Many-to-Many Matching of Scale-Space Feature Hierarchies using Metric Embedding

M. Fatih Demirci[1], Ali Shokoufandeh[1], Yakov Keselman[2], Sven Dickinson[3], and Lars Bretzner[4]

[1] Department of Computer Science, Drexel University,
Philadelphia, PA 19104, USA
{mdemirci,ashokouf}@mcs.drexel.edu
[2] School of Computer Science, Telecommunications and Information Systems,
DePaul University, Chicago, IL 60604, USA
ykeselman@cs.depaul.edu
[3] Department of Computer Science, University of Toronto,
Toronto, Ontario, Canada M5S 3G4
sven@cs.toronto.edu
[4] Computational Vision and Active Perception Laboatory,
Department Of Numerical Analysis and Computer Science,
KTH, Stockholm, Sweden
bretzner@nada.kth.se

Abstract. Scale-space feature hierarchies can be conveniently represented as graphs, in which edges are directed from coarser features to finer features. Consequently, multi-scale feature matching can be formulated as hierarchical graph matching. Most approaches to graph matching assume a one-to-one correspondence between nodes (features) which, due to noise, scale discretization, and feature extraction errors, is overly restrictive. In general, a subset of features in one hierarchy, representing an abstraction of those features, may best match a subset of features in another. We present a framework for the many-to-many matching of multi-scale feature hierarchies, in which features and their relations are captured in a vertex-labeled, edge-weighted graph. The matching algorithm is based on a metric-tree representation of labeled graphs and their low-distortion metric embedding into normed vector spaces. This two-step transformation reduces the many-to-many graph matching problem to that of computing a distribution-based distance measure between two such embeddings. To compute the distance between two sets of embedded, weighted vectors, we use the Earth Mover's Distance under transformation. To demonstrate the approach, we target the domain of multi-scale, qualitative shape description, in which an image is decomposed into a set of blobs and ridges with automatic scale selection.

1 Introduction

The problem of object recognition is often formulated as that of matching configurations of image features to configurations of model features. Such configurations are often represented as vertex-labeled graphs, whose nodes represent

image features (or their abstractions), and whose edges represent relations (or constraints) between the features. For scale-space structures, represented as hierarchical graphs, relations can represent both parent/child relations as well as sibling relations. To match two graph representations (hierarchical or otherwise) means to establish correspondences between their nodes. To evaluate the quality of a match, one defines an overall distance measure, whose value depends on both node and edge similarity.

Due to the importance of the recognition problem (reformulated in terms of graph matching), there has been a growing interest in developing efficient algorithms for matching vertex-labeled graphs. Previous work on graph matching (see Section 2) has typically focused on the problem of finding a one-to-one correspondence between the vertices of two graphs. However, the assumption of one-to-one correspondence is a very restrictive one, for it assumes that the primitive features (nodes) in the two graphs agree in their level of abstraction. In scale-space (hierarchical) structures, this restrictive assumption takes the form of assuming that corresponding features exist at the same level. Unfortunately, there are a variety of conditions that may lead to graphs that represent visually similar image feature configurations yet do not contain a single one-to-one node correspondence. For example, due to noise or segmentation errors, a single feature (node) in one graph may map to a collection of broken features (nodes) in another graph. Or, due to scale differences, a single, coarse-grained feature in one graph may map to a collection of fine-grained features in another graph. In general, we seek not a one-to-one correspondence between image features (nodes), but rather a many-to-many correspondence.

Several existing approaches to the problem of many-to-many graph matching suffer from computational inefficiency and/or from an inability to handle small perturbations in graph structure. This paper seeks a solution to this problem while addressing drawbacks of existing approaches. Drawing on recently developed techniques from the domain of low-distortion graph embedding, we have explored an *efficient* method for mapping a graph's structure to a set of vectors in a low-dimensional space. This mapping not only simplifies the original graph representation, but it retains important information about both local (neighborhood) as well as global graph structure. Moreover, the mapping is *stable* with respect to noise in the graph structure.

The above embedding is applicable only to undirected graphs, in which a metric (undirected) distance can be defined between every pair of nodes. Although scale-space structures may contain undirected edges, information is mostly encoded by hierarchical, non-metric relations, such as parent/child relations. We accommodate these constraints by moving this information into the nodes as feature distributions over the values of incident, oriented edges. Although pulling the oriented edge information into the node would seem to weaken the representation, it's important to note that the resulting node encodes contextual (or neighborhood) information about its relations to adjacent nodes in the graph.

Armed with a low-dimensional, robust vector representation of an input graph's structure, many-to-many graph matching can now be reduced to the

much simpler problem of matching weighted distributions of points in a normed vector space, using a *distribution-based* similarity measure. We consider one such similarity measure, known as the Earth Mover's Distance, and show that the many-to-many vector mapping that realizes the minimum Earth Mover's Distance corresponds to the desired many-to-many matching between nodes of the original graphs. The result is a more efficient and more stable approach to many-to-many graph matching that, in fact, includes the special case of one-to-one graph matching. To illustrate the approach, we apply it to the problem of view-based object recognition, in which an object's views are represented as graphs.

2 Related Work

The problem of object recognition is often formulated as that of matching feature graphs. Several researchers have developed algorithms that find one-to-one correspondences between graph nodes. Shapiro and Haralick [20] proposed a matching algorithm based on comparing weighted primitives (weighted attributes and weighted relation tuples) using a normalized distance for each primitive property that is inexactly matched. Pellilo et al. [16] devised a quadratic programming framework for matching association graphs using a maximal clique reformulation, while Gold and Rangarajan [8] used graduated assignment for matching graphs derived from feature points and image curves. Siddiqi et al. combined a bipartite matching framework with a spectral decomposition of graph structure to match shock graphs [22], while Shokoufandeh et al. [21] extended this framework to directed acyclic graphs that arise in multi-scale image representations. Hancock and his colleagues have also proposed numerous frameworks for graph matching, including [14].

The problem of many-to-many graph matching has also been studied, most often in the context of edit-distance (see, e.g., [13, 19]). In such a setting, one seeks a minimal set of re-labelings, additions, deletions, merges, and splits of nodes and edges that transform one graph into another. However, the edit-distance approach has its drawbacks: 1) it is computationally expensive (polynomial time algorithms are available only for trees); 2) the method in its current form does not accommodate edge weights; and 3) the cost of an editing operation often fails to reflect the underlying visual information (for example, the visual similarity of a contour and its corresponding broken fragments should not be penalized by the high cost of merging the many fragments). In the context of line and segment matching, Beveridge and Riseman [4] addressed this problem via exhaustive local search. Although their method found good matches reliably and efficiently (due to their choice of the objective function and a small neighborhood size), it is unclear how this can be generalized to other types of feature graphs and objective functions.

In a novel generalization of Scott and Longuet [18], Kosinov and Caelli [11] showed how inexact graph matching can be solved using the re-normalization of projections of vertices into the eigenspaces of graphs along with a form of relational clustering. Our framework differs from their approach in that: (1) it

can handle information encoded in a graph's nodes, which is desirable in many vision applications; (2) it does not require an ad hoc clustering step; and (3) it provides a well-bounded, low-distortion metric representation of graph structure. In relation to low-distortion metric representations of graphs, Indyk [10] provides a comprehensive survey of recent advances and applications of low-distortion graph embedding. For recent results related to the properties of low-distortion tree embedding, see [1, 15].

3 Metric Embedding of Graphs

During the last decade, low-distortion embedding has become recognized as a very powerful tool for designing efficient algorithms. In low-distortion embedding of metric spaces into normed spaces, we consider mappings $f : \mathcal{A} \rightarrow \mathcal{B}$, where $\mathcal{A}$ is a set of points in the original metric space, with distance function $\mathcal{D}(.,.)$, $\mathcal{B}$ is a set of points in the (host) d-dimensional normed space $||.||_k$, and for any pair $p, q \in \mathcal{A}$ we have

$$\frac{1}{c}\mathcal{D}(p,q) \leq ||f(p) - f(q)||_k \leq \mathcal{D}(p,q) \tag{1}$$

for a certain parameter c, known as the *distortion*. Intuitively, such an embedding will enable us to reduce problems defined over difficult metric spaces, $(\mathcal{A}, \mathcal{D})$, to problems over easier normed spaces, $(\mathcal{B}, ||.||_k)$. As can be observed from Equation 1, the closer c is to 1, the better the target set $\mathcal{B}$ mimics the original set $\mathcal{A}$. Consequently, the distortion parameter c is a critical characteristic of embedding, f.

Perhaps the most fundamental existence result in computational embedding is due to Bourgain [5]:

Lemma 1. *Any finite metric space* $(\mathcal{A}, \mathcal{D})$ *can be embedded into a finite normed space* $||.||_2$ *of dimension at most* $\log |\mathcal{A}|$ *with distortion* $O(\log |\mathcal{A}|)$.

This result is important since even an exponential[5] matching algorithm in the normed space may be tractable. However, $O(\log |\mathcal{A}|)$ distortion is too high; we seek an embedding with a much lower distortion.

3.1 Low-Distortion Embedding

Our interest in low-distortion embedding is motivated by its ability to transform the problem of many-to-many matching in finite graphs to geometrical problems in low-dimensional vector spaces. Specifically, let $G_1 = (\mathcal{A}_1, E_1, \mathcal{D}_1)$, $G_2 = (\mathcal{A}_2, E_2, \mathcal{D}_2)$ denote two graphs on vertex sets $\mathcal{A}_1$ and $\mathcal{A}_2$, edge sets E_1 and E_2, under distance metrics $\mathcal{D}_1$ and $\mathcal{D}_2$, respectively ($\mathcal{D}_i$ represents the distances between all pairs of nodes in G_i). Ideally, we seek a single embedding that can map each graph to the same vector space, in which the two embeddings can be

[5] in the dimension of the target space.

directly compared. However, in general, this is not possible without introducing unacceptable distortion.

We will therefore tackle the problem in two steps. First, we will seek low-distortion embeddings f_i that map sets $\mathcal{A}_i$ to normed spaces $(\mathcal{B}_i, \|.\|_k)$, $i \in \{1, 2\}$. Next, we will align the normed spaces, so that the embeddings can be directly compared. Using these mappings, the problem of many-to-many vertex matching between G_1 and G_2 is therefore reduced to that of computing a mapping $\mathcal{M}$ between subsets of $\mathcal{B}_1$ and $\mathcal{B}_2$.

In practice, the robustness and efficiency of mapping $\mathcal{M}$ will depend on several parameters, such as the magnitudes of distortion of the $\mathcal{D}_i$'s under the embeddings, f_i's, the computational complexity of applying the embeddings, f_i's, the efficiency of computing the actual correspondences (including alignment) between subsets of $\mathcal{B}_1$ and $\mathcal{B}_2$, and the quality of the computed correspondence. The latter issue will be addressed in Section 5.

The problem of low-distortion embedding has a long history for the case of planar graphs, in general, and trees, in particular. More formally, the most desired embedding is the subject of the following conjecture:

Conjecture 1. [9] Let $G = (\mathcal{A}, E)$ be a planar graph, and let $M = (\mathcal{A}, \mathcal{D})$ be the shortest-path metric for the graph G. Then there is an embedding of M into $\|.\|_p$ with $O(1)$ distortion.

This conjecture has only been proven for the case in which G is a tree. Although the existence of such a distortion-free embedding under $\|.\|_k$-norms was established in [12], no deterministic construction was provided. One such deterministic construction was given by Matoušek [15], suggesting that if we could somehow map our graphs into trees, with small distortion, we could adopt Matoušek's framework.

3.2 Tree Metric of a Distance Function

Before we can proceed with Matoušek's embedding, we must choose a suitable *tree metric* for our graphs. Let $G = (\mathcal{A}, E)$ denote an edge-weighted graph with real edge weights $\mathcal{W}(e)$, $e \in E$. We will say that $\mathcal{D}$ is a metric for G if, for any three vertices $u, v, w \in \mathcal{A}$, $\mathcal{D}(u, v) = \mathcal{D}(v, u) \geq 0$, $\mathcal{D}(u, u) = 0$, and $\mathcal{D}(u, v) \leq \mathcal{D}(u, w) + \mathcal{D}(w, v)$. In general, there are many ways to define metric distances on a weighted graph. The best-known metric is the shortest-path metric $\delta(.,.)$, i.e., $\mathcal{D}(u, v) = \delta(u, v)$, the shortest path distance between u and v for all $u, v \in \mathcal{A}$. We will say that the edge weighted tree $\mathfrak{T} = \mathfrak{T}_G(V', E')$ is a tree metric for G, with respect to distance function $\mathcal{D}$, if for any pair of vertices u, v, the length of the unique path between them in $\mathfrak{T}$ is equal to $\mathcal{D}(u, v)$. The problem of approximating (or fitting) an $n \times n$ distance matrix $\mathcal{D}$ by a tree metric $\mathfrak{T}$ is known as the *Numerical Taxonomy* problem.

The Numerical Taxonomy problem is closely related to that of constructing an *additive metric* distance, i.e., a metric distance D that satisfies the *4-point* condition, $\mathcal{D}[x, y] + \mathcal{D}[z, w] \leq \max\{\mathcal{D}[x, z] + \mathcal{D}[y, w], \mathcal{D}[x, w] + \mathcal{D}[y, z]\}$ $\forall x, y, z, w$. A stronger version of the 4-point condition is the *ultra-metric* condition. A metric

D is an ultra-metric if, for all points x, y, z, $D[x, y] \leq \max\{D[x, z], D[y, z]\}$. Observe that an ultra-metric is a type of tree metric defined on rooted trees, where the distance to the root is the same for all leaves in the tree. This is a critical property in the construction of metric trees for distance functions.

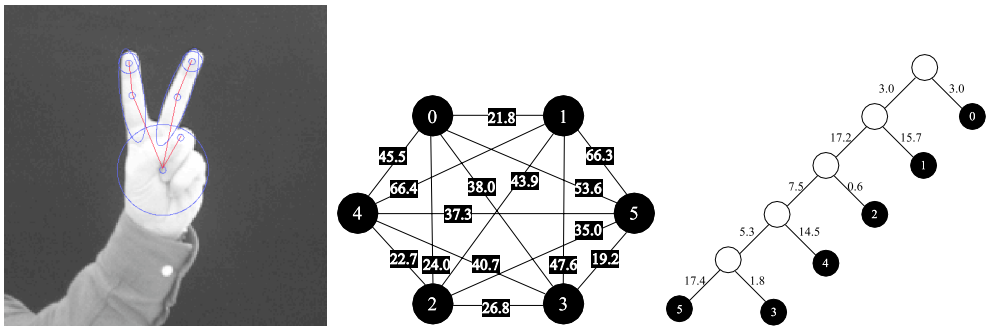
The following Theorem relates the existence of a tree metric to the 4-point condition:

Theorem 1. (see [6]) *A metric $\mathcal{D}$ is additive if and only if it is a tree metric.*

In fact, if there is a tree metric $\mathfrak{T}$ coinciding exactly with $\mathcal{D}$, it is unique and constructible in linear time [24]. If $\mathcal{D}$ is not an additive metric there might be no tree metric $\mathfrak{T}$ that exactly coincides with $\mathcal{D}$. In this case, we can approximate $\mathcal{D}$ under norms, such as $||.||_k$, $k \geq 1$. That is, we want to find a tree metric $\mathfrak{T}$ minimizing $||\mathfrak{T} - \mathcal{D}||_k$.

In the event that G is not a tree, we will use an approximation framework proposed by Agarwala et al. [1]. They consider the approximate numerical taxonomy problem for additive metrics under the $||.||_\infty$ norm. The construction of a tree metric $\mathfrak{T}$ in their algorithm is achieved by transforming the general tree metric problem to that of ultra-metrics. Their algorithm will generate an approximation tree metric $\mathfrak{T}$, to an optimal additive metric under $||.||_\infty$ in time $O(n^2)$ (see [1] for details). It should be noted that this construction does not maintain the vertex set of G invariant, i.e., $V(G) \subseteq V'(\mathfrak{T})$. We will have to make sure that in the embedding process (see Section 3.4), the extra vertices generated during the metric tree construction are eliminated.

An example of the embedding applied to a multi-scale blob decomposition [21] is shown in Figure 1. The gesture image (a) consists of 5 regions (the topmost region is not shown in the image). The complete graph in (b) captures the Euclidean distance between the centroids of the regions, while (c) is the metric tree representation of the multi-scale decomposition (with additional vertices).



(a) (b) (c)

Fig. 1. Metric tree embedding of Euclidean distances for blob and ridge decomposition.

3.3 Path Partition of a Graph

The construction of the embedding depends on the notion of a path partition of a graph. In this subsection, we introduce the path partition, and then use it in the next subsection to construct the embedding. Given a weighted graph $G = (V, E)$ with metric distance $\mathcal{D}(.,.)$, let $\mathfrak{T} = (V', \mathfrak{E})$ denote a metric tree representation of G, whose vertex distances are approximately $\mathcal{D}(.,.)$. In the event that G is a tree, $\mathfrak{T} = G$; otherwise, $\mathfrak{T}$ is the metric tree of G. To construct the embedding, we will assume that $\mathfrak{T}$ is a rooted tree. It will be clear from the construction that the choice of the root does not affect the distortion of the embedding.

The dimensionality of the embedding of $\mathfrak{T}$ depends on the caterpillar dimension, denoted by $\text{cdim}(\mathfrak{T})$, and is recursively defined as follows [15]. If $\mathfrak{T}$ consists of a single vertex, we set $\text{cdim}(\mathfrak{T}) = 0$. For a tree $\mathfrak{T}$ with at least 2 vertices, $\text{cdim}(\mathfrak{T}) \leq k+1$ if there exist paths $P_1, \ldots, P_r$ beginning at the root and otherwise pairwise disjoint, such that each component $\mathfrak{T}_j$ of $\mathfrak{T} - \mathfrak{E}(P_1) - \mathfrak{E}(P_2) - \cdots - \mathfrak{E}(P_r)$ satisfies $\text{cdim}(\mathfrak{T}_j) \leq k$. Here, $\mathfrak{T} - \mathfrak{E}(P_1) - \mathfrak{E}(P_2) - \cdots - \mathfrak{E}(P_r)$ denotes the tree $\mathfrak{T}$ with the edges of the P_i's removed, and the components $\mathfrak{T}_j$ are rooted at the single vertex lying on some P_i. The caterpillar dimension can be determined in linear time for a rooted tree $\mathfrak{T}$, and it is known that $\text{cdim}(\mathfrak{T}) \leq \log(|V'|)$ (see [15]).

The construction of vectors $f(v)$, for $v \in V$, depends on the notion of a *path partition* of $\mathfrak{T}$. The path partition $\mathfrak{P}$ of $\mathfrak{T}$ is empty if $\mathfrak{P}$ is single vertex; otherwise, $\mathfrak{P}$ consists of some paths $P_1, \ldots, P_r$ as in the definition of $\text{cdim}(\mathfrak{T})$, plus the union of path partitions of the components of $\mathfrak{T} - \mathfrak{E}(P_1) - \mathfrak{E}(P_2) - \cdots - \mathfrak{E}(P_r)$. The paths $P_1, \ldots, P_r$ have level 1, and the paths of level $k \geq 2$ are the paths of level $k-1$ in the corresponding path partitions of the components of $\mathfrak{T} - \mathfrak{E}(P_1) - \mathfrak{E}(P_2) - \cdots - \mathfrak{E}(P_r)$. Note that the paths in a path partition are edge-disjoint, and their union covers the edge-set of $\mathfrak{T}$.

To illustrate these concepts, consider the tree shown in Figure 2. The three darkened paths from the root represent the three level 1 paths. Following the removal of the level 1 paths, we are left with 6 connected components that, in turn, induce seven level 2 paths, shown with lightened edges.[6] Following the removal of the seven level 2 paths, we are left with an empty graph. Hence, the caterpillar dimension ($\text{cdim}(\mathfrak{T})$) is 2. It is easy to see that the path partition $\mathfrak{P}$ can be constructed using a modified depth-first search in $O(|V'|)$ time.

3.4 Construction of the Embedding

Given a path partition $\mathfrak{P}$ of $\mathfrak{T}$, we will use m to denote the number of levels in $\mathfrak{P}$, and let $P(v)$ represent the unique path between the root and a vertex $v \in V$. The first segment of $P(v)$ of weight l_1 follows some path P^1 of level 1 in $\mathfrak{P}$, the second segment of weight l_2 follows a path P^2 of level 2, and the last segment of weight l_α follows a path P^α of level $\alpha \leq m$. The sequences $\langle P^1, \ldots, P^\alpha \rangle$ and $\langle l_1, \ldots, l_\alpha \rangle$ will be referred to as the *decomposition sequence* and the *weight sequence* of $P(v)$, respectively.

[6] Note that the third node from the root in the middle level 1 branch is the root of a tree-component consisting of five nodes that will generate two level 2 paths.

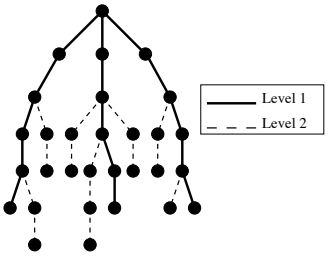


Fig. 2. Path partition of a tree.

To define the embedding $f : V \to \mathcal{B}$ under $||.||_2$, we let the relevant coordinates in $\mathcal{B}$ be indexed by the paths in $\mathfrak{P}$. The vector $f(v)$, $v \in V$, has non-zero coordinates corresponding to the paths in the decomposition sequence of $P(v)$. Returning to Figure 2, the vector $f(v)$ will have 10 components (defined by three level 1 paths and seven level 2 paths). Furthermore, every vector $f(v)$ will have at most two non-zero components. Consider, for example, the lowest leaf node in the middle branch. Its path to the root will traverse three level 2 edges corresponding to the fourth level 2 path, as well as three level 1 edges corresponding to the second level 1 path.

Such embedding functions have become fairly standard in the metric space representation of weighted graphs [15]. In fact, Matoušek [15] has proven that setting the i-th coordinate of $f(v)$, corresponding to path P^k, $1 \leq k \leq \alpha$, in decomposition sequence $\langle P^1, \ldots, P^\alpha \rangle$, to

$$f(v)_i = \sqrt{l_k \left[l_k + \sum_{j=1}^{\alpha} \max\left(0, l_j - l_k/2m\right) \right]}$$

will result in a small distortion of at most $\sqrt{\log \log |V'|}$. It should be mentioned that although the choice of path decomposition $\mathfrak{P}$ is not unique, the resulting embeddings are isomorphic up to a transformation. Computationally, constructions of $\mathfrak{T}$, $\mathfrak{P}$, and $\mathcal{B}$ are all linear in terms of $|V|$ and $|\mathfrak{E}|$. It should be noted that although the tree $\mathfrak{T}$ has been defined based on set V', we have only embedded the vertices of G (set V).

4 Encoding Scale-Space Features

The distance metric defined on the graph structure is based on the undirected edge weights. While the above embedding has preserved the distance metric, it has failed to preserve any oriented relations, such as the hierarchical relations common to scale-space structures. This is due to the fact that oriented relations do not satisfy the symmetry property of a metric. We can retain this important information in our embedding by moving it into the nodes as node attributes, a technique used in the encoding of directed topological structure in [22], directed geometric structure in [21], and shape context in [3]. Encoding in a node the

attributes of the oriented edges incident to the node requires computing distributions on the attributes and assigning them to the node. For example, a node with a single parent at a coarser scale and two children at a finer scale might encode a relative scale distribution (histogram) as a node attribute. The resulting attribute provides a contextual signature for the node which will be used by the matcher (Section 5) to reduce matching ambiguity.

Specifically, let $G = (V, E)$ be a graph to be recognized. For every pair of vertices, we let $R_{u,v}$ denote the attribute vector associated with the pair (u, v). The entries of each such vector represent the set of oriented relations R between u, v. For a vertex $u \in V$, we let $N(u)$ denote the set of vertices $v \in V$ adjacent to u. For a relation $p \in R$, we will denote $\mathcal{P}(u, p)$ as the set of values for relation p between u and all vertices in $N(u)$, i.e., $\mathcal{P}(u, p)$ corresponds to entry p of vector $R_{u,v}$ for $v \in N(u)$. Feature vector $\mathcal{P}_u$ for point u is the set of all $\mathcal{P}(u, p)$'s for $p \in R$. Observe that every entry $\mathcal{P}(u, p)$ of vector $\mathcal{P}_u$ can be considered as a local distribution (*histogram*) of feature p in the neighborhood $N(u)$ of u. We adopt the method of [21], in which the distance function for two such vectors $\mathcal{P}_u$ and $\mathcal{P}_p$ is computed through a weighted combination of Hausdorff distances between $\mathcal{P}(u, p)$ and $\mathcal{P}(u', p)$ for all values of p.

5 Distribution-Based Many-to-Many Matching

By embedding vertex-labeled graphs into normed spaces, we have reduced the problem of many-to-many matching of graphs to that of many-to-many matching of weighted distributions of points in normed spaces. However, before we can match two point distributions, we must map them into the same normed space. This involves reducing the dimension of the higher-dimensional distribution and transforming one of the distributions with respect to the other. Given a pair of weighted distributions in the same normed space, the Earth Mover's Distance (EMD) framework [17] is then applied to find an optimal match between the distributions. The EMD approach computes the minimum amount of work (defined in terms of displacements of the masses associated with points) it takes to transform one distribution into another.

5.1 Embedding Point Distributions in the Same Normed Space

Embeddings produced by the graph embedding algorithm can be of different dimensions and are defined only up to a distance-preserving transformation (a translated and rotated version of a graph embedding will also be a graph embedding). Therefore, in order to apply the EMD framework, we must first perform a "registration" step, whose objective is to project the two distributions into the same normed space. The resulting transformation is expected to minimize the initial EMD between the distributions.

Our transformation is based on Principal Components Analysis (PCA). Namely, the projection of the original vectors onto the subspace spanned by the first K right singular vectors of the covariance matrix retains the maximum information

about the original vectors among all projections onto subspaces of dimension K. Hence, projecting the two distributions onto the first K right singular vectors of their covariance matrices will equalize their dimensions while losing minimal information. Specifically, assuming that K is the minimum of the two dimensions, we define embeddings $P_x(x_i) = W_x^T(x_i - \mu_x)/\sigma_x$ and $P_y(y_i) = W_y^T(y_i - \mu_y)/\sigma_y$ as follows:

$$\begin{array}{ll}
\mu_x & \leftarrow (\sum_i w_i x_i)/\sum_i w_i \\
\mu_y & \leftarrow (\sum_i w_i y_i)/\sum_i w_i \\
\sigma_x^2 & \leftarrow (\sum_i w_i \|x_i - \mu_x\|)/\sum_i w_i \\
\sigma_y^2 & \leftarrow (\sum_i w_i \|y_i - \mu_y\|)/\sum_i w_i \\
\Sigma_{xx} & \leftarrow (\sum_i w_i (x_i - \mu_x)(x_i - \mu_x)^T)/\sum_i w_i \\
\Sigma_{xx} & = U_x D_x V_x^T \text{ is the SVD of } \Sigma_{xx} \\
W_x & \leftarrow \text{first } K \text{ columns of } V_x \\
\Sigma_{yy} & \leftarrow (\sum_i w_i (y_i - \mu_y)(y_i - \mu_y)^T)/\sum_i w_i \\
\Sigma_{yy} & = U_y D_y V_y^T \text{ is the SVD of } \Sigma_{yy} \\
W_y & \leftarrow \text{first } K \text{ columns of } V_y
\end{array}$$

5.2 The Earth Mover's Distance

The Earth Mover's Distance (EMD) [17, 7] is designed to evaluate dissimilarity between two multi-dimensional distributions in some feature space. The EMD approach assumes that a distance measure between single features, called the *ground distance*, is given. The EMD then "lifts" this distance from individual features to full distributions. Moreover, if the weights of the distributions are the same, and the ground distance is a metric, EMD induces a metric distance [17]. However, the main advantage of using EMD lies in the fact that it subsumes many histogram distances and permits partial matches in a natural way. This important property allows the similarity measure to deal with uneven clusters and noisy data sets.

Computing the EMD is based on a solution to the well-known *transportation problem* [2], whose optimal value determines the minimum amount of "work" required to transform one distribution into the other. More formally, let $P = \{(p_1, w_{p_1}), \ldots, (p_m, w_{p_m})\}$ be the first distribution with m points, and let $Q = \{(q_1, w_{q_1}), \ldots, (q_n, w_{q_n})\}$ be the second distribution with n points. Let $D = [d_{ij}]$ be the ground distance matrix, where d_{ij} is the ground distance between points p_i and q_j. Our objective is to find a flow matrix $F = [f_{ij}]$, with f_{ij} being the flow between points p_i and q_j, that minimizes the overall cost:

$$\text{Work}(P, Q, F) = \sum_{i=1}^{m} \sum_{j=1}^{n} f_{ij} d_{ij}$$

subject to the following list of constraints:

$$\begin{array}{l}
f_{ij} \geq 0,\ 1 \leq i \leq m,\ 1 \leq j \leq n \\
\sum_{j=1}^{n} f_{ij} \leq w_{p_i},\ 1 \leq i \leq m \\
\sum_{i=1}^{m} f_{ij} \leq w_{q_j},\ 1 \leq j \leq n \\
\sum_{i=1}^{m} \sum_{j=1}^{n} f_{ij} = \min\left(\sum_{i=1}^{m} w_{p_i}, \sum_{j=1}^{n} w_{q_j}\right)
\end{array}$$

The optimal value of the objective function, $\text{Work}(P, Q, F)$, defines the Earth Mover's Distance between the two distributions.

The above formulation assumes that the two distributions have been aligned. However, recall that a translated and rotated version of a graph embedding will also be a graph embedding. To accommodate pairs of distributions that are "not rigidly embedded", Cohen and Guibas [7] extended the definition of EMD, originally applicable to pairs of fixed sets of points, to allow one of the sets to undergo a transformation. Assuming that a transformation $T \in \mathcal{T}$ is applied to the second distribution, distances d_{ij}^T are defined as $d_{ij}^T = d(p_i, T(q_j))$, and the objective function becomes $\text{Work}(P, Q, F, T) = \sum_{i=1}^{m} \sum_{j=1}^{n} f_{ij} d_{ij}^T$. The minimal value of the objective function defines the Earth Mover's Distance between the two distributions that are allowed to undergo a transformation from $\mathcal{T}$.

Cohen and Guibas [7] also suggested an iterative process (which they call **FT**, short for "an optimal **F**low and an optimal **T**ransformation") that achieves a local minimum of the objective function. Starting with an initial transformation $T^{(0)} \in \mathcal{T}$ from a given $T^{(k)} \in \mathcal{T}$, they compute the optimal flow $F = F^{(k)}$ that minimizes the objective function, $\text{Work}(P, T^{(k)}(Q), F)$, and from a given optimal flow, $F^{(k)}$, they compute an optimal transformation, $T = T^{(k+1)}, \in \mathcal{T}$ that minimizes the objective function, $\text{Work}(P, T(Q), F^{(k)})$. The iterative process stops when the improvement in the objective function value falls below a threshold. The resulting optimal pair (F, T) depends on the initial transformation $T^{(0)}$. Starting the iteration from several initial transformations increases the chances of obtaining a global minimum.

5.3 Choosing an Appropriate Transformation

For our application, the set $\mathcal{T}$ of allowable transformations consists of only those transformations that preserve distances. Therefore, we use a weighted version of the Least Squares Estimation algorithm [23] to compute an optimal distance-preserving transformation given a flow between the distributions. Specifically, given a set of pairings $\{(x_i, y_i, w_i)\}$ (the flow of weight w_i is sent from point x_i to point y_i), we define the transformation $T(x) = cRx + t$ in accordance with [23] as follows:

$$
\begin{aligned}
\mu_x &\leftarrow (\textstyle\sum_i w_i x_i)/\sum_i w_i \\
\mu_y &\leftarrow (\textstyle\sum_i w_i y_i)/\sum_i w_i \\
\sigma_x^2 &\leftarrow (\textstyle\sum_i w_i \|x_i - \mu_x\|)/\sum_i w_i \\
\sigma_y^2 &\leftarrow (\textstyle\sum_i w_i \|y_i - \mu_y\|)/\sum_i w_i \\
\Sigma_{xy} &\leftarrow (\textstyle\sum_i w_i (y_i - \mu_y)(x_i - \mu_x)^T)/\sum_i w_i \\
R &\leftarrow UV^T, \text{ where } UDV^T \text{ is the SVD of } \Sigma_{xy} \\
c &\leftarrow \sigma_y/\sigma_x \\
t &\leftarrow \mu_y - cR\mu_x
\end{aligned}
$$

The original proof of optimality of the transformation [23] is easily adapted to the weighted case. Namely, assuming that the flows from the x_i's to the y_i's are integer, each weighted pairing $\{(x_i, y_i, w_i)\}$ is replaced by w_i unweighted

pairings $\{(x_i^j, y_i^j)\}$, which makes the original proof applicable. Collecting appropriate terms, we get weighted versions of the original equations. Fractional flows are reduced to integer flows by multiplying all fractions by their least common denominator.

5.4 The Final Algorithm

Our algorithm for many-to-many matching is a combination of the previous procedures. Specifically, given two vertex-labeled graphs G_1 and G_2, we first find isometric embeddings of the graphs into low-dimensional normed spaces, obtaining two weighted distributions. We then "register" one distribution with respect to the other so as to minimize the (original) EMD between them. We then apply the FT iteration of the transformation version of the EMD framework [7] to minimize the (extended) EMD. The pairing of points minimizing the EMD corresponds to a weighted many-to-many pairing of nodes. We summarize our approach in Algorithm 1.

Algorithm 1 Many-to-many graph matching

1: Compute the metric tree $\mathfrak{T}_i$ corresponding to G_i (see Section 3.2).
2: Construct low-distortion embeddings $f_i(\mathfrak{T}_i)$ of $\mathfrak{T}_i$ into $(\mathcal{B}_i, \|.\|_2)$ according to Section 3.4.
3: Compute low-distortion embeddings $\mathcal{E}_i = P_i(f_i(\mathfrak{T}_i))$ into $(\mathcal{B}, \|.\|_2)$ according to Section 5.1.
4: Compute the EMD between $\mathcal{E}_i$'s by applying the FT iteration (Section 5.2), computing the optimal transformation T according to Section 5.3.
5: Interpret the resulting optimal flow between $\mathcal{E}_i$'s as a many-to-many vertex matching between G_i's.

6 Experiments

We tested the many-to-many matching algorithm on the COIL-20 database of Columbia University consisting of 72 views per object. A representative view of each object is shown in Figure 3(a). The multi-scale blob decomposition is then computed for each view using the algorithms described in [21] (and illustrated in Figure 1). For the experiments, we compute the tree metric corresponding to the complete edge-weighted graph defined on the regions of the scale-space decomposition of the view. The edge weights are computed as a function of the distances between the centroids of the regions in the scale-space representation. Next, each tree will be embedded into a normed space with low distortion. This procedure results in a database of weighted point-sets, each representing an embedded graph.

To test the matching algorithm on the resulting database, we removed 36 (of the 72) representative views of each object (every other view) and used these as

Fig. 3. Columbia University Image Library (COIL-20) database.

queries to the remaining view database (the other 36 views for each of the 20 objects). We then computed the distance between each "query" view and each of the remaining database views. Ideally, for any given query view i of object j, $v_{i,j}$, the matching algorithm should return either $v_{i+1,j}$ or $v_{i-1,j}$ as the closest view. We will classify this as a correct matching. Figure 4 presents a subset of the matching experiments for Object 9 of the COIL-20 database, with a correct matching in almost all cases.

The results of the experiment are presented in Figure 5, with darker points representing the closer matches. Based on the overall matching statistics, we observed that in all but 10.74% of the experiments, the closest match selected by our algorithm was a neighboring view. This is clearly evident from the darker diagonal entries of Figure 5. Among the mismatches, the closest view belonged to the same object in 80% of the cases. It should be noted that these results can be considered worst case for two reasons. First, the original 72 views per object sampling resolution was tuned for an eigenimage approach. Given the high similarity among neighboring views, it could be argued that our matching criterion is overly harsh, and that perhaps a measure of "viewpoint distance", i.e., "how many views away was the closest match" would be less severe. In any case, we anticipate that with fewer samples per object, neighboring views would be more dissimilar, and our matching results would improve. Second, and perhaps more importantly, many of the objects are symmetric, and if a query neighbor has an identical view elsewhere on the object, that view might be chosen (with equal distance) and scored as an error. Many of the objects in the database are rotationally symmetric, yielding identical views from each viewpoint and likely errors.

Both the embedding and matching procedures can accommodate occlusion. This is due to the fact that the path partitions for unoccluded portions of the graph are unaffected by occlusion. During the projection step, the projections of unoccluded nodes will also be unaffected by occlusion. Finally, the matching procedure is an iterative process driven by flow optimization which, in turn, depends only on local features, and is thereby unaffected by occlusion. We are in the process of conducting occlusion experiments and expect to report them shortly.

Query \ Model												
												
	**2.70**	7.05	6.54	7.78	10.37	5.89	13.41	12.30	20.34	13.90	19.60	19.53
	**4.14**	5.56	5.18	7.98	10.30	4.24	12.34	11.23	20.01	12.24	18.40	17.73
	6.39	**2.34**	2.68	4.17	5.97	5.94	17.03	15.87	25.28	16.74	22.99	22.17
	6.07	4.04	4.04	**3.17**	4.44	6.64	17.26	15.92	25.82	17.17	23.53	22.80
	7.27	6.39	6.55	5.31	**3.88**	8.26	18.20	16.88	26.74	17.85	24.57	23.81
	5.31	4.20	5.25	5.67	**3.21**	5.63	17.08	15.86	25.20	17.07	23.49	22.79
	9.61	11.65	11.21	13.81	16.00	**6.80**	7.07	8.20	14.92	9.05	13.74	14.65
	13.64	15.32	14.85	17.35	19.28	11.80	**2.69**	3.70	14.20	6.75	10.93	12.19
	14.34	16.03	15.23	17.92	19.90	12.21	5.28	**3.54**	14.61	4.61	8.96	10.33
	13.50	14.90	14.41	17.39	19.32	11.44	6.56	**4.13**	15.00	5.25	8.97	9.98
	17.16	18.97	18.34	21.28	23.11	15.70	7.95	7.85	13.52	**4.23**	4.73	6.17
	20.53	22.30	21.25	24.17	26.18	18.77	11.46	11.59	14.48	7.14	3.02	**2.75**
	20.19	20.90	19.92	22.89	24.87	18.18	12.19	12.27	14.91	7.94	6.53	**3.24**

Fig. 4. Sample matching results for object 9 of the COIL-20 database, in which rows and columns can be interleaved to form the set of sequential views. The diagonal and next lower diagonal therefore represent the neighboring views of the query (row). Only one query, entry (10,8), was incorrectly matched.

7 Conclusions and Future Work

We have presented a computationally efficient approach to many-to-many matching of multi-scale feature representations in terms of blobs and ridges. The approach is based on a combination of metric tree representation and low-distortion embedding of graphs to normed spaces with a distribution-based similarity measure. The matching framework attempts to exploit both topological and geometrical properties of multi-scale feature hierarchies. Due to the strengths of the two components, our approach is able to establish robust, many-to-many correspondences in the presence of noise. Preliminary matching experiments on the COIL-20 database demonstrate that our method performs very well subject to view sampling constraints. Our work on matching of multi-scale features is in

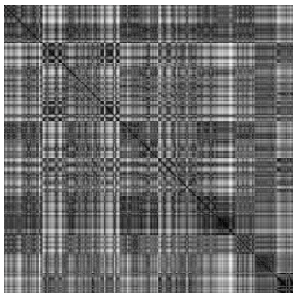


Fig. 5. The matching results for the COIL-20 database. The rows represent the query views (36 views per object), and the columns representing model views (36 views per object). Each row represents the matching results for a query view against the whole database. The intensity of entries represents the quality of the matching, with black representing maximum similarity between the views and white minimum similarity.

its preliminary stages, and we plan to extend its scope in several directions: 1) to study the viewpoint invariance of the multi-scale blob decomposition within our many-to-many matching framework; 2) to study the initial conditions of the FT iteration to improve matching results; 3) to exploit the possibility of using embedded vector representations as signatures for indexing purposes; and 4) to conduct occlusion and noise sensitivity experiments.

8 Acknowledgment

Ali Shokoufandeh acknowledges the partial support provided by a grant from National Science Foundation (NSF) ITR-DM-0219176. The work of Yakov Keselman is supported in part by the NSF grant No. 0125068. Sven Dickinson acknowledges the support of Natural Science and Engineering Research Council of Canada.

References

1. R. Agarwala, V. Bafna, M. Farach, M. Paterson, and M. Thorup. On the approximability of numerical taxonomy (fitting distances by tree metrics). *SIAM Journal on Computing*, 28(2):1073–1085, 1999.
2. R. K. Ahuja, T. L. Magnanti, and J. B. Orlin. *Network Flows: Theory, Algorithms, and Applications*, pages 4–7. Prentice Hall, Englewood Cliffs, New Jersey, 1993.
3. S. Belongie, J. Malik, and J. Puzicha. Shape matching and object recognition using shape contexts. *IEEE PAMI*, 24(4):509–522, April 2002.
4. R. Beveridge and E. M. Riseman. How easy is matching 2D line models using local search? *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(6):564–579, June 1997.
5. J. Bourgain. On Lipschitz embedding of finite metric spaces into Hilbert space. *Israel Journal of Mathematics*, 52:46–52, 1985.

6. P. Buneman. The recovery of trees from measures of dissimilarity. In F. Hodson, D. Kendall, and P. Tautu, editors, *Mathematics in the Archaeological and Historical Sciences*, pages 387–395. Edinburgh University Press, Edinburgh, 1971.
7. S. D. Cohen and L. J. Guibas. The earth mover's distance under transformation sets. In *Proceedings, 7th International Conference on Computer Vision*, pages 1076–1083, Kerkyra, Greece, 1999.
8. Steven Gold and Anand Rangarajan. A graduated assignment algorithm for graph matching. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(4):377–388, 1996.
9. A. Gupta, I. Newman, Y. Rabinovich, and A. Sinclair. Cuts, trees and l_1 embeddings. *Proceedings of Symposium on Foundations of Computer Scince*, 1999.
10. P. Indyk. Algorithmic aspects of geometric embeddings. In *Proceedings, 42nd Annual Symposium on Foundations of Computer Science*, 2001.
11. S. Kosinov and T. Caelli. Inexact multisubgraph matching using graph eigenspace and clustering models. In *Proceedings of SSPR/SPR*, volume 2396, pages 133–142. Springer, 2002.
12. N. Linial, E. London, and Y. Rabinovich. The geometry of graphs and some of its algorithmic applications. *Proceedings of 35th Annual Symposium on Foundations of Computer Scince*, pages 557–591, 1994.
13. T.-L. Liu and D. Geiger. Approximate tree matching and shape similarity. In *Proceedings, 7th International Conference on Computer Vision*, pages 456–462, Kerkyra, Greece, 1999.
14. B. Luo and E.R.Hancock. Structural matching using the em algorithm and singular value decomposition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23:1120–1136, 2001.
15. J. Matoušek. On embedding trees into uniformly convex Banach spaces. *Israel Journal of Mathematics*, 237:221–237, 1999.
16. M. Pelillo, K. Siddiqi, and S. Zucker. Matching hierarchical structures using association graphs. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21(11):1105–1120, November 1999.
17. Y. Rubner, C. Tomasi, and L. J. Guibas. The earth mover's distance as a metric for image retrieval. *International Journal of Computer Vision*, 40(2):99–121, 2000.
18. G. Scott and H. Longuet-Higgins. An algorithm for associating the features of two patterns. *Proceedings of Royal Society of London*, B244:21–26, 1991.
19. T. Sebastian, P. Klein, and B. Kimia. Recognition of shapes by editing shock graphs. In *IEEE International Conference on Computer Vision*, pages 755–762, 2001.
20. L. G. Shapiro and R. M. Haralick. Structural descriptions and inexact matching. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 3:504–519, 1981.
21. A. Shokoufandeh, S.J. Dickinson, C. Jönsson, L. Bretzner, and T. Lindeberg. On the representation and matching of qualitative shape at multiple scales. In *Proceedings, 7th European Conference on Computer Vision*, volume 3, pages 759–775, 2002.
22. K. Siddiqi, A. Shokoufandeh, S. Dickinson, and S. Zucker. Shock graphs and shape matching. *International Journal of Computer Vision*, 30:1–24, 1999.
23. S. Umeyama. Least-squares estimation of transformation parameters between two point patterns. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(4):376–380, April 1991.
24. M. S. Waterman, T. F. Smith, M. Singh, and W. A. Beyer. Additive evolutionary trees. *J. Theor. Biol.*, 64:199–213, 1977.