String Edit Distance, Random Walks and Graph Matching

Antonio Robles-Kelly * and Edwin R. Hancock

Department of Computer Science, University of York
York, Y01 5DD, UK
{arobkell,erh}@cs.york.ac.uk

Abstract. This paper shows how the eigenstructure of the adjacency matrix can be used for the purposes of robust graph-matching. We commence from the observation that the leading eigenvector of a transition probability matrix is the steady state of the associated Markov chain. When the transition matrix is the normalised adjacency matrix of a graph, then the leading eigenvector gives the sequence of nodes of the steady state random walk on the graph. We use this property to convert the nodes in a graph into a string where the node-order is given by the sequence of nodes visited in the random walk. We match graphs represented in this way, by finding the sequence of string edit operations which minimise edit distance.

1 Introduction

Graph-matching is a task of pivotal importance in high-level vision since it provides a means by which abstract pictorial descriptions can be matched to one-another. Unfortunately, since the process of eliciting graph structures from raw image data is a task of some fragility due to noise and the limited effectiveness of the available segmentation algorithms, graph-matching is invariably approached by inexact means [15,13]. The search for a robust means of inexact graph-matching has been the focus of sustained activity over the last two decades. Early work drew heavily on ideas from structural pattern recognition and revolved around extending the concept of string edit distance to graphs [13,6,4]. More recent progress has centred around the use of powerful optimisation and probabilistic methods, with the aim of rendering the graph matching process robust to structural error.

Despite proving effective, these methods lack the elegance of the matrix representation first used by Ullman in his work on subgraph isomorphism [17]. The task of posing the inexact graph matching problem in a matrix setting has proved to be an elusive one. This is disappointing since a rich set of potential tools are available from the field of mathematics referred to as spectral graph theory. This is the term given to a family of techniques that aim to characterise the global structural properties of graphs using the eigenvalues and eigenvectors

* Supported by CONACYT, under grant No. 146475/151752.

T. Caelli et al. (Eds.): SSPR&SPR 2002, LNCS 2396, pp. 104–112, 2002.

of the adjacency matrix [5]. In the computer vision literature there have been a number of attempts to use spectral properties for graph-matching, object recognition and image segmentation. Umeyama has an eigendecomposition method that matches graphs of the same size [18]. Borrowing ideas from structural chemistry, Scott and Longuet-Higgins were among the first to use spectral methods for correspondence analysis [14]. They showed how to recover correspondences via singular value decomposition on the point association matrix between different images. In keeping more closely with the spirit of spectral graph theory, yet seemingly unaware of the related literature, Shapiro and Brady [16] developed an extension of the Scott and Longuet-Higgins method, in which point sets are matched by comparing the eigenvectors of the point proximity matrix. Here the proximity matrix is constructed by computing the Gaussian weighted distance between points. The eigen-vectors of the proximity matrices can be viewed as the basis vectors of an orthogonal transformation on the original point identities. In other words, the components of the eigenvectors represent mixing angles for the transformed points. Matching between different point-sets is effected by comparing the pattern of eigenvectors in different images. Shapiro and Brady's method can be viewed as operating in the attribute domain rather than the structural domain. Horaud and Sossa[8] have adopted a purely structural approach to the recognition of line-drawings. Their representation is based on the immanental polynomials for the Laplacian matrix of the line-connectivity graph. By comparing the coefficients of the polynomials, they are able to index into a large data-base of line-drawings. Shokoufandeh, Dickinson and Siddiqi [2] have shown how graphs can be encoded using local topological spectra for shape recognition from large data-bases.

In a recent paper Luo and Hancock [11] have returned to the method of Umeyama and have shown how it can be rendered robust to differences and graph-size and structural errors. Commencing from a Bernoulli distribution for the correspondence errors, they develop an expectation maximisation algorithm for graph-matching. Correspondences are recovered in the M or maximisation step of the algorithm by performing singular value decomposition on the weighted product of the adjacency matrices for the graphs being matched. The correspondence weight matrix is updates in the E or expectation step. However, since it is iterative the method is relatively slow and is sensitive to initialisation.

The aim in this paper is to investigate whether the eigenstructure of the adjacency matrix can be used to match graphs using a search method rather than by iteration. To do this we draw on the theory of Markov chains. We consider a Markov chain whose transition probability matrix is the normalised edge-weight matrix for a graph. The steady random walk for the Markov chain on the graph is given by the leading eigenvector of the transition probability, i.e. edge weight, matrix. Hence, by considering the order of the nodes defined by the leading eigenvector, we are able to convert the graph into a string. This opens up the possibility of performing graph matching by performing string alignment by minimising the Levenshtein or edit distance [10,20]. We can follow Wagner and use dynamic programming to evaluate the edit distance between strings

and hence recover correspondences [20]. It is worth stressing that although there been attempts to extend the string edit idea to trees and graphs [21,12,13,15], there is considerable current effort aimed at putting the underlying methodology on a rigourous footing. For instance, Bunke and his co-workers who have demonstrated the relationship between graph edit distance and the size of the maximum common subgraph.

2 Random Walks on Graphs

The relationship between the leading eigenvector of the adjacency matrix and the steady state random walk has been exploited in a number areas including routing theory and information retrieval. We are interested in the weighted graph $G = (V, E, P)$ with node index-set V and edge-set $E \subseteq V \times V$. The off-diagonal elements of the transition probability matrix P are the weights associated with the edges. In this paper, we exploit a graph-spectral property of the transition matrix P to develop a surface height recovery method. This requires that we have the eigenvalues and eigenvectors of the matrix P to hand. To find the eigenvectors of the transition probability matrix, P, we first solve polynomial equation

$$|P - \lambda I| = 0 \tag{1}$$

The unit eigenvector ϕ_i associated with the eigenvalue λ_i is found by solving the system of linear equations

$$P\phi_i = \lambda_i \phi_i \tag{2}$$

and satisfies the condition $\phi_i^T \phi = 1$.

Consider a random walk on the graph G. The walk commences at the node j_1 and proceeds via the sequence of edge-connected nodes $\Gamma = \{j_1, j_2, j_3, ...\}$ where $(j_i, j_{i-1}) \in E$. Suppose that the transition probability associated with the move between the nodes j_l and j_m is $P_{l,m}$. If the random walk can be represented by a Markov chain, then the probability of visiting the nodes in the sequence above is $P_\Gamma = P(j_1) \prod_{l=1}^{|V|} P_{j_{l+1},j_l}$. This Markov chain can be represented using the transition probability matrix P whose element with row l and column m is $P_{l,m}$. Further, let $Q_t(i)$ be the probability of visiting the node indexed i after t-steps of the random walk and let $Q_t = (Q_t(1), Q_t(2), ...)^T$ be the vector of probabilities. After t time steps $Q_t = (P^T)^t Q_0$. If λ_i are the eigenvalues of P and ϕ_i are the corresponding eigenvectors of unit length, then

$$P = \sum_{i=1}^{|V|} \lambda_i \phi_i \phi_i^T$$

As a result, after t applications of the Markov transition probability matrix

$$P^t = \sum_{i=1}^{|V|} \lambda_i^t \phi_i \phi_i^T$$

If the row and columns of the matrix P sum to unity, then $\lambda_1 = 1$. Furthermore, from spectral graph theory [5] provided that the graph G is not a bipartite graph, then the smallest eigenvalue $\lambda_{|V|} > -1$. As a result, when the Markov chain approaches its steady state, i.e. $t \rightarrow \infty$, then all but the first term in the above series become negligible. Hence,

$$\lim_{t\rightarrow\infty} P^t = \phi_1\phi_1^T$$

This establishes that the leading eigenvector of the transition probability matrix is the steady state of the Markov chain. For a more complete proof of this result see the book by Varga [19] or the review of Lovasz [9]. As a result, if we visit the nodes of the graph in the order defined by the magnitudes of the co-efficients of the leading eigenvector of the transition probability matrix, then the path is the steady state Markov chain. In this paper we aim to exploit this property to impose a string ordering on the nodes of a graph, and to use this string ordering property for matching the nodes in different graphs by minimising string edit distance.

Our goal is to match the nodes in a "data" graph $G_D = (V_D, E_D, P_D)$ to their counterparts in a "model" graph $G_M = (V_M, E_M, P_M)$. Suppose that the leading eigenvector for the data-graph transition probability matrix P_D matrix is denoted by $\phi_D^* = (\phi_D^*(1), ..., \phi_D^*(|V_D|))^T$ while that for the model graph transition probability matrix P_M is denoted by $\phi_M^* = (\phi_M^*(1), ...\phi_M^*(|V_M|))^T$. The associated eigenvalues are λ_D^* and λ_M^*. The designation of the two graphs as "data" and "model" is a matter of convention. Here we take the data graph to be the graph which possesses the largest leading eigenvalue, i.e. $\lambda_D^* > \lambda_M^*$.

Our aim is to use the sequence of nodes defined by the rank order of the magnitudes of the components of the leading eigenvector as a means of locating correspondences. The rank order of the nodes in the data graph is given by the string of sorted node-indices $X = (j_1, j_2, j_3, ..., j_{|V_D|})$ where $\phi_D^*(j_1) > \phi_D^*(j_2) > \phi_D^*(j_3) > ... > \phi_D^*(j_{|V_D|})$. The subscript n of the node-index $j_n \in V_D$ is hence the rank-order of the eigenvector component $\phi_D^*(j_n)$. The rank-ordered list of model-graph nodes is $Y = (k_1, k_2, k_3, ..., k_{|V_M|})$ where $\phi_M^*(k_1) > \phi_M^*(k_2) > \phi_M^*(k_3) > ... > \phi_M^*(k_{|V_M|})$.

We augment, the information provided by the leading eigenvectors, with morphological information conveyed by the degree of the nodes in the two graphs. Suppose that $\deg(i)$ is the degree of node i. We establish the morphological affinity $\beta_{i,j}$ of nodes $i \in V_D$ and $j \in V_M$ using their degree ratio. Specifically, the morphological affinity of the nodes is taken to be

$$\beta_{i,j} = \exp\left(-\frac{\max(\deg(i), \deg(j)) - \min(\deg(i), \deg(j))}{\max(\deg(i), \deg(j))}\right) \tag{3}$$

If the degree ratio is one then the affinity measure is maximum. If the ratio is small (i.e. $\beta_{i,j} << 1$) then the affinity is zero.

3 Edit Distance

By taking the leading eigenvectors of the model-graph and data-graph adjacency matrices, we have converted the two graphs into strings. Our aim in this paper is to explore whether we can use string edit distance to robustly match the graphs when they are represented in this way. Let X and Y be two strings of symbols drawn from an alphabet Σ. We wish to convert X to Y via an ordered sequence of operations such that the cost associated with the sequence is minimal. The original string to string correction algorithm defined *elementary edit operations*, $(a, b) \neq (\epsilon, \epsilon)$ where a and b are symbols from the two strings or the NULL symbol, ϵ. Thus, changing symbol x to y is denoted (x, y), inserting y is denoted (ϵ, y), and deleting x is denoted (x, ϵ). A sequence of such operations which transforms X into Y is known as an *edit transformation* and denoted $\Delta =< \delta_1, ..., \delta_{|\Delta|} >$. Elementary costs are assigned by an elementary weighting function $\gamma : \Sigma \cup \{\epsilon\} \times \Sigma \cup \{\epsilon\} \mapsto \Re$; the cost of an edit transformation, $W(\Delta)$, is the sum of its elementary costs. The edit distance between X and Y is defined as

$$\mathbf{d}(X, Y) = \min\{W(\Delta) | \Delta \text{ transforms } X \text{ to } Y\} \tag{4}$$

We aim to to locate correspondence matches by seeking the edit-path that minimises the edit distance between the strings representing the steady state random walks on the two graphs. To this end, suppose that $\delta_l = (a, b)$ and $\delta_{l+1} = (c, d)$ represent adjacent states in the edit path between the stready state random walks X and Y. The cost of the edit path is given by

$$W(\Delta) = \sum_{\delta_l \in \Delta} \gamma_{\delta_l \rightarrow \delta_{l+1}} \tag{5}$$

where $\gamma_{\delta_l \rightarrow \delta_{l+1}}$ is the cost of the transition between the states $\delta_l = (a, b)$ and $\delta_{l+1} = (c, d)$. Since, we commenced with a probabilistic characterisation of the matching problem using Markov chains, we define the elementary edit cost to be the negative logarithm of the transition probability for the edit operation. Hence,

$$\gamma_{(a,b)\rightarrow(c,d)} = -\ln P((a, b) \rightarrow (c, d)) \tag{6}$$

We adopt a simple model of the transition probability. The probability is a product of the node similarity weight, and the edge probabilities. Hence we write

$$P((a, b) \rightarrow (c, d)) = \beta_{a,b}\beta_{c,d}R_D(a, c)R_M(b, d) \tag{7}$$

where R_D and R_M are matrices of compatibility weights. The elements of the matrices are assigned according to the following distribution rule

$$R_D(a, c) = \begin{cases} P_D & if(a, c) \in E_D \\ P_\epsilon & \text{if } a = \epsilon \text{ or } c = \epsilon \\ 0 & \text{otherwise} \end{cases} \tag{8}$$

where E_D is the edge set of the data-graph, P_D is the associated normalised transition probability matrix and P_ϵ is the probability associated with a match to the null symbol ϵ. The compatibility weight is hence zero if either the symbol pair (a, c) is unconnected by an edge of the data-graph, or the symbol pair (b, d) is unconnected by a model graph edge. As a result, edit operations which violate edge consistency on adjacent nodes in the strings are discouraged.

The optimal set of correspondences between the two sequences of nodes is found by minimising the stringf edit distance. The optimal sequence of correspondence Δ^* satisfies the condition

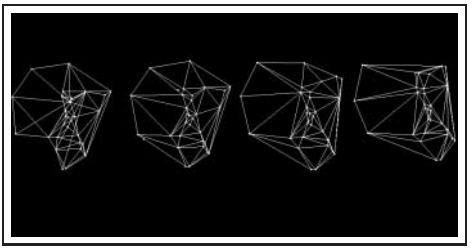
$$\Delta^* = \arg\max_{\Delta} W(\Delta) \tag{9}$$

In practice, we find the optimal edit sequence using Dijkstra's algorithm. Since both the data-graph random walk X and the model-graph random walk are edge-connected, the edit path coils around neighbourhoods in the graphs, while "zippering" the strings together.

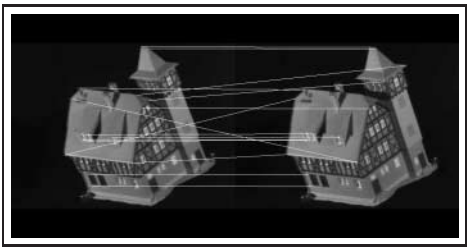
4 Experiments

We have conducted some experiments with the CMU house sequence. This sequence consists of a series of images of a model house which have been captured from different viewpoints. To construct graphs for the purposes of matching, we have first extracted corners from the images using the corner detector of Luo, Cross and Hancock [3]. The graphs used in our experiments are the Delaunay triangulations of these points. The Delaunay triangulations of the example images are shown in Figure 1a. We have matched pairs of graphs representing increasingly different views of the model house. To do this, we have matched the first image in the sequence, with each of the subsequent images. In Figure 1 b, c and d we show the sequence of correspondence matches. In each case the left-hand graph contains 34 nodes, while the right-hand graphs contain 30, 32 and 34 nodes. From the Delaunay graphs it is clear that there are significant structural differences in the graphs. The numbers of correctly matched nodes in the sequence are respectively 29, 24 and 20 nodes. By comparison, the more complicated iterative EM algorithm of Luo and Hancock [11] gives 29, 23 and 11 correct correspondences. As the difference in viewing direction increases, the fraction of correct correspondences decreases from 80% for the closest pair of images to 60% for the most distant pair of images.

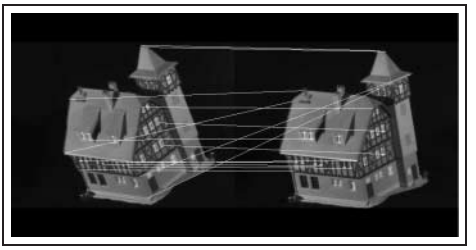
We have conducted some comparison with a number of alternative algorithms. The first of these share with our method the feature of using matrix factorisation to locate correspondences and have been reported by Umeyama [18] and Shapiro and Brady [16]. Since these two algorithms can not operate with graphs of different size, we have taken pairs of graphs with identical numbers of nodes from the CMU sequence; these are the second and fourth images which both contain 32 nodes. Here the Umeyama method and the Shapiro and Brady method both give 6 correct correspondences, while both the Luo and Hancock [11] method and our own give 22 correct correspondences.



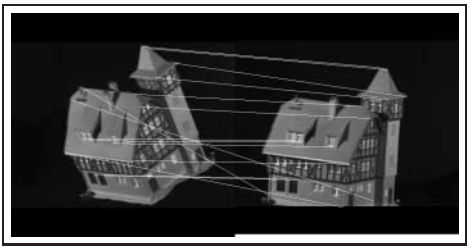
(a)



(b)



(c)



(d)

Fig. 1. Delaunay triangulations and sequence of correspondences

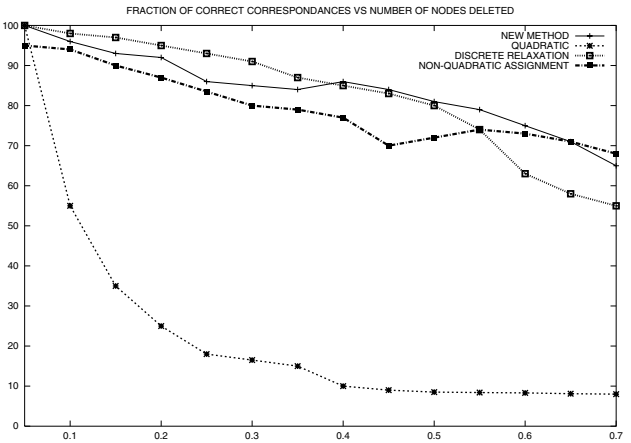


Fig. 2. Sensitivity study results

Finally, we have conducted some experiments with synthetic data to measure the sensitivity of our matching method to structural differences in the graphs and to provide comparison with alternatives. Here we have generated random point-sets and have constructed their Delaunay graphs. We have simulated the effects of structural errors by randomly deleting nodes and re-triangulating the remaining point-set. In Figure 2 we show the fraction of correct correspondences as a function of the fraction of nodes. The performance curve for our new method (marked as "Evidence combining" on the plot) is shown as the lightest of the curves. Also shown on the plot are performance curves for the Wilson and Hancock discrete relaxation scheme, [22], the Gold and Rangarajan [7] quadartic assignment method and the Finch, Wilson and Hancock [1] non-quadratic assignment method. In the case of random node deletion, our method gives performance that is significantly better than the Gold and Rangarajan method, and intermediate in perfomance between the discrete relaxation and non-quadratic assignment methods.

5 Conclusions

The work reported in this paper provides a synthesis of ideas from spectral graph-theory and structural pattern recognition. We use the result from spectral graph theory that the steady state random walk on a graph is given by the leading eigenvector of the adjacency matrix. This allows us to provide a string ordering of the nodes in different graphs. We match the resulting string representations by minimising edit distance. The edit costs needed are computed using a simple probailistic model of the edit transitions which is designed to preserve the edge order on the correspondences.

References

1. R. C. Wilson A. M. Finch and E. R. Hancock. An energy function and continuous edit process for graph matching. *Neural Computation*, 10(7):1873–1894, 1998. 111
2. K. Siddiqi A. Shokoufandeh, S. J. Dickinson and S. W. Zucker. Indexing using a spectral encoding of topological structure. In *Proceedings of the Computer Vision and Pattern Recognition*, 1998. 105
3. Luo Bin and E. R. Hancock. Procrustes alignment with the em algorithm. In *8th International Conference on Computer Analysis of Images and Image Patterns*, pages 623–631, 1999. 109
4. H. Buke. On a relation between graph edit distance and maximum common subgraph. *Pattern Recognition Letters*, 18, 1997. 104
5. Fan R. K. Chung. *Spectral Graph Theory*. American Mathematical Society, 1997. 105, 107
6. M. A. Eshera and K. S. Fu. A graph distance measure for image analysis. *SMC*, 14(3):398–408, May 1984. 104
7. S. Gold and A. Rangarajan. A graduated assignment algorithm for graph matching. *PAMI*, 18(4):377–388, April 1996. 111
8. R. Horaud and H. Sossa. Polyhedral object recognition by indexing. *Pattern Recognition*, 1995. 105
9. L. Lovász. Random walks on graphs: a survey. *Bolyai Society Mathematical Studies*, 2(2):1–46, 1993. 107
10. V. I. Levenshtein. Binary codes capable of correcting deletions, insertions and reversals. *Sov. Phys. Dokl.*, 6:707–710, 1966. 105
11. Bin Luo and E. R. Hancock. Structural graph matching using the EM algorithm and singular value decomposition. *To appear in IEEE Trans. on Pattern Analysis and Machine Intelligence*, 2001. 105, 109
12. B. J. Oommen and K. Zhang. The normalized string editing problem revisited. *PAMI*, 18(6):669–672, June 1996. 106
13. A. Sanfeliu and K. S. Fu. A distance measure between attributed relational graphs for pattern recognition. *IEEE Transactions on Systems, Man and Cybernetics*, 13:353–362, 1983. 104, 106
14. G. Scott and H. Longuet-Higgins. An algorithm for associating the features of two images. In *Proceedings of the Royal Society of London*, number 244 in B, 1991. 105
15. L. G. Shapiro and R. M. Haralick. Relational models for scene analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 4:595–602, 82. 104, 106
16. L. S. Shapiro and J. M. Brady. A modal approach to feature-based correspondence. In *British Machine Vision Conference*, 1991. 105, 109
17. S. Ullman. Filling in the gaps. *Biological Cybernetics*, 25:1–6, 76. 104
18. S. Umeyama. An eigen decomposition approach to weighted graph matching problems. *PAMI*, 10(5):695–703, September 1988. 105, 109
19. R. S. Varga. *Matrix Iterative Analysis*. Springer, second edition, 2000. 107
20. R. A. Wagner. The string-to-string correction problem. *Journal of the ACM*, 21(1), 1974. 105, 106
21. J. T. L. Wang, B. A. Shapiro, D. Shasha, K. Zhang, and K. M. Currey. An algorithm for finding the largest approximatelycommon substructures of two trees. *PAMI*, 20(8):889–895, August 1998. 106
22. R. C. Wilson and E. R. Hancock. Structural matching by discrete relaxation. *PAMI*, 19(6):634–648, June 1997. 111