International Journal of Shape Modeling

Non-Rigid Spectral Correspondence of Triangle Meshes

Varun Jain, Hao Zhang, and Oliver van Kaick
Graphics, Usability and Visualization (GrUVi) Lab
School of Computing Science, Simon Fraser University
Burnaby, BC, Canada
vjain,haoz,ovankaic@cs.sfu.ca

Received (Day Month Year)
Revised (Day Month Year)
Accepted (Day Month Year)

Communicated by (xxxxxxxxxx)

We present an algorithm for finding a meaningful vertex-to-vertex correspondence between two triangle meshes, which is designed to handle general non-rigid transformations. Our algorithm operates on embeddings of the two shapes in the spectral domain so as to normalize them with respect to uniform scaling and rigid-body transformation. Invariance to shape bending is achieved by relying on approximate geodesic point proximities on a mesh to capture its shape. To deal with moderate stretching, we first raise the issue of "eigenmode switching" and discuss heuristics to bring the eigenmodes to alignment. For additional non-rigid discrepancies in the spectral embeddings, we propose to use non-rigid alignment via thin-plate splines. This is combined with a refinement step based on geodesic proximities to improve dense correspondence. We show empirically that our algorithm outperforms previous spectral methods, as well as schemes that compute correspondence in the spatial domain via non-rigid iterative closest points or the use of local shape descriptors, e.g., 3D shape context. Finally, to speed up our algorithm, we examine the effect of using subsampling and Nyström method.

Keywords: Non-rigid ICP; eigenvector switching; principal component analysis; shape correspondence; spectral embedding; thin plate splines.

1991 Mathematics Subject Classification: 22E46, 53C35, 57S20

1. Introduction

Given two 3D shapes represented as triangle meshes, the correspondence problem seeks to establish a meaningful mapping between them. The mapping can be between the two sets of mesh vertices, between two coarse sets of feature points selected on the meshes, or a continuous one between all points on the two shapes. This is a fundamental problem in computer graphics and shape modeling, with such applications as texture mapping[25], mesh deformation[2,43,46], shape registration[28,35], and object recognition[21]. Also, a recent trend in research into the mesh parameterization problem, which essentially computes a continuous dense surface corre-

spondence, is to look for effective techniques to construct a *cross parameterization* between two meshes directly[26,36], i.e., without relying on a simple common parameter domain. Such a parameterization allows one to obtain compatible connectivity among a set of models in a feature-sensitive way[26,34], greatly facilitating tasks such as shape blending and various forms of attribute transfer. A crucial first step for constructing a cross parameterization is the identification of a sparse set of matching feature points on the two shapes; this is followed by patch construction based on the corresponding features. Most such methods[34,25,26,36] rely on the user to specify the initial feature correspondence manually. It is well known that even if the sets of feature points have been given, finding a meaningful correspondence automatically with robustness to both rigid and non-rigid geometric transformations is notoriously difficult. This is the problem we wish to address in this paper.

Given two triangle meshes, possibly non-manifolds (but connected) and with different sizes, we compute a correspondence between the mesh vertices. The spatial coordinates of the mesh vertices are first converted into two affinity matrices, which are obtained by applying a Gaussian kernel to the matrices of approximate *geodesic distances* for the two meshes. This way, each vertex of a mesh is represented using intrinsic structural information. The correspondence is obtained by matching points based on this information and this is carried out in a k-dimensional *spectral* domain. Typically, k is much smaller than the size of the affinity matrices. Thus the dimensionality of the structural information is effectively reduced.

Appropriate choice of the Gaussian kernel width and the use of spectral embeddings described above ensure that our matching procedure is invariant to rigid body transformations, uniform scaling, and shape bending, since the affinities used are invariant to precisely these transformations. We also propose to scale the eigenvectors by the square root of the eigenvalues in forming the spectral embeddings so as to achieve robustness against difference in mesh vertex counts and choice of the dimensionality of the embeddings. However, our experiments show that matching based solely on such embeddings, e.g., using the L_2 metric[40], can be non-robust to stretching in the shapes. A crucial observation is that stretching can cause certain eigenmodes to *switch places*. That is, eigenvectors which form the spectral embeddings may not correspond according to the orderings determined by the magnitude of their eigenvalues. To the best of our knowledge, this issue has not been addressed before and all previous spectral correspondence algorithms[6,8,9,12,37,38,40] have ordered the eigenmodes according to the magnitude of their eigenvalues.

In this paper, we first search for an appropriate permutation of the eigenmodes, as well as their sign assignments. This procedure effectively aligns the spectral embeddings with respect to certain reflections. To handle the remaining non-rigid discrepancies between matched shapes, we perform a non-rigid alignment in the spectral domain using thin-plate splines (TPS)[5]. A refinement step using the original geodesic proximity data enhances the performance of the algorithm in the case of dense correspondence. Through formal arguments and numerous experiments, we demonstrate that our approach outperforms previous methods which operate

on spectral embeddings[8,9,40,45], as well as schemes that find correspondence in the spatial domain via non-rigid iterative closest points[7] or the use of local shape descriptors, such as shape contexts[3] or curvature maps[16].

The rest of the paper is organized as follows. Section 2 gives a brief survey of previous work. Section 3 defines notations, formalizes our problem, and provides an algorithm overview. In Section 4, we describe the construction of the spectral embeddings in more details, argue for its various properties, and demonstrate its non-robustness to stretching. We also explain our eigenvector scaling scheme and its effect on the spectral embeddings. In Section 5, we present the major components of our correspondence algorithm. This is followed by experimental results in Section 6, where we also point out some limitations of our current approach. Finally, we conclude in Section 7 and comment on future work.

2. Background and previous work

Visual correspondence constitutes one of the most fundamental aspects of human intelligence, but the difficulty with which an automatic computer algorithm can imitate this process has long been realized in the field of computer vision. The vision community has mostly focused on 2D correspondence, either between contours[39] or image features[3,4,7,8,9,31], e.g., for object tracking, image registration, and motion analysis[44]. Although in certain aspects, the 2D problem may appear to be more difficult than its 3D counterpart due to factors such as occlusion and illumination artifacts, the latter can offer a different set of challenges. First of all, temporal coherence and spatial proximity can greatly facilitate correspondence computations in vision applications, but the same cannot be said about 3D applications such as texture mapping, mesh morphing and deformation, or object recognition. Secondly, an increase in dimensionality introduces more degrees of freedom. Last but not the least, unlike images or volumes, general 3D surfaces lack a canonical parameterization which complicates matters. As an example relevant to the correspondence problem, we mention that order-preserving assignment for contour matching can be done in polynomial time[39], but the problem of optimizing for a neighbor-preserving assignment between two surfaces is much more complex[a].

In general, point correspondence may be computed based on either absolute coordinates or relative information, e.g., given by weighted graphs. Two main classes of methods exist for spatial correspondence: those using local shape descriptors and those relying on iterative alignment schemes. The first class of techniques describe every point on a shape by encoding shape information from the perspective of that point. Point matching is then based on appropriate distances between the descriptors. Well-known descriptors for images[31] include shape contexts[3] and spin images[22], both utilizing a histogram obtained by binning the space around a point

[a]One possible formulation is via the Quadratic Assignment Problem[33], which is arguably one of the most difficult NP-hard combinatorial optimization problems.

(a) Approximately aligned shapes. (b) Correct matching is obtained. (c) One of the shapes is rotated. (d) Incorrect matching obtained.

Fig. 1. TPS-RPM is sensitive to initial configuration of shapes.

according to the Euclidean metric and collecting point counts. The former has been generalized in a straightforward manner to handle 3D point sets[23].

Neither shape contexts nor spin images are invariant to shape bending. A promising remedy is the curvature maps of Gatzke et al.[16]. Local shape descriptors are constructed on a mesh by dividing the geodesic neighborhood of a vertex into bins. Although geodesic binning is invariant to bending (not stretching), the histograms computed are based on curvature distributions, which, even if estimated robustly, are not invariant to bending.

More recent shape signature based methods include those by Gelfand et al.[17] and Li et al.[28]. Both methods are robust under only rigid transformations. The partial matching scheme of the former do provide a way to detect articulated subparts of a shape and subsequently match them to the subparts of the second shape. However, their work lacks a discussion and analysis of the performance and robustness of the matching method when there are multiple pose changes in the models or when the models consist of a large number of articulated subparts.

Iterative alignment schemes compute a correspondence and a transformation which would transform one shape into another at each step. The correspondence is usually based on a "closest point" criteria and the transformation is obtained by optimizing an energy. Such techniques include the well known iterative closest point (ICP) algorithm of Besl and Mckay[4] and its variants[35], which can handle affine transformations. Recent works, most notably the TPS-RPM method of Chui and Rangarajan[7], attempt to incorporate non-rigid deformations into the ICP framework, using thin-plate splines (TPS) to model the deformation. However, these methods can easily get trapped in bad local minima if the shapes are not approximately aligned initially, since the correspondence, which dictates the optimization, is computed using an Euclidean closest point method. For example, rotation alone can cause a bad matching, as shown in Figure 1.

Sumner et al.[43] and Zayer et al.[46] attempt to alleviate this problem by fixing a small number of feature points on the shapes to be matched. Sumner et al. rely on these feature points as guidance to ICP in order to escape local minima, whereas Zayer et al. use interpolation, based on barycentric coordinates, of the correspondence between the feature points to compute the remaining point correspondences. In both methods, it is imperative that the feature points selected on the two meshes be corresponding in a meaningful way. This can only be done with user assistance

as automatic selection and matching of the feature points is equivalent to the correspondence problem we are trying to solve in the first place.

Spectral shape correspondence involves first constructing intrinsic (relative) point representations of the two shapes, in the form of weighted graph adjacency or affinity matrices. Elad and Kimmel[12] make use of geodesic proximities to construct bending-invariant surface signatures through multi-dimensional scaling. Application to object classification has been considered, but they do not solve the harder correspondence problem. Given a proximity matrix, a k-dimensional spectral embedding can be computed via principal component analysis (PCA). Shapiro and Brady[40] use L_2 distances between the embedded points to compute a correspondence, while Umeyema[45] chooses the correlation between the embedding coordinates. Both Caelli and Kosinov[6] and Carcassoni and Hancock[9] rely on spectral clustering and cluster correspondence to guide point correspondence. The use of spectral embeddings has traditionally been exploited in the computer vision and machine learning literature. Recently, they have found several applications in geometry processing as well, including mesh segmentation[27], spherical parameterization[19], surface reconstruction[24], object retrieval[12,21], and surface quadrangulation[10].

3. Overview

Let us first give a brief overview of the problem we address and the algorithm we propose. Given two 3D shapes M_A and M_B, in the form of triangle meshes and with n_A and n_B vertices, respectively, we wish to compute a correspondence Y between the two sets of vertices in M_A and M_B. That is, $Y(i)$ is the vertex in M_B that best corresponds to vertex i in M_A. Note that the correspondence computed is not required to be bijective. However, in the case where $n_A = n_B$ and a one-to-one correspondence is sought, we can easily modify our method to meet the goal.

Our method of computing Y is as follows: first, we establish an $n_A \times n_A$ affinity matrix A where A_{ij} is the affinity between vertices i and j of M_A. Similarly, we compute an $n_B \times n_B$ matrix B, the affinity matrix for M_B. The affinities that we use in our implementation are based on distances over surfaces in order to attain invariance to bending. Next we find the spectral embeddings $\hat{A}_k$ and $\hat{B}_k$ of the matrices A and B, respectively. These embeddings give k-dimensional coordinates of all the vertices of M_A and M_B. The embeddings are based on the eigenvectors of A and B, properly processed as we describe in Section 4. The purpose of transforming the 3D mesh from spatial domain to spectral domain is to attain invariance to bending, rigid transformations, and uniform scaling, as well as robustness to difference in mesh sizes. Now Y is computed in the spectral domain.

Common to all the existing spectral correspondence techniques is the premise that the eigenmodes from two similar shapes should match up, according to the magnitude of their corresponding eigenvalues. One of our main observations is that this ordering of the eigenmodes is not always reliable. As the eigenvalues characterize data variance in the direction of the corresponding eigenvectors, eigenmode

ordering based on eigenvalues implies ordering by data variance. This may not be appropriate since variance only captures global information and does not reflect the way specific data points would vary. Under shape stretching, certain eigenmodes may be "switched". Failure to resolve such reflections or other non-rigid discrepancies between spectral embeddings will lead to poor matching results. In this paper, we examine heuristics to handle eigenmode switchings. Afterwards, the two k-D spectral embeddings will be corresponded via non-rigid ICP registration based on TPS. This registration is not required to be one-to-one, but we can force bijectivity by using the Hungarian algorithm for bipartite matching.

4. Spectral embedding

In general, intrinsic point representations can be obtained via pairwise point proximities, specified by a symmetric *affinity matrix* $A = \{a_{ij}\}$, where $a_{ij} \geq 0$ characterizes the similarity or simply the graph adjacency[6,45] between points i and j. One may view the affinity matrix A as a data vector whose n columns (or rows) represent n-dimensional data points.

The most common proximity measure used to define relationship between points for shape matching is the Euclidean distance[8,9,38,40], which implies invariance to rotation and translation. For mesh correspondence, we may use geodesic distances between the mesh vertices, computed via fast marching[12], to include invariance to bending as well. But to be able to handle non-manifold situation, which occurs for many of our test models taken from the Princeton Benchmark[41], we use graph distances in a mesh to approximate geodesic distances. The graph is composed simply of mesh vertices and edges with edge weights given by edge lengths. Invariance to uniform scaling is achieved by mapping the approximate geodesic proximities into the interval $[0, 1]$ using a *scale-dependent* kernel function. In this paper, we use Gaussian kernels which is a common choice for spectral correspondence[8,9,38,40].

Although the point proximities contain a great deal of shape information, without a proper point mapping, one cannot compare such representations for two data sets directly. Also, the size of the data sets, or the dimensionality of the point representations, may not be the same. Last but not the least, the high dimensional representations may contain a great deal of redundancy, resulting in unnecessarily high computational cost. These observations naturally lead us to consider transforming two data sets, respectively, into some information-preserving subspaces that share the same low dimensionality. This can be accomplished through principal component analysis (PCA) on the affinity data.

4.1. *Principal component analysis*

Given the data (affinity) matrix $A \in \mathbf{R}^{n\times n}$, we first compute its principal components $\mathbf{e_1}, \ldots, \mathbf{e_n}$, which are the *normalized* eigenvectors of the autocorrelation matrix $R = AA^T$. Since A is symmetric, $R = A^2$ and $\mathbf{e_1}, \ldots, \mathbf{e_n}$ are simply the

Fig. 2. Spectral embeddings (bottom row) of some articulated 3D shapes (top row) from the McGill shape database. The embeddings are constructed using the second, third, and fourth eigenvectors. The ignoring of the first eigenvector is explained in Section 5.

eigenvectors of the affinity matrix A. Let $\lambda_1, \ldots, \lambda_n$ be the corresponding eigenvalues of A and suppose that $\lambda_1 \geq \ldots \geq \lambda_n$. Projecting the data matrix onto the first k principal components yield

$$\hat{A}_k = E_k^T A = \Lambda_k E_K^T, \tag{1}$$

where $E_k = [\mathbf{e_1}|\ldots|\mathbf{e_k}]$, $\Lambda_k = diag(\lambda_1, \ldots, \lambda_k)$, and the columns of $\hat{A}_k$ represent a k-dimensional *spectral embedding* of the data points. A point permutation induces the same permutation of the embeddings but leaves the spectrum invariant.

In the case of mesh spectral embeddings, the data points are mesh vertices. A spectral embedding associates with each mesh vertex a k-dimensional vector. In the 3D spectral domain, one can visualize the embedding of a mesh M by rendering a mesh whose connectivity is the same as M and whose vertices are given by the embedding coordinates, as shown in Figure 2, where the original shapes were taken from the McGill articulated shape database[30]. It is worth noting the normalization of shape articulation in the spectral domain, which is quite evident.

Although $\hat{A}_k$ gives a provably best k-dimensional approximation of A (in terms of the Frobenius norm), it may not be suitable for matching. The more important requirement is for the projection axes, derived from the principal components, to be compatible between two data sets.

4.2. *Eigenvector scaling*

Given two affinity matrices A and B characterizing two shapes, possibly in different scales, a scale-dependent kernel can normalize the affinity values in A and B. If the number of vertices, n_A and n_B, in the two (mesh) shapes differ however, we first need to truncate both spectral embeddings to the same dimension $k \leq \min\{n_A, n_B\}$. In addition, since we normalize each eigenvector, the cardinality of a data set affects the magnitude of the entries in its eigenvectors, which in turn affects the embeddings.

Correspondence algorithms that use un-scaled eigenmodes[8,9,40] as spectral embeddings are common. Shapiro and Brady[40] first suggest a scaling of the the eigenmodes by eigenvalues, as in (1), but did not elaborate. Caelli and Kosinov[6] scale the eigenmodes using *squared* eigenvalues and then project the resulting embeddings

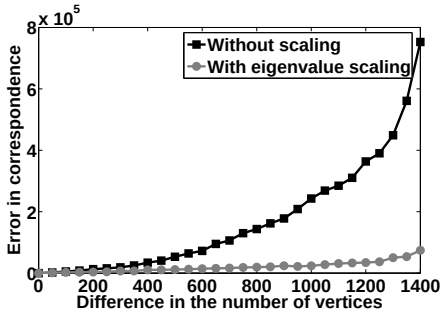


Fig. 3. Effect of eigenvector scaling on correspondence, based on the correspondence error plots averaged over several test models taken from the Princeton Shape Benchmark.

onto the unit k-sphere for matching graphs of different vertex counts. Evidently, proper scaling of the eigenmodes is a crucial normalization step. None of these approaches adequately resolves the discrepancies in the scales of the principal components (or embeddings) due to difference in the cardinality of the data sets. In this paper, we propose to scale the principal components by the *square root* of the eigenvalues, yielding projections

$$\hat{A}_k = \Lambda_k^{\frac{1}{2}} U_k^T \qquad \text{and} \qquad \hat{B}_k = \Gamma_k^{\frac{1}{2}} V_k^T, \tag{2}$$

where $A = U\Lambda U^T$ and $B = V\Gamma V^T$ are the eigenvalue decompositions of A and B, respectively, with $U_k, V_k, \Lambda_k, \Gamma_k$ defined as in (1). Spectral embeddings of this form are well known in the spectral clustering literature[32].

Justifications: Consider the vector of projections $\hat{\mathbf{a}}_i$ from set A. We can estimate the scale of these projections by $s_{A,i} = ||\hat{\mathbf{a}}_i||^2/n_A$. From (2), we have

$$\hat{\mathbf{a}}_i = \sqrt{\lambda_i}\mathbf{u_i} \;\text{ and }\; \hat{\mathbf{b}}_\mathbf{i} = \sqrt{\gamma_\mathbf{i}}\mathbf{v_i}.$$

It follows that $s_{A,i} = \lambda_i/n_A$ and $s_{B,i} = \gamma_i/n_B$. With the affinity matrices having unit diagonal elements, signaling that a point has maximal affinity to itself, we have

$$s_{A,i} = \frac{\lambda_i}{trace(A)} = \frac{\lambda_i}{\sum_{j=1}^{n_A}\lambda_j} \;\text{ and }\; s_{B,i} = \frac{\gamma_i}{\sum_{j=1}^{n_B}\gamma_j}.$$

We do not normalize these scales to some constant, since they represent data variations along the projection axes and thus contain shape information. We only wish to remove the effect of different data size; this is achieved by normalizing the eigenvalues, which represent data variations.

Another justification for (2) is that the dot-product matrices $\hat{A}_k^T\hat{A}_k$ and $\hat{B}_k^T\hat{B}_k$ are respectively the best rank-k approximations, in Frobenius norms, of A and B[11], which are already normalized to scale. Using the same argument, we see that the dot product matrices resulting from eigenmode scaling with eigenvalues themselves[40] become best rank-k approximations of the autocorrelation matrices $AA^T = A^2$ and

Fig. 4. Eigenvector plots for two shapes, both with 252 vertices. The first 8 eigenvalues are [205.6, 11.4, 4.7, 3.8, 1.8, 0.4, 0.26, 0.1] and [201.9, 10.9, 6.3, 3.4, 1.8, 1.2, 0.31, 0.25], respectively.

B^2, respectively, whose entries do depend on the size of the data sets. Finally, one can also justify the scaling by making use of the notion of kernel PCA, see[20].

Experimental verification using correspondence errors: The effectiveness of our eigenvector scaling scheme is shown in Figure 3, where we plot the correspondence errors in the case of scaled versus un-scaled spectral embeddings. The correspondence error is measured as the total graph distance $\sum_{i=1}^{n} d(v_i, v_i')$, where n is the number of vertices to be matched, v_i is the vertex corresponding to i that is computed by an algorithm, and v_i' is the ground-truth match for i. The plots are against the difference in the number of vertices of the two meshes to be matched. The correspondence algorithm used is our own and it is described in Section 5. In producing the two plots, the only difference is whether the eigenmodes are scaled.

Measuring error for dense correspondence is not easy since the ground-truth correspondence is impractical to establish manually. In our evaluation, we successively decimated a 3D mesh using the QSlim mesh decimation program of Garland[15]. Next we use our algorithm to correspond the original mesh with its coarsified versions and measure correspondence errors. The ground-truth can be trivially established since QSlim retains the positions of undecimated vertices.

4.3. *Non-robustness of eigenmodes*

Eigenmode switching: Perturbation theory predicts that when eigenvalues move close to each other, the corresponding eigenvectors may switch order[18]. Geometrically, an eigenmode switching corresponds to a reflection of the spectral embedding about the symmetry axis between the two axes corresponding to the switched eigenmodes. We have observed that eigenmode switching in spectral shape correspondence can occur early in the eigenvalue order, e.g., before the 8-th eigenmode, even when the two shapes being matched are perceptually similar. But there is no general pattern of eigenvalue clustering that is sufficiently reliable to detect the switchings. As switching of two coordinates, the eigenvectors, induces a reflection in the spectral domain, spectral correspondence based on the L_2 distance measure or correlations[40,45], even with the aid of clustering[6,9], can fail.

For a visual illustration of eigenmode switching, we color-plot the eigenvectors in MATLAB, where the entries in an eigenvector are used as indices into the color map. To enhance our illustration, we nonlinearly warp the color map. As shown in Figure 4, two similar shapes have compatible eigenmodes, reflected by consistent

color plots, only up to the 4^{th} eigenvector. The 5^{th} and 6^{th} eigenmodes are switched and color patterns for the next a few eigenvectors, those exhibiting higher-frequency color variations, do not exhibit any discernible patterns. Evidently, correspondence analysis using eigenvalue orderings to pair up eigenmodes beyond the 4^{th} one would be hard to justify in this case.

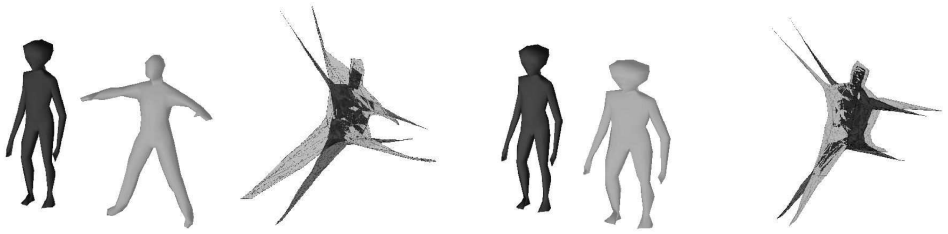
Other effects of eigenvector scaling: One interesting point to note is that as the magnitude of the eigenvalues of the approximate geodesic affinity matrices exhibit rapid decay, as shown in the caption of Figure 4, scaling using eigenvalues has the effect of rapidly attenuating the effects of higher-frequency eigenvectors. This would be quite appropriate since these eigenvectors are less reliable to use for correspondence analysis. As a side effect, the resulting correspondence algorithm will be less sensitive to the number of eigenmodes chosen. In previous works, e.g.,[9], some heuristic has to be adopted to determine the proper dimensionality to use.

Sign flips: Arbitrary determination of the signs of the eigenvectors returned by a numerical solver introduces another form of reflection to be handled, as already noted in previous works[6,40]. Caelli and Kosinov[6] propose to use a dominant sign correction, always ensuring that there are more positive entries in each eigenvector. This is highly unreliable however since specific to spectral correspondence, the eigenvectors tend to have about the same number of positive and negative entries due to orthogonality to a constant eigenvector; see Section 5.1. Shapiro and Brady[40] use a greedy approach to correct one sign at a time by optimizing for a correspondence cost. In the presence of eigenmode switchings, this is clearly not robust.

Other transformations: Consider the spectral embeddings of two similar human meshes using the 2^{nd}, 3^{rd} and 4^{th} eigenvectors (this particular choice of the eigenvectors is explained in Section 5.1), as shown in Figure 5(a). Ideally, the embeddings would be perfectly aligned. However, a rotational difference in the embeddings is clearly visible. In addition, there are also other discrepancies of a non-rigid nature. Another example is given in Figure 5(b), where the second (light gray) mesh is merely a scaled version (scaled along the x direction) of the first (dark gray) mesh. But there again is a rotation in the embedding. We believe that such transformations in the spectral domain, as well as eigenmode switchings, are the result of non-uniform stretching in the shapes. Obviously, a matching algorithm must be able to deal with all these transformations in order to operate robustly.

5. Our algorithm

An outline of our non-rigid spectral mesh correspondence algorithm is given in Figure 6, with details described below. Note that iterative alignment, e.g.,[7], can work quite well if the initial shapes are approximately aligned, while spectral embedding can automatically remove the effects of rigid-body transformations, uniform scaling, and shape bending. Hence, a natural approach would be to perform non-rigid alignment in the spectral domain before computing the matching. The only obsta-



(a) Alien vs. human, and their embeddings. (b) Alien stretched and the embeddings.

Fig. 5. Rotational and non-rigid discrepancies between similar meshes in the spectral domain. (a) A skinny alien vs. a well-proportioned man. (b) The skinny alien (left) is stretched horizontally.

NonRigidSpecCorr(M_A, M_B: two meshes with $n_A \leq n_B = n$ vertices)

(1) $A, B \leftarrow$ Gaussian affinity matrices for M_A and M_B — $\Theta(n^2 \log n)$.
- Compute pairwise graph distances using Dijkstra's algorithm.
- The time complexity can be reduced to $O(pn \log n)$ using Nyström approximation, where $p \ll n$ is the number of samples. See Section 5.1.

(2) $\hat{A}, \hat{B} \leftarrow (k-1)$-dimensional spectral embeddings — $O(kn^2)$.
- With Nyström, this step becomes $O(p^3 + p^2 n)$.

(3) Eigenvector scaling of spectral embeddings — $O(kn)$.
(4) Eigenmode permutation and sign assignments. — depends on heuristic.
(5) Non-rigid alignment using TPS returns $n_A \times n_B$ dissimilarity matrix Z — $O(ln^2)$, with l iterations. In all of our experiments, $5 \leq l \leq 10$.
(6) $Y \leftarrow$ correspondence via best matching and proximity guidance — $O(n^2)$.
- Or optimal 1-to-1 matching via the $O(n^3)$ Hungarian algorithm.

Fig. 6. NonRigidSpecCorr(): Our non-rigid spectral mesh correspondence algorithm and associated asymptotic time complexity of each step.

cle now is to handle reflections caused by eigenvector switching and sign flips, as they can introduce large discrepancies into the initial shape configurations, which will likely compromise the performance of non-rigid registration.

5.1. *Geodesic affinities, spectral embeddings, and Nyström method*

Consider two 3D meshes M_A and M_B with $n_A \leq n_B$ vertices, respectively. We first construct Gaussian affinity matrix A with $A_{ij} = exp(-d_{ij}^2/\sigma_A^2)$, where d_{ij} is the graph or approximate geodesic distance between vertex i and j in M_A. The Gaussian kernel width σ_A is set to be the maximum geodesic distance between any two vertices in M_A. The performance of our method is relatively invariant to the choice of σ_A as long as it is set to a sufficiently large value. Similarly, we construct

B, the Gaussian affinity matrix for mesh M_B. The affinity matrices A and B are then eigenvalue decomposed and the resulting spectrum are truncated to k. Each of the k eigenvectors is scaled with the square root of its corresponding eigenvalue. These steps have already been described in details in Section 4.1 and 4.2.

Note that if the Gaussian width is sufficiently large, the row sums of the affinity matrix are almost constant. As a result, the first eigenvector of the matrix will be close to a constant vector and can be safely ignored. From now on, we denote by $\hat{A} \in \mathbf{R}^{n_A \times (k-1)}$ and $\hat{B} \in \mathbf{R}^{n_B \times (k-1)}$, as first defined in Equation (2), the $(k-1)$-dimensional embeddings of M_A and M_B, respectively, where the first eigenvector is disregarded. $\hat{A}$ and $\hat{B}$ are essentially $n_A \times (k-1)$ and $n_B \times (k-1)$ matrices where the i^{th} rows of $\hat{A}$ and $\hat{B}$ are the $(k-1)$-dimensional spectral embedding coordinates of the i^{th} vertices of meshes M_A and M_B respectively. In all our experiments, we have used $k = 5$ or 6 hence giving a 4 or 5-dimensional spectral embedding.

To reduce the complexity of affinity and eigenvector computations, we can employ a well-known technique called Nyström approximation[13], explained in detail in Appendix A. Using Nyström, we never compute the full affinity matrix A, instead, a small number p of A's rows are sampled. An $p \times p$ eigenvalue problem is solved and extrapolating its eigenvectors, we obtain approximated leading eigenvectors of A. In all of our experiments, we choose $p = 20$ samples using farthest point sampling[29].

5.2. *Eigenvector reordering and sign correction*

We keep the ordering and signs of the eigenvectors of one mesh, e.g., M_A, fixed. The most straightforward way to search for a matching eigenvector ordering and sign assignment for M_B is simply to consider all combinations, all $k! \times 2^k$ of them. We can also employ a simple, greedy heuristic. Let us first consider a very low dimensional embedding, e.g., with only two eigenvectors. We exhaustively find the best possible ordering and signs of these few eigenvectors. Now we incrementally add one eigenvector at a time and at each step, compute the best possible position and sign of the new eigenvector. This results in $O(k^2)$ possibilities to compare, greatly reducing the time complexity. Another possible heuristic would be to perform pairwise swaps of the eigenmodes. Due to the rapid decay of eigenvalues and eigenvector scaling, we never find it necessary to use more than $k = 6$ eigenvectors to arrive at a satisfactory mesh correspondence. So k is always small.

With either the exhaustive search or the greedy heuristic, we need to estimate the cost of a correspondence, which we describe below. First, we obtain a best matching Y based simply on the L_2 metric; other metric, such as the Chi-square or Mahalanobis distance is also possible. Specifically, for a vertex v_i^A of mesh M_A, with best matching, $v_{Y(i)}^B$ is the corresponding vertex of M_B, where

$$Y(i) = \text{argmin}_j \|\hat{A}_i - \hat{B}_j\|. \tag{3}$$

Here $\hat{A}_i$ and $\hat{B}_j$ denote the spectral embedding coordinates of vertex i in mesh M_A

and j in mesh M_B, respectively. Cost of the correspondence Y is simply the sum:

$$cost(Y) = \sum_{i=1}^{n_A} \|\hat{A}_i - \hat{B}_{Y(i)}\|$$

We choose the ordering and signs of the eigenvectors for mesh M_B which give the minimum $cost(Y)$. Note that even with a cost as simple as L_2, finding an optimal solution is a hard global optimization problem.

An alternative to finding an optimal ordering and sign assignment is to utilize *reflection-invariant* shape descriptors in the spectral domain. To this end, we have experimented with high-dimensional shape context and transforms via symmetric polynomials[1] but without a great deal of success. This agrees with findings by others, e.g.,[14], that more invariance properties tend to render the descriptors less descriptive or shape distinguishing, compromising correspondence performances.

5.3. *Non-rigid alignment*

Once the eigenvectors for two shapes have consistent ordering and signs, we wish to transform one embedding into another. Due to the presence of non-rigid deformations in the spectral domain, we modify the original rigid ICP algorithm[4] by replacing its transformation model with the use of TPS. TPS is well-known and has been applied to model non-rigid transformations before [3,7] for 2D shape registration. A brief overview of TPS is given in Appendix B. A pseudo-code for the TPS alignment procedure is given below, for two spectral embeddings $\hat{A}$ and $\hat{B}$.

(1) Initialize parameters d, w, λ.
(2) Transform $\hat{B}$ into $\hat{\hat{B}}$ using the transformation parameters d and w.
(3) Update correspondence Y using Equation (3) after replacing $\hat{B}_j$ with $\hat{\hat{B}}_j$.
(4) Given Y, update transformation parameters using Equation (5).
(5) Update the regularization parameter λ.
(6) Repeat from Step 2 until convergence.

We have found experimentally that 5 to 10 iterations of the iterative alignment are sufficient to align the embeddings. The value of the regularization parameter λ is set to be the mean distance between all embedded point pairs. As shown in [3], this scale-dependent assignment of λ is robust to scaling of the point sets.

5.4. *Proximity-aided matching using anchor points*

For dense correspondence it is hard to distinguish between near-by points using an alignment and correspondence procedure based on optimizing a global energy. In order to improve correspondence locally, we perform matching using a heuristic based on point proximity. Specifically, we first select a small number of *anchor point* pairs. These are point pairs that are best matched (that is, pairs contributing the least to the correspondence cost), but that are also not too close to each other. Now

for finding the correspondence cost between two points, we not only consider the L_2 distance between their (non-rigidly aligned) spectral embeddings, but also the difference between their approximate geodesic proximities to these anchor points.

The anchor point pairs are computed as follows. Consider the $n_A \times n_B$ matrix Z of correspondence costs between all points of M_A and M_B. That is, $Z_{ij} = \|\hat{A}_i - \hat{B}_j\|$. The first anchor point pair $(a_A^{(1)}, a_B^{(1)})$, where $a_A^{(1)}$ is a vertex of M_A and $a_B^{(1)}$ is a vertex of M_B, is selected as the pair with least correspondence cost in Z. The second anchor point pair is calculated in the same way. However, we would need the anchor points to be far from each other over the mesh. Hence, before finding the second pair, we modify the matrix Z so that points close to the first anchor point are penalized. The new correspondence cost matrix is given by:

$$Z_{ij}^{(1)} = Z_{ij} - \frac{1}{2}[dist^{M_A}(i, a_A^{(1)}) + dist^{M_B}(j, a_B^{(1)})],$$

where $dist^M(i, j)$ is the approximate geodesic distance between the i^{th} and j^{th} vertices of mesh M. Now, the second anchor point pair is given by the least-cost pair according to $Z^{(1)}$. This process can be repeated to obtain more anchor point pairs. With the anchor points, we modify Equation (3) for finding the best correspondence Y to incorporate the proximity cost:

$$C(i) = \text{argmin}_j \Big[\|\hat{A}_i - \hat{B}_j\| \sum_{l=1}^{h} \alpha_l \cdot \|dist^{M_A}(i, a_1^{(l)}) - dist^{M_B}(j, a_2^{(l)})\| \Big]$$

where h is the number of anchor pairs and the α_l's are user-set parameters.

The success of the proximity heuristic depends on two factors: quality of the anchor point pairs and proximity measures. Since the meshes are already well aligned, choosing the anchor point pairs to be the most trusted matches are expected to be robust. However, the dependence on geodesic distances may cause sensitivity of the heuristic to stretching in the shapes. Hence, fixing a large number of anchor point pairs can render the matching non-robust. Thus we restrict to fixing only three anchor pairs and set $\alpha_1 = \alpha_2 = \alpha_3 = 1$ for all our experiments.

6. Experimental results

The meshes tested in our experiments are taken from the Princeton Benchmark data set[41] and their vertex counts range from 180 to 250. To evaluate a correspondence algorithm, we hand-pick a small number (between 15 and 20) of feature points on both meshes to be matched. The feature points would cover the shape fairly uniformly, encompassing all visually meaningful parts, e.g., see Figure 7 for feature points picked on two turtle models, a pig, and a rabbit. The ground-truth correspondence between the features is determined by human. Now we compute a correspondence using an algorithm and record the percentage of correct correspondences obtained. Before discussing experimental results, we first describe briefly the other schemes we have experimented with and compared.

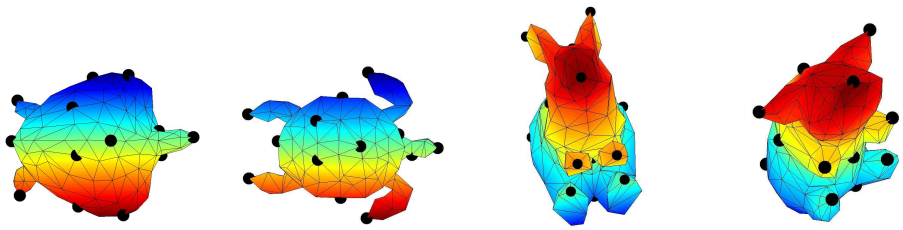


Fig. 7. Feature points hand-picked on four models are shown in dark dots.

- **TPS-RPM[7] in spatial domain:** one of the most successful non-rigid ICP algorithms, operating on point sets without connectivity information. It combines TPS, soft assign, and deterministic annealing to achieve robustness. But as we have shown in Figure 1, it is susceptible to poor initial alignment. To improve its performance, we *manually* and rigidly align the two shapes to be matched to neutralize any rotation or translation between them; this is done for all the three spatial-domain schemes we have experimented with.
- **Robust ICP[47] in spatial domain:** a recent variant of the original ICP[4] algorithm. It uses a hierarchical approach to achieve robust registration of 3D point sets. We use it as a representative of the rigid iterative alignment schemes.
- **Shape context[3] in spatial domain:** a trivial 3D extension of the original 2D shape context of Belongie et al. [3] is adopted as a representative correspondence scheme based on local shape descriptors. For each mesh vertex v, consider a bounding sphere of the mesh centered at v. The sphere is divided into many bins based on distance to the center, longitude, and latitude divisions. A histogram capturing the vertex count for each bin is the 3D shape context assigned to v. Rotation-invariance is not sought as we manually align the shapes before comparing the shape contexts at their vertices.
- **Shapiro and Brady[40]:** one of the early and best-known spectral point correspondence algorithms. It uses L_2 cost to compute a best matching, with their greedy sign correction but no eigenvector scaling or reordering. We also experiment with adding only eigenvector scaling to the Shapiro and Brady scheme.

6.1. *Comparison of correspondence results*

In Figure 8, we compare our algorithm against schemes listed above with respect to percentages of correct correspondences, on 10 test model pairs. The shapes to be matched are shown in Figure 9; each pair of shapes exhibit some degree of non-rigid deformations. In each case, $k = 6$ eigenvectors are computed. More eigenvectors do not change the result due to eigenvector scaling. For Nyström method, 20 samples are chosen via farthest point sampling. Four out of the ten cases have eigenmode switchings before the 6-th eigenvectors; they are armadillo:alien (fifth and sixth), hands4:hand3 (fourth and fifth), lion:horse (fifth and sixth), and rabbit:pig (third

and fourth). More switchings occur later, e.g., on the two turtle models, but they do not influence our results for the current test. Sign flips are rather common throughout the test cases. The greedy heuristic for eigenmode reordering and sign correction is successful in seven of the ten cases. The results shown are obtained by exhaustive search. Hence all results are limited to meshes with a few hundred vertices. In all test cases, no more than 10 iterations of our non-rigid ICP procedure are needed. In several cases, the procedure converges in less than 5 iterations.

In terms of results, we can see from Figure 8 that our non-rigid spectral correspondence algorithms clearly outperform all the schemes mentioned above. Subsampling and Nyström approximation has give the second best performance; further improvements should be possible if we rely on a more shape-sensitive sampling routine. Compared to the best performing scheme, which makes use of full affinity matrices, Nyström improves efficiency by an order of magnitude. Although the spatial algorithms (TPS-RPM, Shape Context, and Robust ICP) have been aided with manual initial alignment, bending in the shapes still cause them to perform poorly, e.g., see armadillo:alien and man1:man2. TPS-RPM performs poorly on the airplane models since they are badly tessellated with highly nonuniform point distributions. Without connectivity information, TPS-RPM also fails on hand1:hand2 as the fingers in hand2 are positioned too closely.

As expected, Shapiro and Brady performs poorly on meshes of different sizes, e.g., hand3:hand2 and hand4:hand3. It also scores badly on rabbit:pig and armadillo:alien, due to eigenmode switchings (in the case of rabbit:pig, an early switching appears to worsen the result more), and on turtle1:turtle2 and lion:horse, due to inadequate sign corrections. The case of the two turtles is chosen deliberately to test our algorithms, as the spatial algorithms are expected to work well. Our result confirms this, as both Robust ICP and TPS-RPM have returned perfect matchings. Our algorithms and shape context are one correct matching short of being perfect. Nevertheless, the one mis-matched point is assigned by our algorithms to a marked feature in close proximity to the ground-truth feature, leading to a correspondence that is not 1-to-1. Finally, note that our algorithms depend on distance measures over the mesh surfaces. Although we are using the crude graph distances to approximate geodesic distances, the correspondence results returned are quite satisfactory. If all the models were manifolds, then using the true geodesic distances should only improve the results further.

In Figure 9, we show some matching results obtained from our algorithm (full affinities and exhaustive search) visually. The matching is shown by coloring the vertices of the meshes in an appropriate way. We first assign colors to the vertices of one of the two meshes, e.g., M_B. Then the color for the i^{th} vertex of mesh M_A is set to be the color of the $Y(i)^{th}$ vertex of M_B, where Y is the correspondence found by our algorithm. This way, a good correspondence will induce a coloring that is consistent on both meshes. To show the meaningfulness of the correspondence obtained, we carefully assign different colors to different visual parts of the mesh

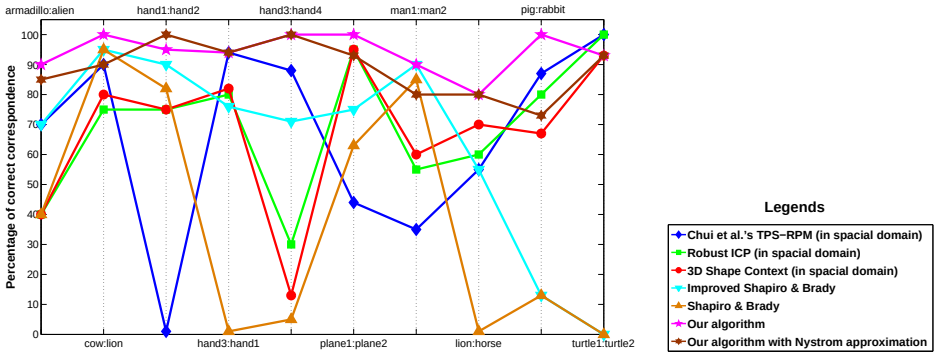


Fig. 8. A comparison between several correspondence algorithms, including ours. The percentage of correct correspondences is plotted.

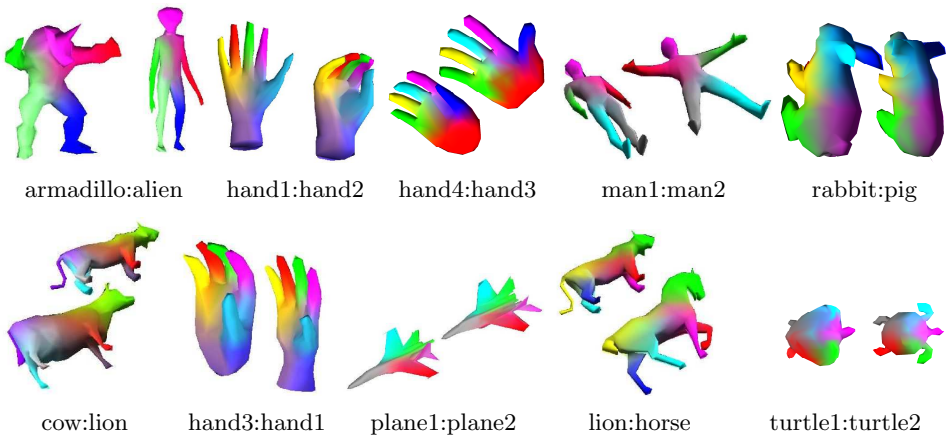


Fig. 9. Visual results for the correspondences obtained from our algorithm (full affinities and exhaustive search), shown with color plots. The model sizes (vertex count) are as follows. Armadillo: 256; hand3: 252; cow: 203; and all the rest: 182.

M_B. As can be seen, our algorithm matches bent shapes well, as well, it behaves robustly against moderate stretching in the shapes, e.g., see the armadillo:alien, lion:horse, cow:lion, hand3:hand1, and turtle1:turtle2 pairs.

However we should note that in the man1:man2 and lion:horse pairs, in Figure 9, which are of symmetric shapes, the correspondence is symmetrically switched. Namely, the right hand of one human is matched to the left hand of the other and vice versa. Similarly, the right legs of the lion are matched to the lefts leg of the horse and vice versa. In our evaluation of correspondence results, we tolerate such symmetry flips. They occur since we define affinities based purely on intrinsic measures. As such, the left hand and the right hand of the human are equally good matches for the right hand of the other human, as long as a consistence is

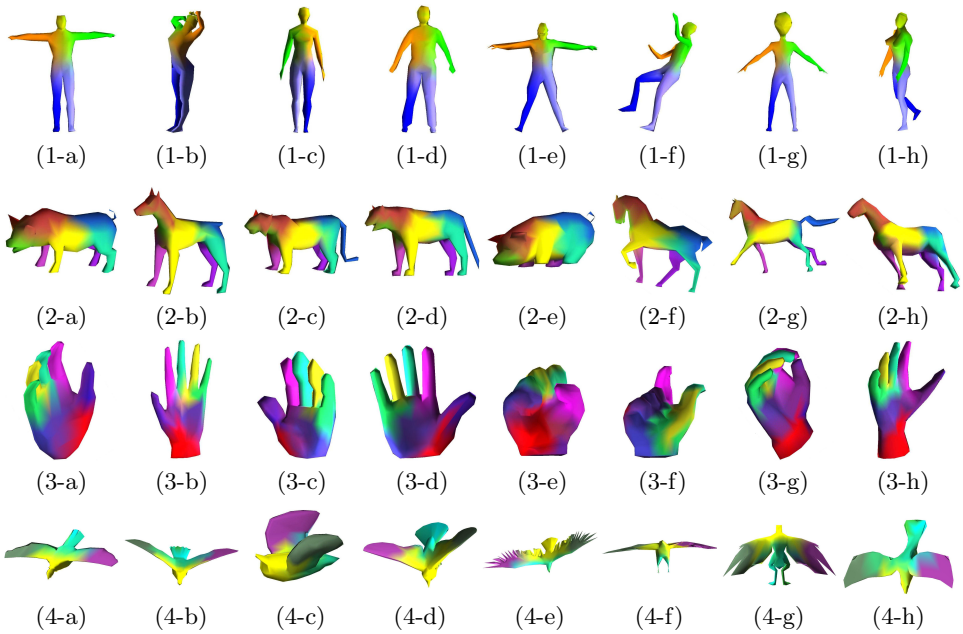


Fig. 10. Row 1: correspondence results for human shapes. Row 2: for animal shapes. Row 3: for hand shapes. Row 4: for bird shapes.

maintained. An interesting consequence is that our algorithm can find a meaningful matching between a left hand and a right hand, e.g., see Figures 10(3-a) and (3-c), even when both are close to making a fist. After all, each point on a hand is only aware of its geodesic distances to the remaining points, along various geodesic paths, it does not maintain an *ordering* between these paths. Note that a rigid algorithm will unlikely be able to match up the fingers on these left and right hands correctly.

Figure 10 gives additional correspondence results obtained using our algorithm on numerous articulated shapes. In each shape class, one per row, all the shapes are matched to a single reference shape (the first shape in the row of 8) and correspondence obtained is color coded in accordance with the colors on the reference shape. Apart from showing the effectiveness of our method, e.g., shown in the second row, these examples also reveal some of its limitations, discussed below.

6.2. *Current limitations*

Effect of using purely intrinsic information: As explained earlier, due to the intrinsic nature of our approach, it is not guaranteed to match symmetric shapes correctly, as shown in Figures 10(1-c), 10(4-c), and (4-d). In all three cases, the sign configuration of the eigenvectors that gives the lowest correspondence cost leads to counterintuitive results. Our method succeeds in all the remaining cases in row 1, 2, and 4, although in each case, the next best eigenvector sign configuration, which

has a correspondence cost extremely close to the lowest cost, would have given a symmetrically flipped matching. Note that although the correspondence returned by our algorithm may be symmetrically flipped, it is nevertheless still consistent across the shape.

One solution to the symmetry problem would be to carefully select the right signs of the eigenvectors within a small threshold. For shapes where there is one plane of symmetry, there will be two possible sign configurations of the eigenvectors that would give the same minimum correspondence cost. These can be detected and the right configuration may be chosen by utilizing additional information, e.g., from the spatial domain. As the number of symmetry planes increase, more sign configurations will need to be examined. An alternative would be to define affinities in a symmetry-distinguishing way.

Effect of topological changes: Since our method largely depends on geodesic distances, topological changes can seriously harm the correspondence computation. This effect is visible in Figure 10(3-e) and (3-f), where the fingers of the hands are connected to the palm which would change the connectivity of the mesh, as well as the geodesic distances drastically, resulting in unnatural correspondence results. Correct recovery of the correspondence between the fingers in this case appears to be a rather difficult problem, without some level of prior knowledge.

Unreliable approximate geodesic distances: Figure 10(4-e) shows a bird shape that is quite similar to the reference figure for this group, Figure 10(4-a). However, the correspondence obtained is incorrect. We suspect that this is mainly due to the unreliability of the geodesic distance approximation on the wings of the bird that contains many "cuts". Hence, even though the shapes look similar in the spatial domain, their spectral embeddings are rather different.

Non-robustness of L_2 cost for exhaustive search: Close inspection of Figure 10(1-b) reveals that the correspondence obtained is inconsistent: the left arm is colored orange which means that the left leg must be colored blue which is not the case (note that this is different from the symmetry issue discussed above). This should not have been the case as the shapes are topologically sound and the geodesic distances are approximated robustly. The problem becomes clear when we examine the result of the exhaustive reordering of eigenvectors. It turns out that for this shape, the exhaustive reordering does not give the right ordering of the eigenvectors, as shown in Figure 11. After further investigation we have found that the problem lies with the crude L_2 cost measure used in arriving at the reordering. A more meaningful cost measure should be sought.

7. Conclusion and future work

In this paper, we present a hybrid approach to finding a one-to-one correspondence between the vertices of two 3D meshes. We first transform the meshes into the spectral domain, based on geodesic affinities, and then match the spectral embeddings

Fig. 11. Incorrect eigenvector ordering is obtained even after exhaustively reordering the eigenvectors for shapes in Figure 10(1-a) and (1-b).

after taking appropriate steps to ensure a consistent ordering and sign assignment of the eigenvectors. Eigenvalue scaling of the eigenvectors renders our algorithm robust against difference in mesh sizes and choice of the dimensionality of the embeddings. Our method does not need a pre-selected set of feature vertices and can be completely automated. It is invariant against rigid transformations, uniform scaling, and shape bending. Experimentally, we find it to be robust against moderate stretching in the shapes as well, relying on thin-plate splines for non-rigid alignment in the spectral domain, and it outperforms well-known existing shape correspondence schemes.

A simple way to reduce the cost of extracting correspondences, where the naïve best matching would require quadratic time, would be to take advantage of the accurate alignments that have already been obtained and apply spatial partitioning to speed up the search for correspondence pairs. A more serious limitation of our current approach, in terms of computational cost, is its reliance on an exhaustive search to find a consistent eigenmode ordering and sign assignment. The greedy reordering approach is fast but it does not always give correct results. Analytically, the problem of finding a reordering and sign assignment which would lead to the best correspondence, e.g., according to the simple L_2 distance, is as hard as the graph isomorphism problem. We would like to look into fast approximation algorithms for this problem and adopt it for our purpose. Alternatively, we may be able to heuristically reorder the eigenvectors based on their corresponding nodal domains[10] and the shape characteristics of these nodal domains.

Quality-wise, an important issue is related to the appropriateness of the correspondence cost used in determining the eigenmode ordering and sign assignment, as well as in TPS alignment. Currently, we are using the crude L_2 cost, as it is simple to optimize for, but in some rare cases as shown earlier, even the exhaustive search could return a poor eigenvector ordering or sign assignment. This shows that a better cost function is still required. In addition, we plan to address the other limitations of our current method, including the handling of symmetry. We would also like to investigate possible definitions of the point affinities that are robust, if not invariant, to stretching within perceptually salient parts of a shape. This would offer an alternative to using non-rigid alignment in the spectral domain and avoid having to find a consistent eigenvector ordering or sign assignment.

Another interesting direction would be to compare our algorithms with skeleton-based methods for handling shape bending and stretching. This latter type of approaches have been extensively studied by Shokoufandeh and co-authors over the

years for classification and retrieval of articulated shapes, e.g., see[42]. A spectral approach is also used, focusing on eigenvalues, but the graph is skeletal, which is different from our case. It would be interesting to see how the skeletons can be used effectively for point correspondence. However, issues such as eigenmode switchings and symmetry will likely still need to be addressed.

Finally, we speculate that the different approaches for correspondence, e.g., intrinsic vs. extrinsic, skeleton vs. surface, rigid vs. non-rigid, may complement each other. For example, intrinsic approaches such as ours can lose the spatial perspective possessed by extrinsic methods, e.g., as in handling of shape symmetries, while it does excel in dealing with shape bending. Therefore, it may be possible to combine the different approaches to achieve more robust correspondence results.

Acknowledgments: This research is supported in part by an NSERC Discovery Grant (No. 611370) and an MITACS research grant (No. 699127).

References

1. H. A. Almohamad, "A Polynomial Transform for Matching Pairs of Weighted Graphs," *Appl. Math. Modeling,* 15:216-222, 1990.
2. M. Alexa, "Recent Advances in Mesh Morphing," *Computer Graphics Forum,* 21(2):173-196, 2002.
3. S. Belongie, J. Malik, and J. Puzicha, "Shape Matching and Object Recognition Using Shape Contexts," *IEEE Trans. on PAMI,* 24(24):509-523, 2002.
4. P. J. Besl and N. D. Mckay, "A Method for Registration of 3-D Shapes," *IEEE Trans. on PAMI,* 14(2):239-256, 1992.
5. F. L. Bookstein, "Principal Warps: Thin-Plate Splines and the Decomposition of Deformations," *IEEE Trans. on PAMI*, 11(6):567-585, 1989.
6. T. Caelli and S. Kosinov, "An Eigenspace Projection Clustering Method for Inexact Graph Matching," *IEEE Trans. on PAMI,* 26(4):515-519, 2004.
7. H. Chui and A. Rangarajan, "A new point matching algorithm for non-rigid registration," *Computer Vision and Image Understanding,* 89:114-141, 2003
8. M. Carcassoni and E. R. Hancock, "Spectral Correspondence for Point Pattern Matching," *Pattern Recognition,* 36:193-204, 2003.
9. M. Carcassoni and E. R. Hancock, "Correspondence Matching with Modal Clusters," *IEEE Trans. on PAMI,* 25(12):1609-1615, 2003.
10. S. Dong,P-T. Bremer, M. Garland, V. Pascucci, and J. C. Hart, "Spectral Surface Quadrangulation," *ACM Trans. on Graph., 2006*.
11. C. Eckart and G. Young, "The Approximation of One Matrix by Another of Lower Rank," *Psychometrika,* 1:211-218, 1936.
12. A. Elad and R. Kimmel, "On Bending Invariant Signatures for Surfaces," *IEEE Trans. on PAMI,* 25(10):1285-1295, 2003.
13. C. Fowlkes, S. Belongie, F. Chung, and J. Malik, "Spectral Grouping Using the Nyström Method," *IEEE Trans. on PAMI,* 26(2):214-225, 2004.
14. A. Frome, D. Huber, R. Kolluri, T. Bulow, and J. Malik, "Recognizing Objects in Range Data Using Regional Point Descriptors," *European Conference on Computer Vision (ECCV),* 2004.
15. M. Garland, *QSlim*: http://graphics.cs.uiuc.edu/˜garland/software/qslim.html.

16. T. Gatzke, C. Grimm, M. Garland and S. Zelinka, "Curvature Maps for Local Shape Comparison," in *Proc. of Shape Modeling International*, 2005.
17. N. Gelfand, N. Mitra, L. Guibas, H. Pottmann, "Robust Global Registration," *Proc. of Symposium of Geometry Processing,* 2005.
18. G. H. Golub and C. F. Van Loan, *Matrix Computations,* John Hopkins University Press, 1996.
19. C. Gotsman, X. Gu, and A. Sheffer, "Fundamentals of Spherical Parameterization for 3D meshes," *ACM Transaction on Graphics,* 22(3):358-363, 2003.
20. V. Jain, *Robust Correspondence and Retrieval of Articulated Shapes,* M. Sc. Thesis, School of Computing Science, Simon Fraser University, June 2006.
21. V. Jain and H. Zhang, "Shape-Based Retrieval of Articulated 3D Models Using Spectral Embeddings," in *Proc. of Geometric Modeling and Processing 2006*, pp. 295-308.
22. A. Johnson and M. Hebert, "Using Spin-Images for Efficient Multiple Model Recognition in Cluttered 3-D Scenes," *IEEE Trans. on PAMI*, pp. 433-449, 1999.
23. M. Körtgen, G-J. Park, M. Novotni, R. Klein, "3D Shape Matching with 3D Shape Contexts," *Seventh Central European Seminar on Computer Graphics,* 2003.
24. R. Kolluri, J. R. Shewchuk, and J. F. O'Brien, "Spectral surface reconstruction from noisy point clouds," *Proc. of the 2004 Eurographics/ACM SIGGRAPH symposium on Geometry processing,* pp. 11-21, 2004.
25. V. Kraevoy, A. Sheffer, and C. Gotsman, "Matchmaker: Constructing Constrained Texture Maps," *ACM SIGGRAPH,* pp. 326-333, 2003.
26. V. Kraevoy and A. Sheffer, "Cross-Parameterization and Compatible Remeshing of 3D Models," *ACM SIGGRAPH,* 2004.
27. R. Liu and H. Zhang, "Segmentation of 3D Meshes through Spectral Clustering," *Proc. of Pacific Graphics,* pp. 298-305, 2004.
28. X. Li, I. Guskov, "Multiscale Features for Approximate Alignment of Point-based Surfaces," in *Proc. of Symposium on Geometry Processing,* 2005.
29. R. Liu, V. Jain, and H. Zhang, "Subsampling for Efficient Spectral Mesh Processing," in *Proc. of Computer Graphics International,* Lecture Notes in Computer Science 4035, H.-P. Seidel, T. Nishita, and Q. Peng, Eds., pp. 172-184, 2006.
30. *McGill 3D Shape Benchmark*: http://www.cim.mcgill.ca/˜shape/benchMark/.
31. K. Mikolajczyk and C. Schimid, "A Performance Evaluation of Local Descriptors," *IEEE Trans. on PAMI,* 27(10):1615-1630, 2005.
32. A. Y. Ng, M. I. Jordan, Y. Weiss, "On Spectral Clustering: Analysis and An Algorithm," in *NIPS 14,*, pp. 857-864, 2002.
33. P. Pardalos, F. Rendl, and H. Wolkowicz, "The Quadratic Assignment Problem: A Survey and Recent Developments," in *Quadratic assignment and related problems (New Brunswick, NJ, 1993),* Amer. Math. Soc., pp. 1-42, 1994.
34. E. Praun, W. Sweldens, and P. Schröder, "Consistent Mesh Parameterizations," *ACM SIGGRAPH,* pp. 179-184, 2001.
35. S. Rusinkiewicz and M. Levoy, "Efficient Variants of the ICP Algorithm," In *Proc. 3rd Int. Conf. on 3-D Digital Imaging and Modeling,* pp. 145-152, 2001.
36. J. Schreiner, A. Asirvatham, E. Praun, and H. Hoppe, "Inter-Surface mapping," *ACM Trans. Graph.,*, 23(3):870-877, 2004.
37. S. Sclaroff and A. Pentland, "Model Matching for Correspondence and Recognition," *IEEE Trans. PAMI,* 17(6):545-561, 1995.
38. G. Scott and H. Longuet-Higgins, "An Algorithm for Associating the Features of Two Patterns," *Proc. Royal Soc. London,* Vol. B244, 1991.
39. C. Scott and R. Nowak, "Robust Contour Matching via the Order Preserving Assignment Problem," *IEEE Trans. on Image Processing,* 15(7):1831-1838, July 2006.

40. L. S. Shapiro and J. M. Brady, "Feature Based Correspondence: An Eigenvector Approach," *Image and Vision Computing,* 10(5):283-288, 1992.
41. P. Shilane, P. Min, M. Kazhdan, and T. Funkhouser, "The Princeton shape benchmark," *Proc. of Shape Modelling International,* pp. 167-178, 2004.
42. A. Shokoufandeh, D. Macrini, S. Dickinson, K. Siddiqi, and S. Zucker, "Indexing Hierarchical Structures using Graph Spectra," *IEEE Trans. on PAMI, Special Issue on Syntactic and Structural Pattern Recognition,* 27(7):1125-1140, 2005.
43. R. Sumner and J. Popovic, "Deformation Transfer for Triangle Meshes," *ACM Transactions on Graphics,* Vol. 23(3), 2004.
44. S. Ullman, *The Interpretation of Visual Motion,* The MIT Press, 1979.
45. S. Umeyama, "An Eigen Decomposition Approach to Weighted Graph Matching Problems," *IEEE Trans. on PAMI,* 10:695-703, 1988.
46. R. Zayer, C. Rössl, Z. Karni and H.-P. Seidel, "Harmonic Guidance for Surface Deformation," *Computer Graphics Forum (Proc. of Eurographics),* 24(3):601-609, 2005.
47. T. Zinber, J. Schmidt and H. Niemann, "A Refined ICP Algorithm For Robust 3D Correspondence Estimation," in *Proc. of Int. Conf. on Image Processing*, 2003.

Appendix A: Nyström Approximation

Consider a set of n points $\mathcal{Z} = \mathcal{X} \bigcup \mathcal{Y}$ partitioned by sets $\mathcal{X}$ and $\mathcal{Y}$ of sizes p and q, respectively. Write the symmetric affinity matrix $W \in \mathbf{R}^{n\times n}$ for $\mathcal{Z}$ in block form:

$$W = \begin{bmatrix} A & B \\ B^T & C \end{bmatrix},$$

where $A \in \mathbf{R}^{p\times p}$ and $C \in \mathbf{R}^{q\times q}$ are affinity matrices for points in $\mathcal{X}$ and $\mathcal{Y}$, respectively; $B \in \mathbf{R}^{p\times q}$ contains the cross-affinities between points in $\mathcal{X}$ and $\mathcal{Y}$. Without loss of generality, we designate the points in $\mathcal{X}$ as *sample points*. Let $A = U\Lambda U^T$ be the eigenvalue decomposition of A, then the eigenvectors of W can be approximated, using the Nyström method[13], as

$$\bar{U} = \begin{bmatrix} U \\ B^T U \Lambda^{-1} \end{bmatrix}.$$

This allows us to approximate the eigenvectors of W by only knowing the sampled sub-block $[A \;\; B]$. The overall complexity is thus reduced from $O(n^3)$, without subsampling, down to $O(pn \log n + p^3 + p^2 n)$, where $p \ll n$ in practice.

The rows of $\bar{U}$ define the spectral embeddings of points in $\mathcal{Z}$. We see that the i^{th} row of U, which is completely determined by A, gives the embedding $\bar{x}_i$ of point x_i in $\mathcal{X}$ and the j^{th} row of $B^T U \Lambda^{-1}$ is the embedding $\bar{y}_j$ of point y_j in $\mathcal{Y}$. If we let $\bar{y}_j^d$ denote the d^{th} component of $\bar{y}_j$, then the above equation can be rewritten as

$$\bar{y}_j^d = \frac{1}{\lambda_d} \sum_{i=1}^{p} \bar{x}_i^d B(i,j) = \frac{1}{\lambda_d} \sum_{i=1}^{p} \bar{x}_i^d W(i, j+p), \qquad 1 \le d \le p.$$

Namely, the embedding $\bar{y}_j$ is extrapolated using the coordinates of the $\bar{x}_i$'s, weighted by the corresponding cross-affinities in B. With $\bar{U}$, we obtain an approximation $\bar{W}$ of the original affinity matrix W,

$$\bar{W} = \bar{U}\Lambda\bar{U}^T = \begin{bmatrix} A & B \\ B^T & B^T A^{-1} B \end{bmatrix}.$$

Appendix B: Thin-plate splines

Thin plate splines[5] are a generalization of cubic splines to higher dimensions and it contains affine transformation as a special case. With non-rigid transformations, there are infinitely many ways of transforming a point set into another. Thin plate splines are effective because of their smoothness constraints which discourage arbitrary mappings. In the limit of this smoothness constraint the thin plate spline model reduces to an affine transformation model. The thin plate spline transformation functions $f_p : x \in \mathbf{R}^k \rightarrow \mathbf{R}, 1 \leq p \leq k$ map a point set $X = \{x_1, x_2, \ldots, x_n\}$ in k (say $k = 2$) dimensional space to another point set $Y = \{y_1, y_2, \ldots, y_n\}$ by minimizing the following energy functionals:

$$E(f_p) = \sum_{i=1}^{n} \|y_{ip} - f_p(x_i)\| + \lambda \int\int \left[(\frac{\partial^2 f_p}{\partial x^2})^2 + 2(\frac{\partial^2 f_p}{\partial x \partial y})^2 + (\frac{\partial^2 f_p}{\partial y^2})^2 \right] dxdy \quad (4)$$

where λ is the regularization (smoothing) parameter. Note that the correspondence between X and Y is assumed to be given. Hence, point $y_i = (y_{i1}, y_{i2}, \ldots, y_{ik})$ is the matching point for x_i. The unique set of f_p's that minimize the above energy functionals can be written in matrix form as:

$$f(x_i, d, w) = x_i \cdot d + \phi(x_i) \cdot w,$$

where x_i is now in $(k+1)$-dimensional homogeneous coordinates, d is a $(k+1) \times (k+1)$ affine transformation matrix, w is an $n \times (k+1)$ warping coefficients matrix and $\Phi(x_i)$ is a vector of length n such that $\phi_j(x_i) = -\|x_j - x_i\|$.

As shown in[3], the transformation (d, w) that minimizes the energy can be calculated by solving the following system:

$$\begin{bmatrix} K & X^T \\ X & 0 \end{bmatrix} \begin{bmatrix} w \\ d \end{bmatrix} = \begin{bmatrix} Y \\ 0 \end{bmatrix} \quad (5)$$

Here, K is the matrix $(\Phi - \lambda I)$ where, I is an identity matrix of appropriate size and Φ is an $(n \times n)$ matrix whose i^{th} row is $\phi(x_i)$, that is, $\Phi_{ij} = -\|x_j - x_i\|$.

Using these transformation parameters, we transform the point set X to point set Y and then recompute the correspondence. This process is iterated until convergence as described in Section 5.