On the Representation and Matching of Qualitative Shape at Multiple Scales

Ali Shokoufandeh[1⋆], Sven Dickinson[2⋆⋆], Clas Jönsson[3], Lars Bretzner[3], and Tony Lindeberg[3]

[1] Department of Mathematics and Computer Science, Drexel University, USA
[2] Department of Computer Science, University of Toronto, Canada
[3] Computational Vision and Active Perception Laboratory, Department of Numerical Analysis and Computer Science, KTH, Stockholm, Sweden

Abstract. We present a framework for representing and matching multi-scale, qualitative feature hierarchies. The coarse shape of an object is captured by a set of blobs and ridges, representing compact and elongated parts of an object. These parts, in turn, map to nodes in a directed acyclic graph, in which parent/child edges represent feature overlap, sibling edges join nodes with shared parents, and all edges encode geometric relations between the features. Given two feature hierarchies, represented as directed acyclic graphs, we present an algorithm for computing both similarity and node correspondence in the presence of noise and occlusion. Similarity, in turn, is a function of structural similarity, contextual similarity (geometric relations among neighboring nodes), and node contents similarity. Moreover, the weights of these components can be varied on a *node by node basis*, allowing a graph-based model to effectively parameterize the saliency of its constraints. We demonstrate the approach on two domains: gesture recognition and face detection.

1 Introduction

Matching is a fundamental problem in computer vision, and plays an important role in many tasks, ranging from stereo reconstruction to object recognition. Given two images, *matching* can be simply defined as establishing a correspondence between features in one image and similar features in the other image. The representation of image features at multiple scales has long been a powerful paradigm in computer vision, offering a number of attractive properties. From a representational standpoint, scale spaces support the processing of information at appropriate (even variable) levels of resolution. Moreover, the computational complexity of many tasks can be reduced by using the results of processing at coarser scales to constrain processing at finer scales. Matching multi-scale feature hierarchies therefore raises a number of important challenges, including: 1) what

⋆ Dr. Shokoufandeh gratefully acknowledges the support of the US National Science Foundation (NSF-IDM0136337).

⋆⋆ Dr. Dickinson gratefully acknowledges the support of NSERC Canada.

A. Heyden et al. (Eds.): ECCV 2002, LNCS 2352, pp. 759–775, 2002.

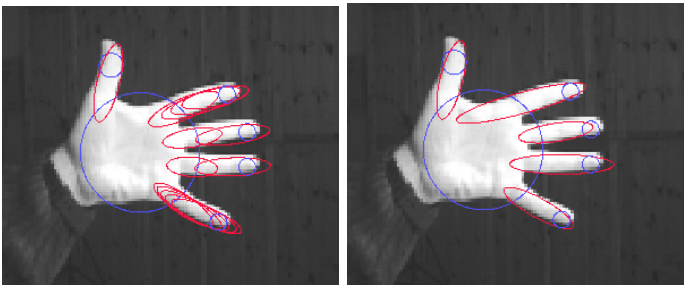
features make up the hierarchy?; and 2) how do we effectively and efficiently match two feature hierarchies in the presence of noise and occlusion?

In this paper, we address the problem of matching two image structures based on qualitative shape. We describe the construction of a qualitative feature hierarchy, based on the detection of ridges and blobs with *automatic scale selection*. These features, and the relations between them, attempt to capture the coarse shape of an object in terms of a set of parts. Given two such feature hierarchies, each represented by a directed acyclic graph (DAG), we describe a matching algorithm that computes both an overall measure of similarity (or distance) as well as node correspondence. The matching algorithm takes into account the structure of the feature hierarchy, the contents of the nodes, and the attributed relations (i.e., context) among nodes. Moreover, the weights of these components can be varied on a *node by node basis*, allowing a graph-based model to effectively parameterize the saliency of its constraints. To demonstrate the approach, we apply it to two separate domains: gesture recognition and face detection.

2 Related Work

There has been considerable effort devoted to both scale space theory and hierarchical structure matching, although much less effort has been devoted to combining the two paradigms. Although space prohibits an extensive review, a few examples are worth noting. Coarse-to-fine image descriptions are plentiful, including work by Burt [2] and Lindeberg [6]. Some have applied such models to directing visual attention, e.g., Tsotsos [19], while others have applied multi-scale structures to matching, including Crowley and Sanderson [3], Rao et al. [12], Bretzner and Lindeberg [1], and Pizer et al. [11].

Although graph matching is a popular topic in the literature, including both inexact and exact graph matching algorithms, there is far less work on dealing with hierarchical graphs, i.e., DAGs, in which lower levels reflect less saliency. The work most relevant to the work reported here is the work of Shokoufandeh et al. [15], which matched multi-scale blob representations represented as directed acyclic graphs. Our framework differs in that: 1) it includes geometric relations among features; 2) the importance of structure and geometry as a matching criteria can be varied continuously within a single framework (as opposed to two separate algorithms); and 3) our matching framework offers several orders of magnitude less complexity. Another related approach is due to Wiskott et al., who apply elastic graph matching to a planar graphs whose nodes represent collections of wavelet jets. Although their features are multi-scale, their representation is not hierarchical, and matching requires that the graphs be coarsely aligned in scale and image rotation [21]. A similar approach was applied to hand posture recognition by Triesch and von der Malsburg [18].



(a) (b)

Fig. 1. Feature Extraction: (a) extracted blobs and ridges at locally adapted scales; (b) extracted features after removing multiple responses and ridge linking.

3 Building a Qualitative Shape Feature Hierarchy

As mentioned in Section 1, our qualitative feature hierarchy represents image structures in terms of blob and ridge features with a set of attribute relations. The representation is an extension of the work presented in [1]. Blob and ridge extraction is performed using automatic scale selection, as described in previous work (see [8] and [7]). It should be noted, however, that the resulting representation is not intended as complete representation. In a more general setting, extensions to other types of image features and feature relations should, of course, be considered.

3.1 Extracting Qualitative Shape Features

A scale-space representation of the image signal f is computed by convolution with Gaussian kernels $g(\cdot;\ t)$ of different variance t, giving $L(\cdot;\ t) = g(\cdot;\ t) * f(\cdot)$. Blob detection aims at locating compact objects or parts in the image. The entity used to detect blobs is the square of the normalized Laplacian operator,

$$\nabla^2_{norm} L = t\,(L_{xx} + L_{yy}). \tag{1}$$

Blobs are detected as local maxima in scale-space. Figure 1(a) shows an image of a hand with the extracted blobs superimposed. A blob is graphically represented by a circle defining a *support region*, whose radius is proportional to the scale.

Elongated structures are localized where the multi-scale ridge detector

$$\mathcal{R}_{norm} L = t^{3/2}\,|L_{pp} - L_{qq}|^2 = t^{3/2}\,((L_{xx} - L_{yy})^2 + 4L^2_{xy}) \tag{2}$$

assumes local maxima in scale-space. Figure 1(a) also shows the extracted ridges, represented as superimposed ellipses, each defining a support region, with width proportional to the scale. To represent the spatial extent of a detected image structure, a windowed second moment matrix

$$\Sigma = \int_{\eta \in \mathbb{R}^2} \begin{pmatrix} L_x^2 & L_x L_y \\ L_x L_y & L_y^2 \end{pmatrix} g(\eta; t_{int})\, d\eta \tag{3}$$

is computed at the detected feature position and at an integration scale t_{int} proportional to the scale of the detected image feature. There are two parameters of the directional statistics that we make use of here: the *orientation* and the *anisotropy*, given from the eigenvalues λ_1 and λ_2 ($\lambda_1 > \lambda_2$) and their corresponding eigenvectors $\boldsymbol{e}_{\lambda_1}$ and $\boldsymbol{e}_{\lambda_2}$ of Σ. The anisotropy is defined as $\tilde{Q} = \frac{1-\lambda_2/\lambda_1}{1+\lambda_2/\lambda_1}$, while the orientation is given by the direction of $\boldsymbol{e}_{\lambda_1}$.

To improve feature detection in scenes with poor intensity contrast between the image object and background, we utilize color information. This is done by extracting features in the R, G and B color bands separately, along with the intensity image. Re-occurring features are awarded with respect to significance. Furthermore, if we have advance information on the color of the object, improvements can be achieved by weighting the significance of the features in the color bands differently.

When constructing a feature hierarchy, we extract the N most salient image features, ranked according to the response of the scale-space descriptors used in the feature extraction process. From these features, a *Feature Map* is built according to the following steps:

Merging multiple feature responses. This step removes multiple overlapping responses originating from the same image structure, the effect of which can be seen in Figure 1(a). To be able to detect overlapping features, a measure of inter-feature similarity is needed. For this purpose, each feature is associated with a 2-D Gaussian kernel $g(\boldsymbol{x}, \Sigma)$, where the covariance is given from the second moment matrix. When two features are positioned near each other, their Gaussian functions will intersect. The similarity measure between two such features is based on the *disjunct volume* D of the two Gaussians [5]. This volume is calculated by integrating the square of the difference between the two Gaussian functions (g_A, g_B) corresponding to the two intersecting features A and B: $D(A,B) = \sqrt{\frac{|\Sigma_A|+|\Sigma_B|}{2}} \int_{\Re^2} (g_A - g_B)^2 d\mathbf{x}$. The disjunct volume depends on the differences in position, variance, scale and orientation of the two Gaussians, and for ridges is more sensitive to translations in the direction perpendicular to the ridge.

Ridge Linking. The ridge detection will produce multiple responses on a ridge structure that is long compared to its width. These ridges are linked together to form one long ridge, as illustrated in Figure 1(b). The criteria for when to link two ridges is based on two conditions: 1) they must be aligned, and 2) their support regions must overlap. After the linking is performed, the anisotropy and support region for the resulting ridge is re-calculated. The anisotropy is re-calculated from the new length/width relationship as 1-(*width of structure)//(length of structure)*.

3.2 Assembling the Features into a Graph

Once the Feature Map is constructed, the component features are assembled into a directed acyclic graph. Associated with each node (blob/ridge) are a number of

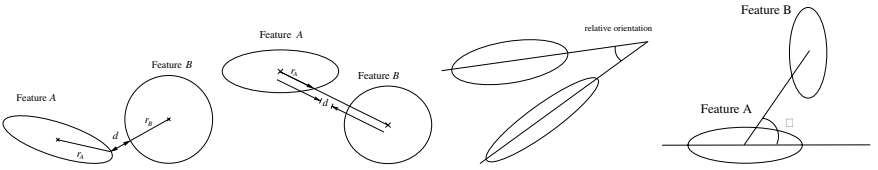


Fig. 2. The four edge relations: two normalized distance measures, relative orientation, and bearing

attributes, including position, orientation, and support region. A feature at the coarsest scale of the Feature Map is chosen as the root. Next, finer-scale features that overlap with the root become its children through hierarchical edges. These children, in turn, select overlapping features at any scale to be their children, etc. From the unassigned features, a new root is chosen and the process is repeated until all features are assigned to the graph. Two nodes that share a parent have a sibling edge between them.[1] In cases where a feature could have two parents, the feature at the coarsest scale is chosen.

Associated with each sibling or hierarchical edge are a number of important geometric attributes. For an edge $\mathcal{E}$, directed from a vertex $\mathcal{V}_A$ representing feature $\mathcal{F}_A$, to a vertex $\mathcal{V}_B$ representing feature $\mathcal{F}_B$, we define the following attributes, as shown in Figure 2:

- **Distance.** Two measures of inter-feature distance are associated with the edge: 1) the smallest distance d from the support region of $\mathcal{F}_A$ to the support region of $\mathcal{F}_B$, normalized to the the largest of the radii r_A and r_B; and 2) the distance between their centers normalized to the radius r_A of $\mathcal{F}_A$ in the direction of the distance vector between their centers.
- **Relative orientation.** The relative orientation between $\mathcal{F}_A$ and $\mathcal{F}_B$.
- **Bearing.** The bearing of a feature $\mathcal{F}_B$, as seen from a feature $\mathcal{F}_A$, is defined as the angle of the distance vector $x_B - x_A$ with respect to the orientation of A measured counter-clockwise.
- **Scale ratio.** The scale invariant relation between $\mathcal{F}_A$ and $\mathcal{F}_B$ is a ratio between scales $t_{\mathcal{F}_A}$ and $t_{\mathcal{F}_B}$.
- **Type.** Classifies the edge as parental or sibling.

4 Matching Problem Formulation

Given two images and their corresponding feature map graphs, $G_1 = (V_1, E_1)$ and $G_2 = (V_1, E_1)$, with $|V_1| = n_1$ and $|V_2| = n_2$, we seek a method for computing their similarity. In the absence of noise, segmentation errors, occlusion, and clutter, computing the similarity of G_1 and G_2 could be formulated as a label-consistent graph isomorphism problem. However, under normal imaging

[1] Sibling edges are not included in a structural description of the graph, thereby ensuring that it is a DAG.

conditions, there may not exist significant subgraphs common to G_1 and G_2. We therefore seek an approximate solution that captures both the topological and geometrical similarity of G_1 and G_2 as well as corresponding node similarity. Topological similarity is a domain-independent measure that accounts for similarity in the "shapes" of two graphs, in terms of numbers of nodes, branching factor distributions, etc. Geometrical similarity accounts for consistency in the relative positions, orientations, and scales of nodes in the two graphs, while node similarity is a domain-dependent function that accounts for the similarity of the contents of corresponding nodes.

In previous work, we mapped the structure of a unique rooted tree to a low-dimensional vector [16]; two vectors that were close together represented two trees that were nearly isomorphic in terms of topology, or structure. In this paper, we extend the approach to cover directed acyclic graphs, a more powerful characterization of hierarchical structure, of which trees are a special case. Moreover, whereas our previous work examined the stability of this mapping under a restricted class of structural perturbations (the perturbed tree was a subtree of the original tree or vice versa), we now extend this analysis to directed acyclic graphs and strengthen it to include any perturbation. Our previous work focused primarily on structural similarity, with node information (restricted to label information) playing a minor role. In this paper, we describe a method for mapping the *geometry* of a directed acyclic graph into a low-dimensional vector; in this case, two vectors that are close together represent two graphs whose nodes are similar in terms of feature type, scale, position, orientation, and support region. Finally, we combine these functions in an extension of our original algorithm (originally intended for rooted trees) that will take two directed acyclic graphs and compute their similarity.

4.1 Encoding Graph Structure

To describe the topology of a DAG, we turn to the domain of eigenspaces of graphs, first noting that any graph can be represented as an antisymmetric $\{0, 1, -1\}$ adjacency matrix, with 1's (-1's) indicating a forward (backward) edge between adjacent nodes in the graph (and 0's on the diagonal). The eigenvalues of a graph's adjacency matrix encode important structural properties of the graph. Furthermore, the eigenvalues of an antisymmetric (or Hermitian) matrix A are invariant to any orthonormal transformation of the form $P^t AP$. Since a permutation matrix is orthonormal, the eigenvalues of a graph are invariant to any consistent re-ordering of the graph's branches. However, before we can exploit a graph's eigenvalues for matching purposes, we must establish their stability under minor topological perturbation, due to noise, occlusion, or deformation.

We will begin by showing that any structural change to a graph can be modeled as a two-step transformation of its original adjacency matrix. The first step transforms the graph's original adjacency matrix to a new matrix having the same spectral properties as the original matrix. The second step adds a noise matrix to this new matrix, representing the structural changes due to noise and/or occlusion. These changes take the form of the addition/deletion of

nodes/arcs to/from the original graph. We will then draw on an important result that relates the distortion of the magnitudes of the eigenvalues of the matrix resulting from the first step to the magnitude of the noise added in the second step. Since the eigenvalues of the original matrix are the same as those of the transformed matrix (first step), the noise-dependent bounds on the magnitudes of the eigenvalues therefore apply to the original matrix. The result will establish the insensitivity of a graph's spectral properties to minor topological changes.

Let's begin with some definitions. Let $A_{_G} \in \{0,1,-1\}^{m\times m}$ denote the adjacency matrix of the graph G on m vertices, and assume H is an n-vertex graph obtained by adding $n-m$ new vertices and a set of edges to the graph G. Let $\Psi : \{0,1,-1\}^{m\times m} \rightarrow \{0,1,-1\}^{n\times n}$, be a *lifting operator* which transforms a subspace of $R^{m\times m}$ to a subspace of $R^{n\times n}$, with $n \geq m$. We will call this operator *spectrum preserving* if the eigenvalues of any matrix $\mathcal{A} \in \{0,1,-1\}^{m\times m}$ and its image with respect to the operator $(\Psi(\mathcal{A}))$ are the same up to degeneracy, i.e., the only difference between the spectra of $\mathcal{A}$ and $\Psi(\mathcal{A})$ is the number of zero eigenvalues ($\Psi(\mathcal{A})$ has $n-m$ more zero eigenvalues then $\mathcal{A}$).

As stated above, our goal is to show that any structural change in graph G can be represented in terms of a spectrum preserving operator and a noise matrix. Specifically, if $A_{_H}$ denotes the adjacency matrix of the graph H, then there exists a spectrum preserving operator $\Psi()$ and a noise matrix $E_H \in \{0,1,-1\}^{n\times n}$ such that:

$$A_{_H} = \Psi(A_{_G}) + E_{_H}. \tag{4}$$

We will define $\Psi()$ as a lifting operator consisting of two steps. First, we will add $n-m$ zero rows and columns to the matrix $A_{_G}$, and denote the resulting matrix by $A'_{_G}$. Next, $A'_{_G}$ will be pre- and post-multiplied by a permutation matrix P and its transpose P^t, respectively, aligning the rows and columns corresponding to the same vertices in $A_{_H}$ and $\Psi(A_{_G})$. Since the only difference between the eigenvalues of $A'_{_G}$ and $A_{_G}$ is the number of zero eigenvalues, and $PA'_{_G}P^t$ has the same set of eigenvalues as the matrix $A'_{_G}$, $\Psi()$ is a spectrum preserving operator. As a result, the noise matrix $E_{_H}$ can be represented as $A_{_H} - \Psi(A_{_G}) \in \{0,1,-1\}^{n\times n}$.

Armed with a spectrum-preserving lifting operator and a noise matrix, we can now proceed to quantify the impact of the noise on the magnitudes of the original graph's eigenvalues. Specifically, let λ_k for $k \in \{1,...,n\}$ denote the magnitude of the k^{th} largest eigenvalue of the matrix A. A seminal result of Wilkinson [20] (see also Stewart and Sun [17]) states that:

Theorem 1. *If A and $A+E$ are $n \times n$ antisymmetric matrices, then:*

$$\lambda_k(A) + \lambda_k(E) \leq \lambda_k(A+E) \leq \lambda_k(A) + \lambda_1(E), \quad \text{for all } k \in \{1,...,n\}. \tag{5}$$

For H (perturbed graph) and G (original graph), the above theorem yields, for all $k \in \{1,...,n\}$:

$$\begin{aligned} &\lambda_k(\Psi(A_G)) + \lambda_k(E_H) \leq \lambda_k(A_H) \leq \lambda_k(\Psi(A_G)) + \lambda_1(E_H) \Longleftrightarrow \\ &\lambda_k(E_H) \leq \lambda_k(A_H) - \lambda_k(\Psi(A_G)) \leq \lambda_1(E_H) \Longleftrightarrow \qquad (6) \\ &|\lambda_k(A_H) - \lambda_k(\Psi(A_G))| \leq \max\{|\lambda_1(E_H)|, |\lambda_k(E_H)|\}, \end{aligned}$$

and since $|\lambda_1(E_H)| = ||E_H||_2$ (spectral radius), we have:

$$|\lambda_k(A_H) - \lambda_k(\Psi(A_G))| \leq ||E_H||_2. \qquad (7)$$

The above chain of inequalities gives a precise bound on the distortion of the magnitudes of the eigenvalues of $\Psi(A_G)$ in terms of the largest eigenvalue of the noise matrix E_H. Since $\Psi()$ is a spectrum preserving operator, the magnitudes of the eigenvalues of A_G follow the same bound in their distortions.

The above result has several important consequences for our application of a graph's eigenvalues to graph matching. Namely, if the perturbation E_H is small in terms of its complexity, then the magnitudes of the eigenvalues of the new graph H (e.g., the query graph) will remain close to their corresponding non-zero eigenvalues of the original graph G (e.g., the model graph), independent of where the perturbation is applied to G. The magnitude of the eigenvalue distortion is a function of the number of vertices added to the graph due to the noise or occlusion. Specifically, if the noise matrix E_H introduces k new vertices to G, then the distortion of every eigenvalue can be bounded by $\sqrt{k-1}$ (Neumaier [10]). This bound can be further tightened if the noise matrix has simple structure. For example, if E_H represents a simple path on k vertices, then its norm can be bounded by $(2\cos\pi/(k+1))$ (Lovász and Pelikán [9]). In short, large distortions are due to the introduction/deletion of large, complex subgraphs to/from G, while small structural changes will have little impact on the higher order eigenvalues G. The magnitudes of the eigenvalues of a graph are therefore stable under minor perturbations in graph structure.

Having established the stability of a DAG's eigenvalues under minor perturbation of the graph, we can now proceed with our task of capturing a graph's structure with a low-dimensional vector. We could, for example, define a vector to be the sorted eigenvalues of a DAG. However, for large DAGs, the dimensionality of the index (and model DAG database) would be prohibitively large. Moreover, since the graph's eigenvalues reflect global structure, they cannot accommodate significant occlusion. Our solution to these problems will be based on: 1) computing eigenvalue sums to reduce dimensionality, and 2) to compute a vector at each node that reflects the node's local structure in the hierarchy.

We will now briefly review our encoding of directed acyclic graph structure, originally used to encode undirected, rooted tree structure [16]. For brevity, subsequent references to eigenvalues imply magnitudes of eigenvalues. Specifically, let D be a DAG whose maximum branching factor is $\Delta(D)$, and let the subgraphs of its root be $D_1, D_2, \ldots, D_S$, as shown in Figure 3(a). For each subgraph, D_i, whose root degree is $\delta(D_i)$, we compute the eigenvalues of D_i's submatrix, sort the eigenvalues in decreasing order by absolute value, and let S_i be the sum

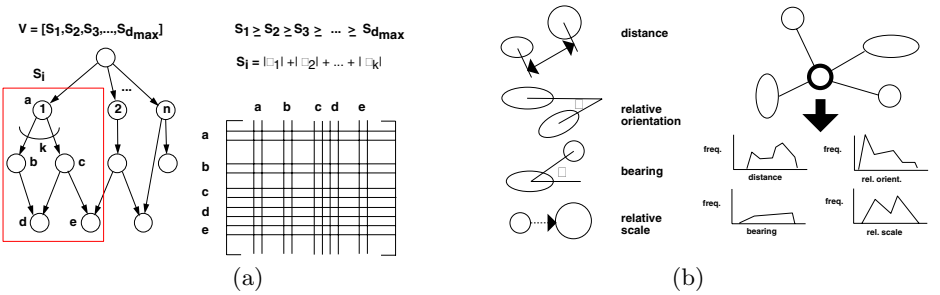


Fig. 3. Forming the structural and geometric signatures.

of the $\delta(D_i) - 1$ largest absolute values. The sorted S_i's become the components of a $\Delta(D)$-dimensional vector assigned to the DAG's root. If the number of S_i's is less than $\Delta(D)$, then the vector is padded with zeroes. We can recursively repeat this procedure, assigning a vector to each nonterminal node in the DAG, computed over the subgraph rooted at that node. We call each such vector a *topological signature vector*, or TSV.

Although the eigenvalue sums are invariant to any consistent re-ordering of the DAG's branches, we have given up some uniqueness (due to the summing operation) in order to reduce dimensionality. We could have elevated only the largest eigenvalue from each subgraph (non-unique but less ambiguous), but this would be less representative of the subgraph's structure. We choose the $\delta(D_i) - 1$ largest eigenvalues for two reasons: 1) the largest eigenvalues are more informative of subgraph structure, and 2) by summing $\delta(D_i) - 1$ elements, we effectively normalize the sum according to the local complexity of the subgraph root. In [16], we describe an efficient method for computing the sum of the k largest eigenvalues directly (without computing the individual eigenvalues).

4.2 Encoding Graph Geometry

The above encoding of structure suffers from the drawback that it does not capture the geometry of the nodes. For example, two graphs with identical structure may differ in terms of the relative positions of their nodes, the relative orientations of their nodes (for elongated nodes), and the relative scales of their nodes. Just as we derived a topological signature vector, which encodes the "neighborhood" structure of a node, we now seek an analogous "geometrical signature vector", which encodes the neighborhood geometry of a node. Below, we derive a geometrical signature of a node's local context. This geometrical signature will be combined with our topological signature in a new algorithm that computes the distance between two directed acyclic graphs.

Let $G = (V, E)$ be a graph to be recognized (input image). For every pair of vertices $u, v \in V$, if there is an edge $\mathcal{E} = (u, v)$ between them, we let $R_{u,v}$ denote the attribute vector associated with edge $\mathcal{E}$. The entries of each such vector represent the set of relations $\mathcal{R} = \{$distance, relative orientation, bearing,

scale ratio} between u and v, as shown in Figure 3(b). For a vertex $u \in V$, we let $N(u)$ denote the set of vertices $v \in V$ such that the directed edge (u, v) corresponds to a sibling relation. For a relation $p \in \mathcal{R}$, we will denote $\mathcal{P}(u, p)$ as the set of values of relation p between vertex u and all the vertices in the set $N(u)$, i.e., $\mathcal{P}(u, p)$ corresponds to the entry p of the vectors $R_{u,v}$ for $v \in N(u)$.[2]

Given two graphs $G = (V, E)$ and $G' = (V', E')$ with vertices $u \in V$ and $u' \in V'$, we compute the similarity between u and u' in terms of their respective values in the sets $\mathcal{P}(u, p)$ and $\mathcal{P}(u', p)$, for all $p \in \mathcal{R}$. Now, let $d_p(u, u')$ denote the Hausdorff distance between sets $\mathcal{P}(u, p)$ and $\mathcal{P}(u', p)$, i.e.

$$d_p(u, u') = \max \left\{ \max_{a \in \mathcal{P}(u,p)} \min_{b \in \mathcal{P}(u',p)} |a - b| \right\}.$$

In other words, the distance between sets $\mathcal{P}(u, p)$ and $\mathcal{P}(u', p)$ is the smallest value d such that every element of $\mathcal{P}(u, p)$ has an element of $\mathcal{P}(u', p)$ within distance d. If the relation is not normalized, we can introduce a translation, t, and compute the distance $w_p(u, u')$ between sets $\mathcal{P}(u, p)$ and $\mathcal{P}(u', p)$ by finding the value of t for which the distance between points of $\mathcal{P}(u, p)$ and $\mathcal{P}(u', p)$ is minimized, i.e.,

$$w_p(u, u') = \min_{t \in R} d_p(u, u') = \min_{t \in R} \max \left\{ \max_{a \in \mathcal{P}(u,p)} \min_{b \in \mathcal{P}(u',p)} |a + t - b| \right\},$$

$w_p(u, u')$ can be computed using the algorithm of Rote [14], in time $O((|N(u)| + |N(u')|) \log(|N(u)| + |N(u')|))$. Given the the values of $w_p(u, u')$ for all $p \in \mathcal{R}$, we arrive at a final node similarity function for vertices u and u':

$$\mathcal{W}(u, u') = e^{-\sum_{p \in \mathcal{R}} w_p(u,u')}.$$

4.3 Matching Algorithm

In previous work, we developed an algorithm for computing the distance (and correspondence) between two unique rooted undirected trees [16]. We now extend our algorithm to compute the distance between two directed acyclic graphs. As mentioned earlier, our major challenge is in computing an approximate isomorphism when, due to noise, occlusion, etc., a nontrivial subgraph isomorphism may not exist. Let's consider relaxing the isomorphism constraint for a moment. Each node in our graph (query or model) is assigned a TSV, which reflects the underlying structure in the subgraph rooted at that node. In addition, each node is assigned a set of relation histograms, which capture the geometry of the graph. If we simply removed all the edges in our two graphs, we would be faced with the problem of finding the best correspondence between the nodes in the query and the nodes in the model; two nodes could be said to be in close correspondence

[2] The exception to this rule is the orientation relation. Rather than use absolute orientation, measured with respect to a reference direction, we instead use the angle from the previous edge in a clockwise ordering edges emanating from a vertex.

if the distance between their TSVs and the distances between their geometric feature histograms was small. In fact, such a formulation amounts to finding the maximum cardinality, maximum weight matching in a bipartite graph spanning the two sets of nodes.

At first glance, such a formulation might seem like a bad idea (by throwing away all that important graph structure!) until one recalls that the graph structure is really encoded in the node's TSV, and the graph geometry is encoded in the node's relation histograms. Is it then possible to reformulate a noisy, largest isomorphic subgraph problem as a simple bipartite matching problem? Unfortunately, in discarding all the graph structure, we have also discarded the underlying hierarchical structure. There is nothing in the bipartite graph matching formulation that ensures that hierarchical constraints among corresponding nodes are obeyed, i.e., that parent/child nodes in one graph don't match child/parent nodes in the other. This reformulation, although softening the overly strict constraints imposed by the largest isomorphic subgraph formulation, is perhaps too weak. We could try to enforce the hierarchical constraints in our bipartite matching formulation, but no polynomial-time solution is known to exist for the resulting formulation. Clearly, we seek an efficient approximation method that will find corresponding nodes between two noisy, occluded DAGs, subject to hierarchical constraints.

Our algorithm, a modification to Reyner's algorithm [13], combines the above bipartite matching formulation with a greedy, best-first search in a recursive procedure to compute the corresponding nodes in two rooted DAGs. As in the above bipartite matching formulation, we compute the maximum cardinality, maximum weight matching in the bipartite graph spanning the two sets of nodes. Edge weight will encode a function of the topological similarity, the geometric similarity, and a domain-dependent node similarity. Moreover, the relative weights of these two components can be specified in the model on a *node by node basis!*, allowing certain geometrical constraints to be relaxed while others strengthened. The result will be a selection of edges yielding a mapping between query and model nodes. As mentioned above, the computed mapping may not obey hierarchical constraints. We therefore greedily choose only the best edge (the two most similar nodes in the two graphs, representing in some sense the two most similar subgraphs), add it to the solution set, and recursively apply the procedure to the subgraphs defined by these two nodes.

Unlike a traditional depth-first search which backtracks to the next statically-determined branch, our algorithm effectively recomputes the branches at each node, always choosing the next branch to descend in a best-first manner. In this way, the search for corresponding nodes is focused in corresponding subgraphs (rooted DAGs) in a top-down manner, thereby ensuring that hierarchical constraints are obeyed. In moving from rooted trees to DAGs, we have to deal with multiple reachability of a vertex in a DAG. Specifically, there may exist two vertices u and v, such that one is not an ancestor of the other, that share a subgraph rooted at vertex w. If the vertices in w are matched, for example, while recursively descending from u, they must not be made available for matching

```
procedure isomorphism(G,H)
    Φ(G, H) ← ∅ ;solution set
    d ← max(δ(G), δ(H)) ;TSV degree
    for u ∈ V_G { ;compute TSV at each node and unmark all nodes in G
        compute χ(u) ∈ R^{d-1} (see Section 4.1)
        unmark u
    }
    for v ∈ V_H { ;compute TSV at each node and unmark all nodes in H
        compute χ(v) ∈ R^{d-1} (see Section 4.1)
        unmark v
    }
    call match(root(G),root(H))
    return(cost(Φ(G, H))
end

procedure match(u,v)
    do
        {
        let G_u ← rooted unmarked subgraph of G at u
        let H_v ← rooted subgraph of H at v
        compute |V_{G_u}| × |V_{H_v}| weight matrix Π(G_u, H_v)
        M ← max cardinality, max weight bipartite matching
                in G(V_{G_u}, V_{H_v}) with weights from Π(G_u, H_v) (see [4])
        (u', v') ← max weight pair in M
        Φ(G, H) ← Φ(G, H) ∪ {(u', v')}
        call match(u',v')
        mark G'_u
        mark G'_v
        }
    while (G_u ≠ ∅ and H_v ≠ ∅)
```

Fig. 4. Algorithm for Matching Two Hierarchical Structures (DAG's) (modified from our previous algorithm for matching undirected rooted trees [16])

while descending v. We therefore need to keep track of matched vertices in our top-down matching process.

As mentioned above, the algorithm computes, at each step, a maximum cardinality, maximum weight matching in the bipartite graph. This requires that we define an objective function that can evaluate the quality of a matching. Given feature map graphs, $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$, representing query and model images, respectively, along with a partial matching function, $\Phi : V_1 \to V_2$, between the regions of G_1 and G_2, we define the mapping matrix, $M(\Phi)$, between G_1 and G_2, to be a $|V_1| \times |V_2|$, $\{0, 1\}$ matrix as follows:

$$M_{u,v} = \begin{cases} 1 \text{ if } u \in V_1,\ v \in V_2,\ u = f(v), \\ 0 \text{ otherwise.} \end{cases}$$

Since $\Phi()$ is a bijective mapping, $M(\Phi)$ will satisfy the following conditions:

$$\begin{array}{l} \sum_{v \in V_2} M_{u,v} \leq 1 \ \forall u \in V_1, \\ \sum_{u \in V_1} M_{u,v} \leq 1 \ \forall v \in V_2. \end{array}$$

Given this formulation of the mapping, $\Phi()$, we define the similarity of G_1 and G_2 to be:

$$\mathcal{S}(\Phi) = \varepsilon \sum_{u \in V_1} \sum_{v \in V_2} M_{u,v} |\Phi(u, v)| + (1 - \varepsilon) \sum_{u \in \mathcal{V}_\in} \gamma(u). \tag{8}$$

The terms of the convex combination $\mathcal{S}(\Phi)$ reflect the similarity between the query and model graphs. Specifically, the first term represents the quality of

correspondence between the matched regions, while the second term penalizes unmatched regions in model graph. By definition, the value of $\gamma(v)$ is one if the the region v in G_2 is matched to a region u in G_1; otherwise, it will have a value strictly less than one if the vertex u is unmatched under $\Phi()$. Specifically, for a vertex v we will define an error function $\mathcal{E}(v)$ which grows proportionally to the complexity of its topological signature, the relative scale of v with respect to its neighbors, and the normalized support size. The penalty for an unmatched region v is defined as:

$$\gamma(v) = e^{-\mathcal{E}(v)}. \tag{9}$$

Finally, the parameter of convexity ε will be computed through a least-squares approximation, using a set of sample training query and model correspondences. Note that the penalizing of unmatched model nodes (regions) is a a domain-dependent assumption for domains in which the target object is unoccluded, i.e., all model regions should be visible. For domains in which target occlusion can be expected, ε can be set to 1.0.

In terms of algorithmic complexity, observe that during the depth-first construction of the matching chains, each vertex in G or H will be matched at most once in the forward procedure. Once a vertex is mapped, it will never participate in another mapping again. The total time complexity of constructing the matching chains is therefore bounded by $O(n^2\sqrt{n \log\log n})$, for $n = \max(n_1, n_2)$ [4]. Moreover, the construction of the $\chi(v)$ vectors will take $O(n\sqrt{n}L)$ time, implying that the overall complexity of the algorithm is $\max(O(n^2\sqrt{n \log\log n}), O(n^2\sqrt{n}L))$. The above algorithm therefore provides, in polynomial time better than $O(n^3)$, an approximate optimal solution to the largest isomorphic subgraph problem in the presence of noise.

It is important to note that since TSV's are computed at each node, thereby providing both global structural descriptions (at coarser scale nodes) and local structural descriptions (at finer scale nodes), the algorithm can exploit locality of representation to accommodate occlusion and clutter. Moreover, the matching algorithm does not require connectedness, allowing multiple roots to spawn disjoint DAG's. However, there is a potential problem when there are, for example, two candidate roots. If there are (finer scale) nodes overlapping the two roots, then the underlying DAG structure will depend on the choice of root, affecting the stability of the description. Under these conditions (and more generally), we allow a node to be a child of both (multiple) roots, thereby providing invariance to root selection order.

5 Demonstration

We demonstrate our representation and matching framework on a gesture recognition problem and a face detection problem. Figure 5(a) shows an example gesture image, along with extracted graph superimposed. Hierarchical edges are shown as red lines, so we see that the ridges corresponding to the fingers are all children of the blob corresponding to the palm. This means that all fingers

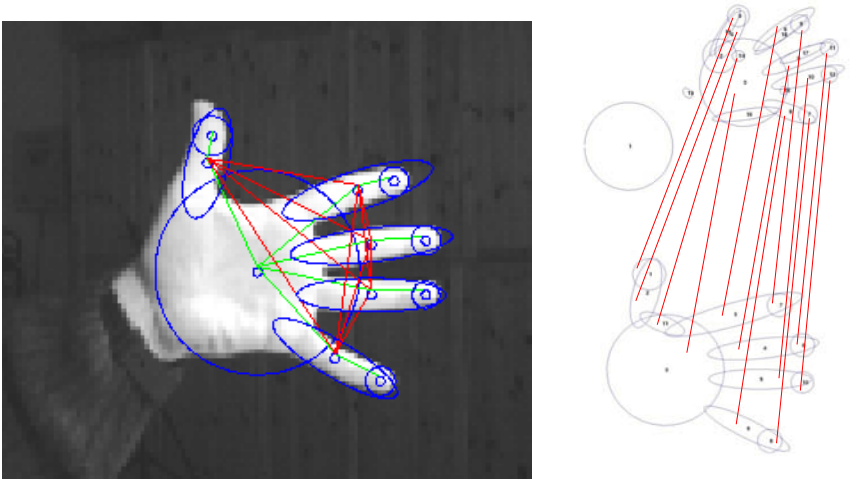


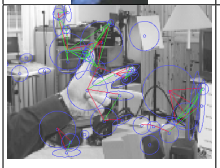
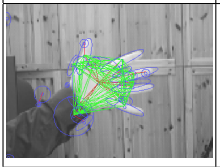
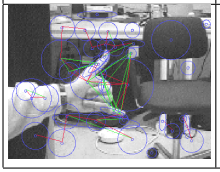
Fig. 5. Gesture Recognition. On the left are detected features as vertices superimposed on the image with the hierarchical relations/edges (red-parent, green-sibling), while on the right is an example computed correspondence between two Feature Maps.

are siblings, connected with green lines. We conducted experiments with query gestures against a uniform background and with query gestures against complex backgrounds.

For each of seven gesture classes, we had five different exemplar queries, varying slightly in shape, finger articulation, scale, etc. That exemplar whose sum distance to all other members of the same class was minimum was designated as the model and removed from the database. All other queries were then compared to each model, yielding 100% recognition over the 28 trials. The first three rows of Table 1 illustrate three of the 28 trials, showing one example query from each of the first three classes, while Figure 5(b) illustrates the computed correspondence between an example pair of gesture Feature Maps. The distances in the matrix reflect similarity, while the boxed entry is that prototype (with maximum similarity) chosen to be the label of the query on the left.

In the next set of experiments, the uniform query background was replaced by a complex background, varying in degree of clutter. Five cluttered queries from each of seven classes were compared to the same model graphs, yielding a 57% (20/35) recognition rate. Of the 15 queries that were most similar to another class, 10 were second closest to the correct class. The next three rows of Table 1 illustrate two correctly matched queries and one incorrectly matched query. Upon examination, it was found that the abundance of blobs and ridges, due to clutter, tended to favour the more complex models whose additional fingers could be accounted for by the additional "accidental" features. We are currently conducting experiments in which the geometric signature weight is strengthened, to see if that filters out extraneous features (at the possible cost of allowing less within-class deformation/articulation).

Table 1. Similarity Between Query and Each Prototypical Model View (most similar model view is boxed)

Query	Similarity to Model						
	12.3519	5.7314	6.0184	7.3514	10.2547	11.1544	9.1162
	4.0512	**6.8804**	5.2172	3.2655	2.8425	4.1812	5.9484
	5.3701	3.1344	**8.3965**	4.4714	7.5609	4.2079	2.7155
	15.1816	9.0173	5.4417	13.1903	10.1817	**15.9527**	13.2277
	**21.8351**	11.0117	12.1684	15.8837	9.2105	17.7449	16.3712
	**10.4261**	3.4102	4.1917	4.0021	7.2581	5.6933	4.9601

In the final experiment, we attempt to find a face in cluttered scenes containing a face. Five cluttered scenes, each containing a face, were matched to a face model. In each of the five trials, the scene object most similar to the model face was the scene face (or some portion thereof). Figure 6 shows two such examples, with the face feature correspondences overlaid. The results illustrate the algorithm's ability to deal with cluttered backgrounds, missing features, and deformed objects.

It is important to note that we are not claiming a superior framework for either gesture recognition or face detection; there are specific solutions to these problems that will yield better results. Rather, we are simply demonstrating that our framework for representing and matching blob decompositions is applicable to a variety of domains (we are currently exploring its application to other domains) and under a variety of imaging conditions.

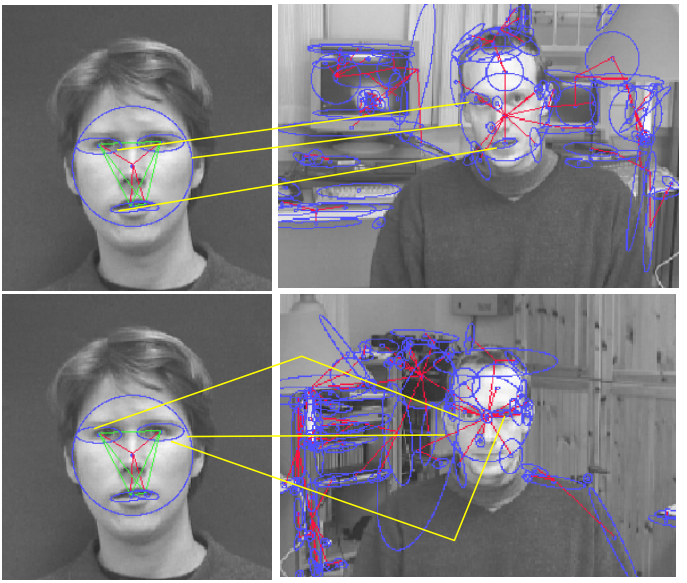


Fig. 6. Face Detection Experiment. On the left is the model face image with extracted blobs, while on the right is the closest match, with corresponding features shown. In the top example, the face, left eye, and mouth regions were detected, while in the bottom example, the face and both eyes were detected. The threshold for blob detection is low to generate more clutter.

6 Conclusions

Matching two images whose similarity exists at a qualitative level and at different scales requires a hierarchical representation in terms of a set of qualitative parts. Blob and ridge features, together with an array of relational attributes, provide a hierarchical characterization of the coarse shape of an object. The matching framework presented attempts to exploit both topological and geometrical properties of multi-scale feature hierarchies. Preliminary testing on two separate domains (without domain-specific tuning) suggests that the approach may offer general applicability. However, much further work and experimentation remains, including the exploration of alternative graph formulations of the blob decomposition, enriching the description of a blob, and comprehensively testing the framework in the presence of occlusion, background clutter, and lighting variation. Finally, we plan to test the framework on other generic recognition domains, including view-based 3-D object recognition.

References

1. L. Bretzner and T. Lindeberg. Qualitative multi-scale feature hierarchies for object tracking. *Journal of Visual Communication and Image Representation*, 11:115–129, 2000.

2. P. J. Burt. Attention Mechanisms for Vision in a Dynamic World. In *Proceedings of the International Conference on Pattern Recognition, Vol.1*, pages 977–987, The Hague, The Netherlands, 1988.
3. J. L. Crowley and A. C. Sanderson. Multiple Resolution Representation and Probabilistic Matching of 2–D Gray–Scale Shape. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 9(1):113–121, January 1987.
4. H. Gabow, M. Goemans, and D. Williamson. An efficient approximate algorithm for survivable network design problems. *Proc. of the Third MPS Conference on Integer Programming and Combinatorial Optimization*, pages 57–74, 1993.
5. I. Laptev and T. Lindeberg. Tracking of multi-state hand models using particle filtering and a hierarchy of multi-scale image features. In *Proc. Scale-Space'01*, Vancouver, Canada, Jul. 2001.
6. T. Lindeberg. Detecting Salient Blob–Like Image Structures and Their Scales With a Scale–Space Primal Sketch—A Method for Focus–of–Attention. *International Journal of Computer Vision*, 11(3):283–318, December 1993.
7. T. Lindeberg. Edge detection and ridge detection with automatic scale selection. *International Journal of Computer Vision*, 30:117–154, 1998.
8. T. Lindeberg. Feature detection with automatic scale selection. *International Journal of Computer Vision*, 30:77–116, 1998.
9. L. Lovász and J. Pelicán. On the eigenvalues of a tree. *Periodica Math. Hung.*, 3:1082–1096, 1970.
10. A. Neumaier. Second largest eigenvalue of a tree. *Linear Algebra and its Applications*, 46:9–25, 1982.
11. S. Pizer, C. Sarang, P. Joshi, P. Fletcher, M. Styner, G. Tracton, and J. Chen. Segmentation of single-figure objects by deformable m-reps. In *Proceedings, MICCAI*, pages 862–871, 2001.
12. R. P. N. Rao, G. J. Zelinsky, M. M. Hayhoe, and D. H. Ballard. Modeling Saccadic Targeting in Visual Search. In D. Touretzky, M. Mozer, and M. Hasselmo, editors, *Advances in Neural Information Processing Systems 8*, pages 830–836. MIT Press, Cambridge, MA, 1996.
13. S. W. Reyner. An analysis of a good algorithm for the subtree problem. *SIAM J. Comput.*, 6:730–732, 1977.
14. G. Rote. Computing the minimum Hausdorff distance between two-point sets on a line under translation. *Information Processing Letters*, 38(3):123–127, 1991.
15. A. Shokoufandeh, I. Marsic, and S. Dickinson. View-based object recognition using saliency maps. *Image and Vision Computing*, 17:445–460, 1999.
16. K. Siddiqi, A. Shokoufandeh, S. Dickinson, and S. Zucker. Shock graphs and shape matching. *International Journal of Computer Vision*, 30:1–24, 1999.
17. G. Stewart and J.-G. Sun. *Matrix Perturbation Theory.* Academic Press, San Diego, 1990.
18. J. Triesch and C. von der Malsburg. Robust classification of hand postures against complex background. In *Proc. Int. Conf. on Face and Gesture Recognition*, pages 170–175, Killington, Vermont, Oct. 1996.
19. J. Tsotsos. An inhibitory beam for attentional selection. In L. Harris and M. Jenkin, editors, *Spatial Vision in Humans and Robots.* Cambridge University Press, 1993.
20. J. Wilkinson. *The Algebraic Eigenvalue Problem.* Clarendon Press, Oxford, England, 1965.
21. L. Wiskott, J.-M. Fellous, N. Krüger, and C. von der Malsburg. Face Recognition by Elastic Bunch Graph Matching. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(7):775–779, July 1997.