination conditions. *IJCV*, 28(3):245–260, July 1998.

[2] I. Biederman. Recognition by components: A theory of human image understanding. *PsychR*, 94(2):115–147, 1987.

[3] I. Biederman. Geon based object recognition. In *BMVC93*, page xx, 1993.

[4] H. Bunke and K. Shearer. A graph distance metric based on the maximal common subgraph. *Pattern Recognition Letters*, 19:255–259, 1998.

[5] Chatfield C. and Collins A.J. *Introduction to multivariate analysis*. Chapman & Hall, 1980.

[6] W.J. Christmas, J. Kittler, and M. Petrou. Structural matching in computer vision using probabilistic relaxation. *IEEE PAMI*, 17(8):749–764, 1995.

[7] M.S. Costa and L.G. Shapiro. 3d object recognition and pose with relational indexing. *CVIU*, 79(3):364–407, September 2000.

[8] M.A. Eshera and K.S. Fu. An image understanding system using attributed symbolic representation and inexact graph-matching. *Journal of the Association for Computing Machinery*, 8(5):604–618, 1986.

[9] Z. Gigus and J. Malik. Computing the aspect graph for line drawings of polyhedral objects. *IEEE TPAMI*, 12(2):113–122, 1990.

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

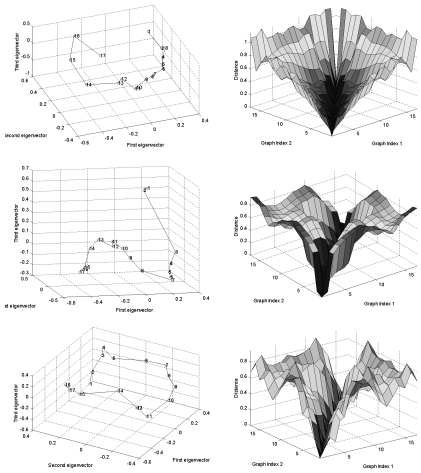

Figure 6. MDS analysis of graph distances, weight histogram distances and proximity histogram distances.

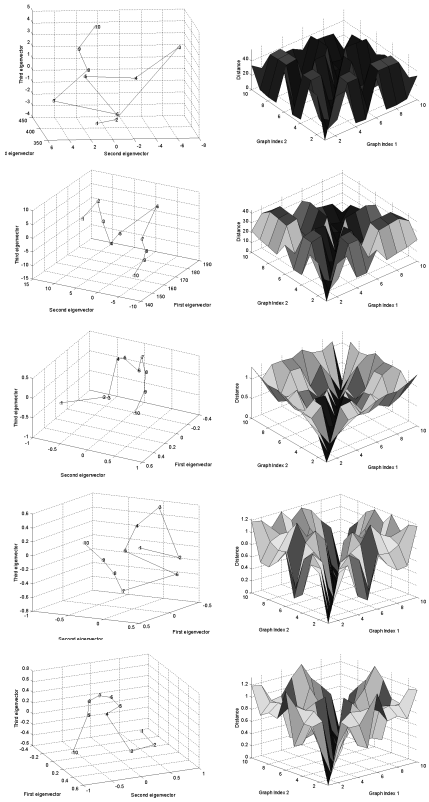

Figure 7. Eigenspace analysis of weight histogram, proximity histogram, and MDS analysis of graph distances, weight histogram distances and proximity histogram distances for house images.

[11] Murase H. and Nayar S.K. Illumination planning for object recognition using parametric eigenspaces. *IEEE TPAMI*, 16(12):1219–1227, 1994.

[12] Kruskal J.B. Nonmetric multidimensional scaling: A numerical method. *Psychometrika*, 29:115–129, 1964.

[13] Gower J.C. Some distance properties of latent root and vector methods used in multivariate analysis. *Biometrika*, 53:325–328, 1966.

[14] J. Jia and K. Abe. Recognizing 3d objects by using models learned automatically from 2d training images. *IEEE TPAMI*, 14(3):315–338, 2000.

[15] D.J. Kriegman and J. Ponce. Computing exact aspect graphs of curved objects: Solids of revolution. *IJCV*, 5(2):119–135, November 1990.

[16] A. Leonardis and H. Bischof. Robust recognition using eigenimages. *CVIU*, 78(1):99–118, April 2000.

[17] B. Luo and E.R. Hancock. Symbolic graph matching using the EM algorithm and Singular Value Decomposition. In *Proceedings of 15th ICPR*, volume 2, pages 141–144, 2000.

[18] B. Luo and E.R. Hancock. Corner detection via topographic analysis of vector potential. *PRL*, 20:635–650,1999.

[19] R. Malik and T. Whangbo. Angle densities and recognition of 3d objects. *IEEE TPAMI*, 19(1):52–57, 1997.

[20] A.M. Martinez and A.C. Kak. PCA versus LDA. *PAMI*, 23(2):228–233, February 2001.

[21] A. Mohan, C. Papageorgiou, and T. Poggio. Example-based object detection in images by components. *PAMI*, 23(4):349–361, April 2001.

[22] H. Murase and S.K. Nayar. Visual learning and recognition of 3-d objects from appearance. *IJCV*, 14(1):5–24, 1995.

[23] R. Myers, R.C. Wilson, and E.R. Hancock. Bayesian graph edit distance. *IEEE TPAMI*.

[24] S.K. Nayar, S.A. Nene, and H. Murase. Subspace methods for robot vision. *IEEE T-RA*, 12(5):750–758, October 1996.

[25] S. Petitjean, J. Ponce, and D.J. Kriegman. Computing exact aspect graphs of curved objects: Algebraic surfaces. *IJCV*, 9(3):231–255, 1992.

[26] A. Sanfeliu and K.S. Fu. A distance measure between attributed relational graphs for pattern recognition. *IEEE Trans. Systems, Man and Cybernetics*, 13(3):353–362, May 1983.

[27] M. Seibert and A.M. Waxman. Adaptive 3-d object recognition from multiple views. *IEEE TPAMI*, 14(2):107–124, 1992.

[28] P.D. Simic. Constrained nets for graph matching and other quadratic assignment problems. *Neural Computation*, 3:268–281, 1991.

[29] R. Wang and H. Freeman. Object recognition based on characteristic view classes. *Proc. ICPR*, I:8–12, 1990.

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

[31] Torgerson W.S. Multidimensional scaling. i. theory and method. *Psychometrika*, 17:401–419, 1952.