Embedding Retrieval of Articulated Geometry Models

Gary K.L. Tam, *Member, IEEE,* and Rynson W.H. Lau, *Senior Member, IEEE*

Abstract—Due to the popularity of computer games and animation, research on 3D articulated geometry model retrieval is attracting a lot of attention in recent years. However, most existing works extract high dimensional features to represent models and suffer from practical limitations. First, misalignment in high dimensional features may produce unreliable Euclidean distances and affect retrieval accuracy. Second, the curse of dimensionality also degrades efficiency. In this paper, we propose an embedding retrieval framework to improve the practicability of these methods. It is based on a manifold learning technique, the Diffusion Map (DM). We project all pairwise distances onto a low dimensional space. This improves retrieval accuracy because inter-cluster distances are exaggerated. Then we adapt the Density-Weighted Nyström extension and further propose a novel step to locally align the Nyström embedding to the eigensolver embedding so as to reduce extension error and preserve retrieval accuracy. Finally, we propose a heuristic to handle disconnected manifolds by augmenting the kernel matrix with multiple similarity measures and shortcut edges, and further discuss the choice of DM parameters. We have incorporated two existing matching algorithms for testing. Our experimental results show improvement in precision at high recalls and in speed. Our work provides a robust retrieval framework for the matching of multimedia data that lie on manifolds.

Index Terms—geometry retrieval, articulated model retrieval, geometry analysis, geometry recognition.

1 Introduction

3D geometry models are essential components in many of the latest multimedia applications, including 3D games, animation movies, virtual environments and object recognition. With the increasing number of geometry models available on the Web, many geometry model retrieval methods have been proposed to facilitate searching and resue of these models. However, most of these methods can only handle exact match, i.e., the retrieved models need to be very similar in shape and pose to the input model. This seriously affects the retrieval performance, as these methods consider models of similar shape but different poses as different models. To address this limitation, recent research focus is moving towards the development of geometry retrieval techniques that consider the similarity of model shapes only and tolerate different poses. However, tolerating different poses turns out to be much more challenging than simply performing an exact match. A number of methods have been proposed for this purpose. The main idea is to extract deformation invariant features based on some kind of metric measures on the surface. In this paper, we refer to models of similar shape but different poses as *articulated geometry models*, models of different objects but similar skeleton (e.g., wolf and dog) as *similar-skeleton models*, and models of dissimilar skeletons as *dissimilar-skeleton models*.

Recent development of retrieval techniques focus on defining distinctive geometric features [1], [2]. The vast amount of features (over many hundreds in dimension) have created two problems. First, in high dimensional spaces, feature misalignment produces large intra-cluster distances, leading to poor retrieval performance if intra-cluster distances are greater than inter-cluster distances. Second, the curse of dimensionality results in large storage space and low retrieval efficiency. There are relatively few methods that discuss how to build a fast retrieval system.

Our idea begins from the observation that most articulated geometry models look somewhat similar. When we put these models together, they can easily be arranged to form an animation sequence. This suggests that there are only a limited number of parameters that govern shape and pose variations. If we can effectively reduce the high-dimensionality of these features, we may be able to improve retrieval accuracy and speed. Hence, we propose in this paper an unsupervised embedding retrieval framework for articulated geometry models to tackle the two problems discussed above. We first argue with evidence that existing methods project data on separate manifolds. We then apply a manifold learning technique, the Diffusion Map (DM) [3], for dimension reduction. The Diffusion Map differs from other manifold learning techniques in that it is highly effective for compressing data to arbitrary dimension [4]. By selecting appropriate DM parameters and defining a low output dimension, we avoid the curse of dimensionality and exaggerate the inter-cluster distances of model groups in the induced embedding space.

One major problem of using DM, however, is that it requires the use of Nyström extension to unknown queries. Although the quality of the extended embedding improves as the number of landmarks (i.e., the

Manuscript submitted on March 6, 2010, revised on March 31 and July 21, 2011 and accepted on Dec 28, 2011.

- *Gary K.L. Tam is with the Department of Computer Science, University of Durham, UK. E-mail: kltam327@gmail.com*
- *Rynson W.H. Lau is with Department of Computer Science, City University of Hong Kong, HK. E-mail: rynson@cs.cityu.edu.hk*

number of selected reference samples for computing Nyström extension) increases, increasing the number of landmarks hinders the retrieval speed. To overcome this problem, we have adapted the Density-Weighted Nyström extension [5] for DM. We then propose a novel step to locally align the Nyström embedding to the eigensolver embedding. This novel scheme is able to effectively reduce both projection and retrieval errors even when the number of landmarks are small. Finally, we propose a heuristic method to handle the situation when the data lies on separate manifolds.

We summarize our main contributions as follows:

1) To the best of our knowledge, this is the first comprehensive empirical study to explore the concept of applying manifold learning techniques on the *distance measure space* of articulated geometry models in the context of retrieval.
2) If the underlying similarity measures can separate data in different manifolds, we propose a heuristic approach to find retrieval parameters to improve retrieval accuracy. If these data are not separable in manifolds, we propose to augment the kernel matrix by combining different similarity measures together and introduce shortcut edges to improve the embedding distance and retrieval accuracy.
3) For the retrieval to be practical, we adapt the Density-Weighted Nyström extension for the Diffusion Map with a distribution separation step and align the Nyström embedding to the eigensolver embedding. The proposed retrieval framework produces comparatively good quality of embedding with fast online query speed.

The rest of this paper is organized as follows. Section 2 summarizes existing works on articulated geometry model retrieval and embedding retrieval. Section 3 justifies the use of manifold learning in our work. Section 4 summarizes Diffusion Maps. Section 5 presents our retrieval framework in details. Section 6 evaluates the performance of our framework. Finally, Section 7 briefly concludes this paper.

2 Related Work

2.1 Articulated Geometry Model Retrieval

Geometry model retrieval is a challenging research topic. The difficulty in matching and retrieving geometry models is due to the semantic gap between human perception and feature representation, which is always complicated by irregular shape, orientation, scale, and data format (e.g., triangle soup, point clouds and meshes). Many successful methods have been proposed for retrieving non-articulated geometry models [6], [7]. However, the analysis of articulated geometry models is still in its infancy. It faces not only the same challenges as for the non-articulated models, but also the deformation and articulation of shapes. In this section, we review the main works on articulated geometry model retrieval. We refer the reader to [8] for works on non-articulated one.

Articulated geometry model retrieval methods can be roughly classified into two types based on feature representation: a single feature vector and a bag of features. The first type uses a single feature vector, called shape descriptor, to represent the whole model. Most of these methods construct histograms based on metrics defined on the surface. Such metrics are deformation invariant, including eccentricity [9] and the part-aware metric [10]. Recently, the spectrum (leading eigenvalues) of the Laplace-Beltrami operator is also used as a descriptor [11], [12]. It corresponds to the significant components (structure) of a surface. [13] computes a feature vector by encoding a bag of visual features and converting them into a histogram. [14] computes the feature vector based on heat signature, while [15] further extracts spatial information into the feature vector. The second type uses a bag of scalars or vectors as features. Some methods partition a model based on a metric on the surface and arrange local features as graph structures. These local features include area and length [1], volume, cords and curvature [16], and spherical harmonic [17]. Graph matching techniques can then be used to match these graphs. Other methods use bag-based matching techniques like bipartite matching or Earth Mover Distance [18] to avoid graph matching. These methods include geodesic histogram [19], curvature, area and thickness histogram [2], and stretching histogram [20].

We have two observations. First, both types of methods are related to high dimensional Euclidean distances. It is trivial for the single vector type of methods because Euclidean distance is directly employed to define the similarity measure. In Section 3, we show that when comparing similar-skeleton models, graph or bag-based matching also becomes Euclidean distance. Euclidean distance, as explained in [21], is not reliable under slight misalignment. It leads to large intra-cluster variance in data, meaning that these methods may not handle similar-skeleton models well. Second, the practicability and scalability of these methods on large databases have not been explored. Most feature representations are high dimensional and suffer from the curse of dimensionality. In addition, the dissimilarity measures for graph or bag-based methods are usually non-metric. Hence, traditional indexing techniques cannot be applied. Some methods have been proposed to speed up the matching process. In [22], [23], the matching process is separated into two steps. The first step makes use of graph spectrum to encode structure for fast pruning. The second step applies a graph matching algorithm to select models based on geometric similarity. This scheme, however, still suffers if the database contains a lot of similar-skeleton models (e.g., dog, wolf, lion), as the first step cannot prune these models by structures, which leads to high computational cost in the second step. In [2], we have defined a metric measure and used a distance-based indexing technique for fast k-nearest neighbor search. This method works when k (the number of retrieved models) is small. When k is large (e.g., greater than 10), the indexing scheme soon approaches brute-force. These two observations lead us to consider ways to reduce dimensionality and to develop this embedding retrieval framework.

2.2 Embedding Retrieval

Applying manifold learning techniques to analyze 3D articulated geometry models is not new. For example, [11], [12] study the Laplace-Beltrami operator, and [14], [15] study the Diffusion Kernel. All these methods use manifold learning techniques to study the *geometry* and obtain a signature for each 3D model. The novelty of our work is that we investigate the application of manifold learning techniques to obtain a better embedding distance space, so as to improve retrieval accuracy. As far as we know, [13] is most relevant to our work. It uses distance learning techniques to adjust distances with manifold ranking. However, it requires the re-computation of nearest neighbors for all models in the database on every query. It is unclear if it is applicable to large databases.

There are many manifold learning techniques in the literature, such as Laplacian eigenmaps (LE) [24], ISOMAP [25], Locally Linear Embedding (LLE) [26], Locality Preserving Projections (LPP) [27] and Diffusion Maps (DM) [3]. The use of manifold learning techniques in other media has been successful. For example, LLE, ISOMAP, Multi-Dimensional Scaling [28], and LPP have been applied to various image retrieval works, e.g., image clustering [29] and relevancy feedback [27]. The success of these methods depends on the assumption that visual perception is better represented by nonlinear distance than its original distance. As will be shown in the next section, this assumption can also be applied to geometry model features as they also lie on manifolds.

In our context, we choose to apply the Diffusion Map here as it is able to handle the non-uniform distribution of data [3] and effective in reducing the dimension by diffusion, which ensures that the manifold is properly smoothed with important information retained. Hence, it differs from other techniques that simply discard useful higher dimensions [4]. Recently, DM has also been applied to image databases as a transductive relevancy feedback tool [30]. There are also supervised learning methods, referred to as Metric Learning, that exaggerate the distances among data so that the metric produces small distances for objects within the same category and large distances for those of different categories [31]. Our method, however, focuses more on dimension reduction.

In embedding retrieval literature, it is popular to use Nyström extension to approximate embedding of large datasets. General embedding retrieval algorithms like FastMap, MetricMap and Landmark MDS are all based on similar ideas [32]. It is well-known that projecting both the database and queries using a few landmark objects leads to large approximation errors and causes a degradation on the overall retrieval accuracy. This problem becomes more severe when more dimensions are required. To improve Nyström extension, [33] uses a matrix completion view to approximate full kernel matrix, and [5] incorporates density into the computation. We adapt ideas from these two works in our retrieval framework to improve the quality of the embedding.

3 Manifolds in Feature Space

3.1 Limitations of Existing Approaches

When similar-skeleton models are matched, the best way to tell them apart is by using geometric features as the skeletal and topological features are likely similar. A general idea is to use more geometric features for comparison. To study the performance of these methods, we have created a database of 1,020 articulated geometry models. In order for the database to contain models with large span of gestures, we output the geometry models every five frames of an animation sequence. It ensures that all models are different from each other and gives a fairer evaluation than simply rotating or scaling the models as in [2]. After obtaining this database, we compute and embed all distances using Multi-Dimensional Scaling (MDS) [28], which is a popular visualization tool for preserving all pairwise distances.

We have tried two methods on our database, (Multi-resolution Reeb Graph) MRG [1] and (Topological Point Ring) TPR [2]. They use very high numbers of geometric features (approximately 800 and 900, respectively). MRG builds a multi-resolution tree by partitioning a 3D mesh into multiple intervals. The distance measure is defined by matching all graph nodes from top to bottom between two trees in a greedy manner. TPR defines a bag of features – a set of points and rings, where points are protrusion tips and rings are segment boundaries. Each point or ring is associated with three histograms (curvature, area and thickness) to describe surface details. The histogram is constructed by partitioning the mesh using geodesic. It employs Earth Mover Distance (EMD) [18] as a dissimilarity measure. Both methods are reliable towards small noise; for MRG, the partition function is an integral and for TPR, it inherits the robustness from EMD.

Fig. 1 shows our experimental results. We have two observations here. First, these models form many nearby manifolds in the embedding space. Second, the intra-cluster variances among similar-skeleton models are all larger than the inter-cluster distances. To explain these, we may analyze the features used in

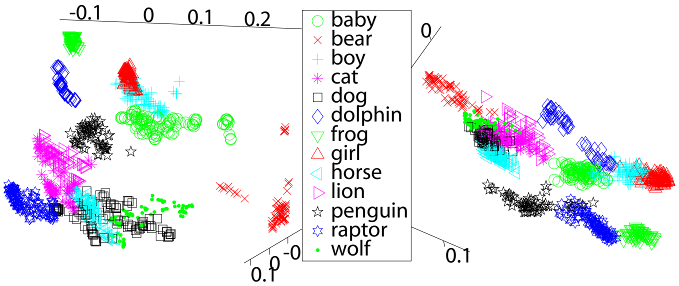


Fig. 1. MDS visualization of MRG (left), a Graph-based method, and TPR (right), a bag-based method. Similar-skeleton models may not form separable clusters.

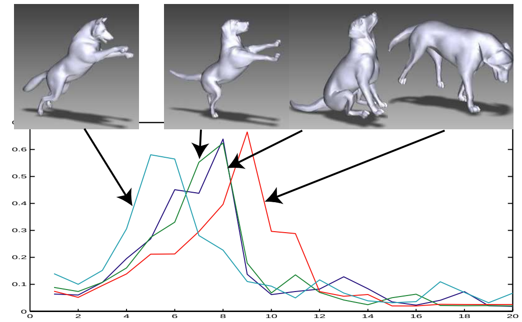


Fig. 2. Feature histograms of 4 different models (from left: wolf, dog, dog, dog).

these methods. Both MRG and TPR capture features that adapt to the underlying topology and hence deformation. When two similar-skeleton models are matched, the similarity measure is like high dimension Euclidean distance as graph matching and EMD are designed in a way to find feature correspondences of two models. Euclidean distance can be considered as a distance in a very high dimension (nm, with n being the number of nodes in the MRG tree or number of features in TPR, and m being the number of scalars in the MRG tree or number of bins in each TPR histogram).

3.2 Justifications for Manifold Learning

The main limitation of Euclidean distance is that it is very sensitive to even slight misalignment. As pointed out in [21] (Chapter 2), Euclidean distance is not a smooth function with respect to natural parameters (deformation in our concern). To illustrate this, we consider the four model signatures shown in Fig. 2. These histograms are obtained based on geodesic partitioning of a feature extracted from one of the legs of each animal. We see that the histograms of all dog models have a sharp peak while that of the wolf model has a round peak. The two dogs on the left are close to each other while the right dog (especially the peak) is slightly misaligned due to articulation change. We have also compared the Euclidean histograms of the four models. The distance between the left and the middle dogs is only 0.2271. The distance between the left and right dogs is 0.7285, while the distance between the left dog and the wolf is 0.5853. In other words, though the histogram shape of the right dog is similar to that of the left dog, the wolf has a smaller Euclidean distance instead. This shows that the misalignment of histograms may easily lead to a large intra-cluster variance. When the variance is greater than the inter-cluster distance, retrieval accuracy will be affected.

There is another justification for using manifold learning. It is the high dimensionality of the features itself. Each 3D model can be considered as a point in the high dimensional space. As the number of dimensions increases, these points spread on a surface (manifold structure) and any analysis will depend on the surrounding neighborhood (edge effect) of these points [34]. This corresponds well to the notion of manifold learning techniques. Therefore, though we validate the hypothesis on two methods (Multi-resolution Reeb Graph) MRG [1] and (Topological Point Ring) TPR [2], we believe that our study is equally applicable to many other methods for articulated geometry model retrieval.

Our idea of improving retrieval accuracy (Sections 5.5 and 5.6), in an unsupervised manner, is to find a new embedding such that the inter-cluster distances among different manifolds may be exaggerated. Manifold learning has been applied to some other media applications, including image segmentation [35] and mesh clustering [36]. The reason that these segmentation and clustering algorithms work is evidenced by the Polarization Theorem. As discussed in [37] (Theorem 5.6), the angles between eigenvectors become polarized when the projected dimensionality is reduced. In other words, when a data representation is projected onto leading eigenvectors (i.e., a low dimensional embedding space), the embedding distances among data items are exaggerated, which enhance the clustering structure of the data representation. Although our work is based on the Diffusion Map, we expect that other manifold learning techniques should be equally applicable.

4 The Diffusion Map (DM)

To summarize the Diffusion Map (DM) [21], suppose O is a set of n data points approximately lying along a submanifold Ω. (In our case, O is the set of all objects in the database.) When n approaches ∞, the random walk on this discrete graph of O converges to the random walk (diffusion process) on the continuous space Ω. In real applications, however, the number of data points is finite, and the DM provides a discrete approximation of the underlying diffusion process.

Let $x, y \in O$ be two models, and $W(x, y)$ be an entry of the pairwise distance matrix W, which is obtained by a graph-based / bag-based method. We compute a kernel $K_W(x, y) = \exp\left(-\frac{W^2(x,y)}{\sigma}\right)$, where σ is a parameter that defines the local scale of the neighborhood. The use of an exponential function suggests that small distances are more important than large ones. This is essential to learning manifolds as they are defined by local neighborhoods. Since these data may have different distributions, it is best to separate distribution from the geometry so that the embedding is not affected by local factors. [21] estimates the distribution by letting $p_W(x) = \sum_{y \in O} K_W(x, y)$. $K(x, y)$, which has the distribution separated from the geometry, is defined as:

$$K(x,y) = \frac{K_W(x,y)}{p_W(x)p_W(y)} \tag{1}$$

However, kernel $K(x, y)$ is not symmetric and the symmetric anisotropic transition kernel $P(x, y)$ of Markov chain on O is usually considered. Let $q(x) = \sum_{y \in O} K(x, y)$,

$$P(x,y) = \frac{K(x,y)}{\sqrt{q(x)}\sqrt{q(y)}} \tag{2}$$

The diffusion distance $D_t(x, y)$ of the embedding space is defined as:

$$D_t^2(x,y) = \sum_{u \in O} (P^t(x,u) - P^t(y,u))/\pi(u) \tag{3}$$

where $\pi(u) = q(u)/\sum_{z \in O} q(z)$ and P^t are the stationary distribution and the t^{th} time step of Markov chain, respectively. Let d be the dimension of the embedding space. The diffusion distance can be approximated as $D_t(x,y) = \left(\sum_i^d \lambda_i^{2t} \left(\psi_i(x) - \psi_i(y)\right)^2\right)^{\frac{1}{2}}$, where $\psi_{1..d}$ and $\lambda_{1..d}$ are the d eigenvectors and eigenvalues of $P(x, y)$. The DM $\Psi_t(x) : O \to \mathbb{R}^d$ embeds all geometry models into a Euclidean space:

$$\Psi_t(x) = \left(\lambda_1^t \psi_1(x), \lambda_2^t \psi_2(x), ..., \lambda_d^t \psi_d(x)\right)^T \tag{4}$$

In this space, inter-cluster distances among different manifolds are exaggerated. Since it is a Euclidean space, we can apply a spatial indexing method (e.g., the kd-tree) for fast retrieval. To extend DM to out-of-sample data, [21] uses Nyström Extension:

$$\hat{\psi}(x) = \frac{1}{n\lambda} \sum_{y \in O} P(x,y)\psi(y) \tag{5}$$

such that the diffusion coordinate $\hat{\psi}(x)$ of sample x can be extrapolated from coordinates $\psi(y)$ of all n models in O, weighted by P.

5 Our Retrieval Framework

Multimedia retrieval usually involves high dimensional features and large datasets containing thousands to millions of records. Manifold learning techniques can be used to find a subspace that preserves the distances among the manifolds of data. Given a continuous diffusion process, the discrete finite eigenfunctions of the diffusion operator usually provide a good approximation of the manifold. However, solving eigen-decomposition directly for online queries is not practical due to its high computational cost. Hence further approximation is usually sought.

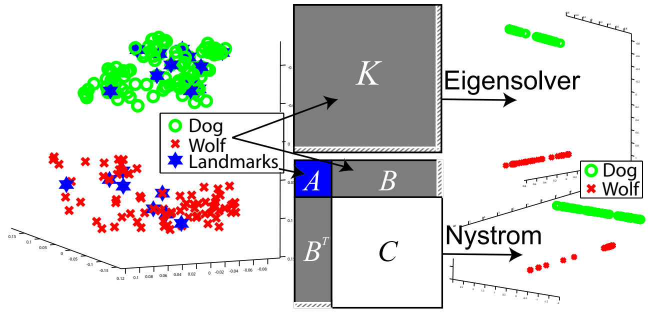


Fig. 3. Using eigensolver and Nyström Extension for retrieval. K is the kernel distance matrix. A, B and C, which are submatrices of K, are the distance matrices among landmarks, between landmarks and non-landmarks, and among non-landmarks, respectively. The hatched white regions are the distance matrices between the new query and the rest of the matrices. Embedding at upper right is obtained from eigensolver (true embedding). Embedding at lower right is obtained from Nyström (approximated embedding).

Nyström extension, as shown in Fig. 3, is a popular technique for finding numerical approximations to the eigenproblem. As discussed in Section 4, it can also be used to extrapolate DM to out-of-sample data. However, in Eq. 5, the embedding of out-of-sample data is assumed to be extrapolated using all samples from the database. This is not practical for large databases. The latest Nyström technique is to draw a few samples (called landmarks) from the database and extrapolate the computed eigenfunctions using the quadrature rule [33]. [5] further takes into account that different landmarks should have different weights and proposes a Density-Weighted Nyström technique. Our framework has adopted this idea.

Fig. 4 left shows the general flow diagram of existing embedding retrieval methods that use Nyström extension to compute embedding for both the database (offline) and the queries (online). However, as will be shown in Section 5.3 (and the lower part of Fig. 3), using Nyström extension for both the database and the queries causes projection errors, which distort pairwise distances in the embedding space and thus affect retrieval accuracy. The problem becomes more severe when the number of landmarks is small while the dimension is high.

As we have observed that eigensolver embedding and Nyström embedding are both approximation of the underlying continuous diffusion process and are highly similar, we propose to obtain a true embedding (eigensolver embedding) for the database (offline) based on full eigen-decomposition as shown in Fig. 4 right. Dotted boxes indicate blocks that differ from

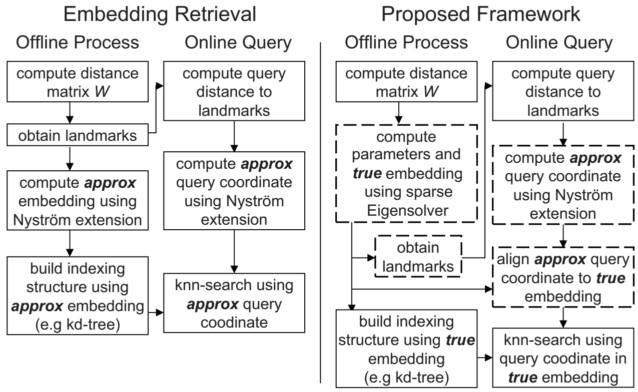


Fig. 4. Retrieval Framework. Left: Nyström extension used in existing work. Right: Our proposed framework using both eigensolver and Nyström extension embeddings. Dotted boxes indicate the differences.

existing works. To compute the query coordinate (online), we also use Nyström extension, but we project the approximated query coordinate back to the true embedding using correspondence analysis during online query search. This gives us a more accurate query coordinate for fast and reliable retrieval.

In the following subsections, we present our framework in detail. Section 5.1 presents a reliable weighted Nyström Extension particularly for DM by separating distribution from the geometry. Section 5.2 discusses how to speed up Nyström for online query search. Section 5.3 discusses how to obtain a true query coordinate by landmark correspondences. Section 5.4 presents an algorithm for selecting landmarks automatically. Section 5.5 discusses how to augment the kernel by one or more similarity measures in the form of shortcut edges. Section 5.6 discusses a heuristic algorithm to compute parameters for DM.

5.1 Nyström Extension for the Diffusion Map

Since we want to reduce the number of landmarks in retrieval applications, we have adapted the Density-Weighted Nyström technique here. [5] solves the normalized cut eigenproblem:

$$\frac{K(x,y)}{D^{1/2}(x)D^{1/2}(y)}\psi(y) = \lambda\psi(y) \tag{6}$$

with the following Nyström equation by taking density of landmarks $S(y)$ into account:

$$\hat{\psi(x)} = \frac{1}{\lambda}\sum D^{-1/2}(x)K(x,y)S(y)D^{-1/2}(y)\bar{\psi}(y) \tag{7}$$

where K is the kernel similarity matrix and D is diagonal degree matrix. Eq. 7 extrapolates eigenvectors (embedding) $\bar{\psi(y)}$ computed on a subset $\hat{O} \subset O$, called landmarks, to the whole database O by using density-adjusted distances $K(x,y)S(y)$ from x to all landmarks y as weights. This Density-Weighted Nyström technique was original designed for Kernel Principal Component Analysis.

Before we can use Eq. 7 to solve the eigenproblem of Eq. 2, we need to obtain K. Recall that K is the kernel matrix obtained from K_W with distribution separated from the geometry. K is required to compute $\bar{\psi}$, $D(x) = \sum_y K(x,y)$ and the extension. Separating distribution from the geometry is an essential step for DM to obtain a true Laplace-Beltrami operator and to analyze the underlying manifold. To obtain K, we need to compute $p_W(x) = \sum_{y \in O} K_W(x,y)$, which involves the whole matrix K_W. In our retrieval application, we usually have submatrices A_W and B_W of $K_W = \begin{bmatrix} A_W & B_W \\ B_W^T & C_W \end{bmatrix}$ only.

To obtain distribution separated K from K_W efficiently without requiring the full matrix of K_W as inspired by [33], we first follow the derivation in [5] by considering the following eigenproblem:

$$\int K_W(x,y)S(y)\psi(y)dy = n\lambda\psi(x) \tag{8}$$

where K_W is a symmetric kernel. Using $K_W(x,y)S(y)$ instead of $K_W(x,y)$ takes density $S(y)$ of landmarks into account. Since $K_W(x,y)S(y)$ is not symmetric, we let $\bar{\psi}(y) = S(y)^{-\frac{1}{2}}\mu(y)$ and convert the eigenproblem into:

$$\int S(x)^{1/2}K_W(x,y)S(y)^{1/2}\mu(y)dy = n\lambda\mu(y) \tag{9}$$

where $S(x)^{1/2}K_W(x,y)S(y)^{1/2}$ is symmetric. This suggests the following Nyström approximation scheme if we want to eigendecompose K_W:

$$\hat{\psi}(x) = \frac{1}{n\lambda}\sum K_W(x,y)S(y)\bar{\psi}(y) \tag{10}$$

The Nyström approximated eigenvectors can be rewritten in matrix form: $\hat{\psi} = \begin{bmatrix} A_W \\ B_W^T \end{bmatrix} S\tilde{U}\Lambda^{-1}$, where $A_W = K_W(x,y)$ for $x,y \in \hat{O}$, $B_W = K_W(x,y)$ for $x \in O - \hat{O}, y \in \hat{O}$, $\tilde{U} = \bar{\psi}$, and $\Lambda = n\lambda$. As $\hat{\psi}$ are the approximated leading eigenvectors of K_W, we have:

$$K_W \approx \hat{\psi}\Lambda\hat{\psi}^T = \begin{bmatrix} A_W \\ B_W^T \end{bmatrix} S\tilde{U}\Lambda^{-1}\tilde{U}^T S^T \begin{bmatrix} A_W & B_W \end{bmatrix} \tag{11}$$

Let $\mu = S^{1/2}\tilde{U}$. As $S^{1/2}A_W S^{1/2}$ is symmetric, we have $(S^{1/2}A_W S^{1/2})^{-1} = (\mu\Lambda\mu^T)^{-1} = S^{1/2}\tilde{U}\Lambda^{-1}\tilde{U}^T$ and, $A_W^{-1} = S\tilde{U}\Lambda^{-1}\tilde{U}^T S^T$. By substituting it to Eq. 11:

$$\begin{aligned} K_W \approx \bar{K_W} &= \begin{bmatrix} A_W \\ B_W^T \end{bmatrix} A_W^{-1} \begin{bmatrix} A_W & B_W \end{bmatrix} \\ &= \begin{bmatrix} A_W & B_W \\ B_W^T & B_W^T A_W^{-1} B_W \end{bmatrix} \end{aligned} \tag{12}$$

This derivation shows that the Density-Weighted Nyström has the same matrix completion view as general Nyström [33]. Suppose that we now want to solve the following eigenproblem:

$$\begin{aligned} \int K(x,y)S(y)\psi(y)dy &= \int \frac{K_W(x,y)}{p_W(x)p_W(y)}S(y)\psi(y)dy \\ &= n\lambda\psi(x) \end{aligned} \tag{13}$$

where $p_W(x) = \sum_y K_W(x,y)$, and since we do not have the full matrix K_W, we approximate $p_W(x) \approx \bar{p}_W(x) = \sum_y \bar{K_W}(x,y)$ as follows:

$$\bar{p}_W = \begin{bmatrix} \sum_y A_W(x,y) + \sum_y B_W(x,y) \\ \sum_y B_W^T(x,y) + B_W^T A_W^{-1} \sum_y B_W(x,y) \end{bmatrix} \tag{14}$$

To compute K from K_W, we compute submatrices A and B of $K = \begin{bmatrix} A & B \\ B^T & C \end{bmatrix}$ as:

$$\begin{aligned} A(x,y) &\leftarrow \frac{A_W(x,y)}{\bar{p}_W(x)\bar{p}_W(y)}, \qquad x, y \in \hat{O} \\ B(x,y) &\leftarrow \frac{B_W(x,y)}{\bar{p}_W(x)\bar{p}_W(y)}, \qquad x \in O - \hat{O}, y \in \hat{O} \end{aligned} \tag{15}$$

After obtaining A and B, we can use the Density-Weighted Nyström extension of Eq. 7 to solve graph normalization and extrapolate eigenfunctions to the whole database. We avoid computing the huge submatrix C, which is the distance matrix among non-landmarks, to gain speed up.

5.2 Nyström Speed-up for Retrieval

Nyström extension both avoids the expensive eigen-solver on large matrices and reduces computing the expensive similarity measures. Given an unknown query, it is sufficient to evaluate the distances between the new query and the landmarks (the hatched regions of the lower matrix in Fig. 3) to obtain the query embedding coordinate. Since the coordinate is Euclidean, a spatial indexing technique, such as kd-tree, can be used. This is very attractive because graph- / bag-based methods are usually slow and non-metric. Nyström extension thus provides a way to scale these algorithms to large databases.

To compute a query coordinate, we construct:

$$\hat{B_W}(x,y) \leftarrow \begin{bmatrix} B_W(x,y) & \exp\left(-\frac{W(q,y)^2}{\sigma}\right) \end{bmatrix} \tag{16}$$

where $x \in O - \hat{O}$. $W(q,y)$ is the distance between new query q and landmark $y \in \hat{O}$. To further speed up online query search, we may precompute eigen-decomposition. We note that eigen-decomposition of submatrix A depends on our proposed distribution separation step. Given a new query, distribution $\bar{p}_W$ can be written as:

$$\begin{aligned} &\begin{bmatrix} \sum_y A_W(x,y) + \sum_y \hat{B_W}(x,y) \\ \sum_y \hat{B_W^T}(x,y) + \hat{B_W}^T A_W^{-1} \sum_y \hat{B_W}(x,y) \end{bmatrix} \\ &= \begin{bmatrix} \sum_y A_W(x,y) + \sum_y B_W(x,y) + \exp\left(-\frac{W(q,y)^2}{\sigma}\right) \\ \sum_y \hat{B_W^T}(x,y) + \hat{B_W}^T A_W^{-1} \sum_y \hat{B_W}(x,y) \end{bmatrix} \\ &\simeq \begin{bmatrix} \sum_y A_W(x,y) + \sum_y B_W(x,y) \\ \sum_y B_W^T(x,y) + B_W^T A_W^{-1} \sum_y B_W(x,y) \end{bmatrix} \end{aligned} \tag{17}$$

The last step is a good approximation since the addition of $\exp\left(-\frac{W(q,y)^2}{\sigma}\right)$ is negligible for large databases. The eigenproblem then solely depends on A and S and can be precomputed offline. It is thus sufficient to extrapolate the embedding for all objects (databases and queries) by simple matrix multiplication during online query search. It gives significant speedup especially for large databases.

```
%%% Offline precomputation %%%
Aw_inv = pinv(Aw);
% precompute distribution
bar_p1 = sum([Aw; Bw'], 1);                    Eq.17
% separate distribution from geometry
A = Aw./(bar_p1*bar_p1');                      Eq.15
% eigen decomposition
d_Az_si = diag(1./sqrt(sum(A*S,2)));           Eq.9
[V E] = eig(d_Az_si*(A*S)*d_Az_si);
[V,E] = SortEigen(V,E);
% precomputed result
product_VE = d_Az_si*V*pinv(E);
```

```
%%% Online query extension %%%
% append query to landmarks distance to B
hat_Bw = [Bw W];                               Eq.16
% distribution adjustment
p1=sum([Aw;hat_Bw'],1);                        Eq.17
p2=sum(hat_Bw,1)+sum(hat_Bw',1)*Aw_inv*hat_Bw;
bar_p = 1./[p1 p2]';
A = Aw.*(bar_p(1:n)*bar_p(1:n)');              Eq.15
hat_B=hat_Bw.*(bar_p(1:n)*bar_p(n+(1:m))');
% weighted Nystrom
d_Ax = sum([A;hat_B']*S,2);                    Eq.7
d_Ax_si = diag(1./sqrt(d_Ax));
d_pi = d_Ax ./ sum(d_Ax);
% extension by precomputation
V_ex=d_Ax_si*([A;hat_B']*S)*product_VE;        Eq.7
% diffusion map embedding
for i=1:size(V_ex,2)                           Eq.3
  V_left(:,i) = V_ex(:,i)./sqrt(d_pi);
end
```

Fig. 5. Example MATLAB code for computing Diffusion Map using distribution adjustment, density-weighted Nyström extension and heuristic precomputation. A and B are submatrices of K.

Fig. 5 shows some Matlab codes, with corresponding equation numbers listed on the right. We separate offline precomputation from online query extension. In the online section, a simple multiplication of `[A;hat_B']*S` with the precomputed eigenvector matrix `product_VE` is required to extrapolate all coordinates including the database and query objects.

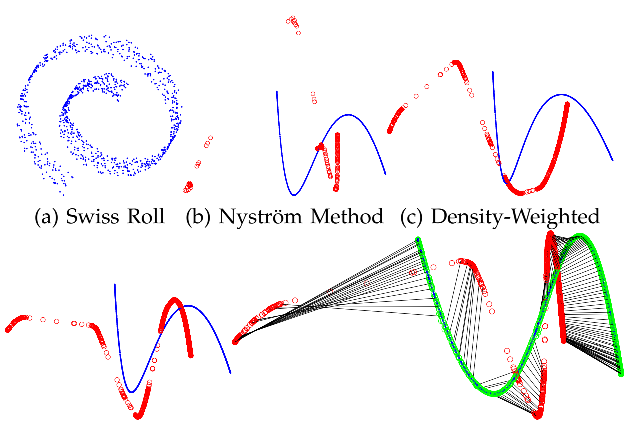
5.3 Query Alignment

Although Nyström embedding is generally assumed to give best approximation, its effect on retrieval accuracy is rarely discussed. We have tested Nyström and our Density-Weighted Nyström algorithms to obtain the diffusion embedding for a toy example of Swiss Roll, shown in Fig. 6(a). Distortions are observed in Fig. 6(b)-6(c). When applied to some other models, as shown in the right hand side of Fig. 3, the quality of Nyström embedding is not good since some coordinates of Dog are closer to those of Wolf. When we apply the nearest neighbor search on the Nyström embedding, retrieval accuracy degrades. This is a practical limitation of Nyström extension. Although increasing the number of landmarks can improve the quality of embedding, it also increases the computational cost due to the increase in the number of expensive similarity comparisons of landmarks.

Our proposed solution is based on the understanding that *eigensolver* (computed from $O \subset \Omega$) and *Nyström* (computed from $\hat{O} \subset O$) embeddings are both approximations of the continuous diffusion process on Ω and are highly similar. If we can align the query Nyström coordinate to the eigensolver one as in Fig. 6(d)-6(e), we may conduct the nearest neighbor search by building a spatial indexing tree on the eigensolver embedding. This results in a fast (Nyström) and accurate (eigensolver) retrieval scheme.

Aligning two embeddings from two different sources and establishing correspondences between them is a difficult problem. [38] establishes correspondences for extrinsic geometry of 3D non-rigid shapes by aligning their intrinsic geodesic embeddings. Three subproblems are factored: *eigenmode switching* – the switching order of eigenvectors when corresponding eigenvalues are close, *sign flip* – the arbitrary determination of signs by the numerical eigensolver, and *non-rigid alignment* – the misalignment is due to the distortion of the embedding processes, the use of geodesic for distance approximation and the sampling issue. The first two subproblems are practical issues of the numerical eigensolver, and can be handled by exhaustive pairing and evaluating the leading eigenvectors. [38] handles the third by *Iterative Closest Point* (ICP) and *Thin Plate Spline* (TPS).

Inspired by [38], we adopt a similar approach but ours is much simpler because our embeddings are based on the *same* underlying diffusion process.



(a) Swiss Roll (b) Nyström Method (c) Density-Weighted

(d) Alignment (e) TPS aligned embedding

Fig. 6. A test example: (a) Swiss Roll. (b) Embedding obtained from Nyström extension. (c) Embedding obtained from Density-Weighted Nyström. (d) Embedding obtained after eigenvector swapping and sign change. (e) Embedding obtained after TPS transformation (green) using landmark correspondences (black lines) with (d) (red). Embedding overlaps nicely with eigensolver embedding (blue).

Algorithm 1 Aligning Eigenmode-Switched and Sign-Flipped Embeddings.

Require: $\psi_{1..d}$, $\hat{\psi}_{1...s}$, d, s
Ensure: $\widehat{\psi_{align}}_{1..d}$
define a set $J \leftarrow 1..s$
for $i \leftarrow 1$ **to** d **do**
  find the eigenpair with largest $\|\psi_i \cdot \hat{\psi}_j\|$, $j \in J$
  if $\psi_i \cdot \hat{\psi}_j < 0$ **then**
    $\hat{\psi}_j \leftarrow -1 \times \hat{\psi}_j$
  end if
  $J \leftarrow J \backslash j$
  $\widehat{\psi_{align}}_i \leftarrow \hat{\psi}_j$
end for

To handle eigenmode switching and sign flip, we propose to perform a greedy search on s leading eigenvectors, as shown in Algorithm 1. We find that setting $s = 3d$, where d is the dimension, generally performs well. Instead of evaluating the Euclidean distance between two whole sets of embedding, we maximize the value: $\arg\max_{\widehat{\psi_{align}}} \sum_i \psi_i \cdot \widehat{\psi_{align}}_i$, where ψ is the eigensolver and $\hat{\psi}$ is the Nyström embeddings. $\widehat{\psi_{align}}$ is the aligned embedding of $\hat{\psi}$. The larger its value, the better the alignment is between the two sets of embedding. Simple dot product is sufficient here as ψ and $\hat{\psi}$ are very similar to each other.

After finding the best eigenpairs, we project the query coordinate onto the eigensolver embedding. Contrary to [38], our method does not require ICP to establish correspondences. We use TPS to transform Nyström to the eigensolver embedding through landmark correspondences $\hat{O}$, as shown in Fig. 6(e). TPS is a coordinate transform method [39]. Given some correspondence anchors (x_i, y_i), TPS finds the function f that passes through them with minimal error. f can then be used to interpolate the transformed coordinate of an arbitrary query.

$$
\begin{aligned}
E(f) &= \sum_i \|\psi(y_i) - f(x_i)\| + I_f \\
I_f &= \iint \left(\frac{\partial^2 f}{\partial x^2}\right)^2 + 2\left(\frac{\partial^2 f}{\partial x \partial y}\right)^2 + \left(\frac{\partial^2 f}{\partial y^2}\right)^2 dxdy
\end{aligned}
\tag{18}
$$

Following [40], we compute the transformation parameters (D, w) by minimizing Eq. 18, which is equivalent to solving the following linear equation:

$$
\begin{bmatrix} \Gamma - I & X^T \\ X & 0 \end{bmatrix} \begin{bmatrix} w \\ D \end{bmatrix} = \begin{bmatrix} Y \\ 0 \end{bmatrix} \tag{19}
$$

where X and Y are $(d+1)$ homogeneous dimension coordinates of Nyström and eigensolver embeddings of landmarks $\hat{O}$. I is an identity matrix. D is a $(d+1) \times (d+1)$ affine transform matrix. w is a matrix of $\left|\hat{O}\right| \times (d+1)$ warping coefficients. $\Gamma_{ij} = -\log\|\widehat{\psi_{align}}(x_j) - \widehat{\psi_{align}}(x_i)\| = \gamma_j(\widehat{\psi_{align}}(x_i))$. Once the landmarks are fixed, D and w can be precomputed offline. Given an online query, the aligned query coordinate can be computed from Nyström coordinate: $f(\widehat{\psi_{align}}(q), D, w) = \widehat{\psi_{align}}(q) \cdot D + \gamma(\psi_{align}) \cdot w$.

5.4 Choosing Landmarks

As in [5], we first generate the set of landmarks using K-means clustering, with random sampling as initialization. We compute the density of each cluster and the Nyström embedding, and then align the Nyström embedding to the eigensolver embedding. The previously defined dot-product (Algorithm 1, line 3) can then be used for fast pruning of poor embedding. The higher the score, the better the quality of the Nyström embedding is. We repeat the whole step a few times (20 in our implementation) simply because K-means is initialized by random parameters. Once we get the best set of landmarks of a given k, we check if the score is good enough for retrieval. We compute the Precision and Recall (PR) using such embedding and compare this PR with that of the true embedding. This error is defined as: $\|true\_pr - nystrom\_pr\|$, where $true\_pr(i)$ and $nystrom\_pr(i)$ are the i^{th} precision values of the average PR curves of the true and Nyström embeddings. If this error is less than a certain threshold, the algorithm outputs the best set of landmarks; otherwise, we increment k. This algorithm is an offline process.

Summarizing the above subsections, our method may be considered as optimizing the cost function:

$$\operatorname*{arg\,min}_{\hat{O} \in O, (D,w)} \sum_{i \in \hat{O}} (\psi(x_i) - f(\hat{\psi}(x_i)))^2 + \hat{\psi}^T K \hat{\psi} + \lambda I_f \quad (20)$$

where $I_f \propto w^T \Gamma w$, $\lambda = 1$. $f(\hat{\psi}(x_i))$ and $\psi(x_i)$ are the coordinates of the Nyström and eigensolver embeddings, respectively, of landmark x_i. The first term requires the coordinates of Nyström and eigensolver embeddings to be the same at x_i. The second term defines the inner product of the spectral embedding [41], with K defined in Eq. 15. The last term I_f imposes smoothness constraints through TPS (non-rigid transformation) [40]. Optimizing Eq. 20 gives the best set of landmarks, $\hat{O} \in O$, and the associated TPS parameters (D, w). However, we do not optimize this function here because it requires enumerating all possible subsets of $\hat{O} \in O$ and solving Eq. 19 in each step, which involves a dense matrix Γ of size: $\left|\hat{O}\right| \times \left|\hat{O}\right|$. Instead, we follow [5] to find landmarks by K-means and [38] to handle sign-flip and eigenmode switch, which are practical and fast.

5.5 Augmenting the Kernel Matrix

So far, we have assumed that all data lie on individual manifolds. If the features are unstable or there are insufficient samples in the database, manifolds may become disjoint. Here, we propose to directly augment the kernel matrix with "shortcut edges", to link these disjoint manifolds together. In general, if one method is not sufficient to differentiate two models, it is common to add other similarity measure(s) for adjustment. For example, [23] proposes a two-step pruning process. Our idea is similar in that we use two or more similarity measures to build an automated system. However, our approach is much simpler and more efficient because it allows retrieval in one step as shown in Section 5.1.

We recall that kernel matrix K_W is the Markov matrix defining the probability of diffusion. With respect to spectral graph theory, a probability greater than 0 means that there is an edge connecting the two nodes. After the parameter optimization step in Section 5.6, the kernel matrix becomes a sparse matrix and each of the non-zero entries corresponds to the nearest neighbor of local scale σ. We consider the following augmented kernel:

$$K_W = \sum_i \exp\left(-\frac{W_i^2}{\sigma_i}\right) \quad (21)$$

where W_i are different distance matrices. The non-zero entries of W_i are the diffusion probabilities to their neighbors. By adding them all together, we introduce extra probabilities, in the form of shortcut edges, to the original kernel, where such connections might not exist. In general, this approach can be applied to 2 or more similarity measures. Note that though K_W may now have probabilities greater than 1, it is normalized by Eq. 1.

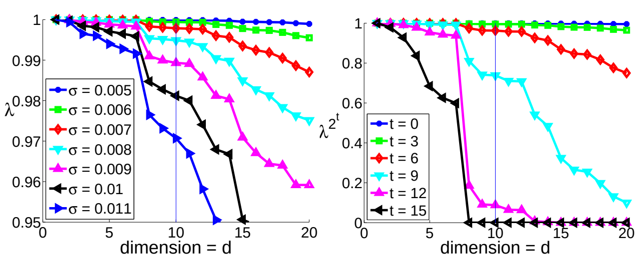
The above idea is based on the following observation. Given a large database with a lot of similar models, general retrieval methods are usually reliable for the first few matches. Therefore, it is possible to combine the first few good *nearest neighbors* to form a new embedding distance. In other words, we are introducing reliable short-cut edges (nearest neighbors from another method) to bridge the disjoint manifolds of the original data set or to improve the embedding distances. Since we restrict this to a few nearest neighbors only, the effect due to the possible introduction of unreliable short-cut edges is small. [41] also considers using a sparse set of "short-circuits" to align manifold. It improves the embedding quality by manually defining some sparse set of points. We, on the other hand, take advantage of the dense set of correspondences to form short-cut edges from two or more distance measures to improve the resulting embedding and retrieval results.

5.6 Choosing Parameters for the Diffusion Map

There is not much discussion on choosing the appropriate parameter values, i.e., σ (the gaussian width), t (the diffusion step) and d (dimension), for the Diffusion Map in the original work [4] and relevant literature. Finding the values for these parameters are generally data-dependent. One of our assumptions of using manifold learning to improve retrieval accuracy is that the dissimilarity measure is able to separate different classes of models. Here we take a heuristics approach to determine these parameters values.

In DM, σ determines the local scale of the neighborhood. [21] proves that when $\sigma \rightarrow 0$, the kernel approaches the Laplace-Beltrami Operator. This produces smooth eigenvectors that are good for harmonic extension. A very small σ, however, results in high multiplicity of 1 in the spectrum (list of eigenvalues). From spectral graph theory, multiplicity of 1 counts the number of disconnected components (manifolds in our case). The disconnected components are stored in the associated eigenvectors. On the one hand, this is good because it exaggerates distances among different manifolds. On the other hand, if the multiplicity of 1 is greater than d, we lose important eigenvectors and so grouping information in the embedding space. This imposes a constraint that the best σ should give the largest multiplicity of 1 that is less than dimension d. For example, σ is set to 0.006 in Fig. 7(a).

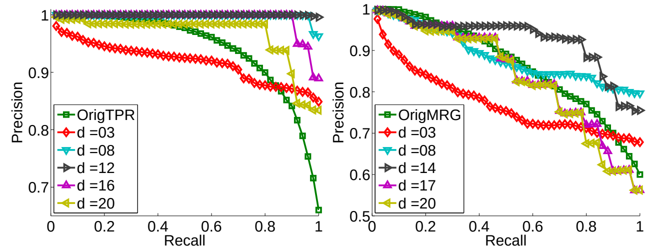
Parameter t is the dyadic power to diffuse local distances to infer far distances. It holds the key to dimension reduction as shown in Fig. 7(b). From the signal processing point of view, summation of spectrum represents the total energy. It is desirable to have all the energy concentrated at the first few d dimensions so that all the important information is well-represented. Since noise is usually located at high frequencies, it may be desirable to truncate high frequency components. Due to these reasons, we compute t by defining $\xi = {\lambda_d}^{2^t}/{\lambda_1}^{2^t} = 0.1$ to truncate noise [4], where λ_d is the d^{th} eigenvalue of the spectrum. Fig. 7(b) shows the effect of t on the spectrum with $\sigma = 0.006$. $t \approx 12$ achieves $\xi = 0.1$.



(a) λ vs. d at varying σ (b) λ^{2^t} vs. d at varying t and $\sigma = 0.006$

Fig. 7. Relationships among σ (local scale), d (dimension), and t (dyadic power for diffusion).

From the above discussion, σ and t are both dependent on d. From Fig. 8(a) and 8(b), we observe that if the dimension is too small or too large, the retrieval accuracy will be affected, and there is usually one best dimension d such that the method attains the best retrieval accuracy (i.e., TPR: $d = 12$ and MRG: $d = 14$). We believe that such dimension is related to the intrinsic dimension of the data. Therefore, we take a simple approach by trying different possible dimensions d and compute the average precision and recall for the dataset. We then take the dimension d with the minimum cost:



(a) TPR (b) MRG

Fig. 8. Determining the dimensions of Diffusion Map for the TPR and MRG methods.

$$\arg\min_{d} \sum_{i \in [1\ldots 50]} |1 - pr(i)| \tag{22}$$

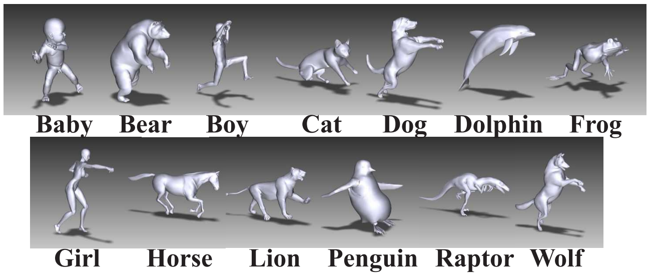
where $pr(i)$ is the i^{th} precision value on the average PR curve. Although this search may still be slow for very large datasets, it is an offline process. We believe that trading off speed for accuracy here is important for reliable retrieval. We have applied and evaluated these settings empirically in the next section.

6 EXPERIMENTAL RESULTS

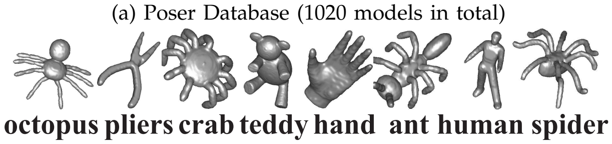
We have incorporated two distance measures, MRG [1] and TPR [2] into our retrieval framework. To evaluate the performance, we have created a database consisting of 1,020 articulating geometry models. They are divided into 13 groups. Some are very distinct (e.g., Frog), while others are very similar to each other (e.g., Dog and Wolf, Lion and Cat) as shown in Fig. 9(a). These models are generated from animation sequences so that every model is different from the others. We refer to this dataset as the Poser dataset. We have also downloaded a publicly available McGill dataset [42] for testing, as shown in Fig. 9(b). Since TPR [2] has a basic assumption that the input models should have more than 1 limb. We have manually removed two model groups, snake and spectacle, from the original McGill dataset in order to carry out the following experiments.

6.1 Precision and Recall Comparison

Fig. 10(a) and 10(b) show the Precision and Recall (PR) curves of the two distance measures (TPR, MRG), and the performances of applying multi-dimensional scaling (MDS) and diffusion map (DM) to these distance measures on the Poser and McGill databases. These curves show the best PR values obtained by varying the number of dimensions between 3 to 20 (Eq. 22), and the corresponding number of dimensions is displayed next to the method names. The dimension of MDS is set to the maximum of all since the higher the dimension, the better it can preserve the original



(a) Poser Database (1020 models in total)



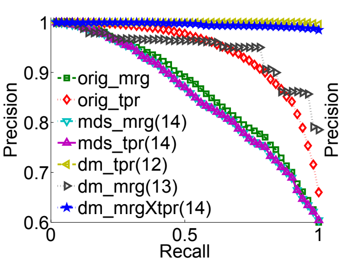
(b) McGill Database (205 models in total)

Fig. 9. Datasets used for evaluation.

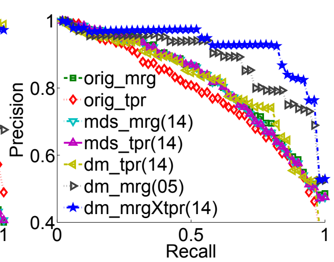
distances. Fig. 10(c) and 10(d) show the corresponding PR performances on smaller databases, created by removing even indices from the two databases.

We have several observations from Fig. 10:

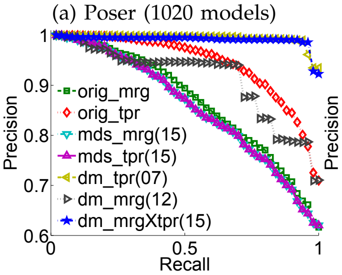
1) With the Poser database, DM can improve the performance of TPR and MRG, while MDS cannot. Although MDS and TPR can separate these model groups well in Fig. 1, they are affected by the high-dimensionality and misalignment of features. With MDS, all (far and close) pairwise distances are preserved in the embedding space, which is not useful to analyze data lying on manifold. Therefore, our method can handle these data better.
2) With the McGill database, MDS again fails to improve the distance measures. We observe that DM fails to improve with TPR too. The main reason is that TPR relies on capturing surface details (e.g., curvature and thickness) for retrieval, but most of these models have similar/smooth rod-like structures and so the original TPR also performs poorly. DM gives much better performance when combined with MRG.
3) With both Poser and McGill databases, combining both MRG and TPR methods seems to give a much better retrieval results. The reason is that though one distance measure may not always reflect the true distances between models (e.g., due to noise, unreliable features), other distance measures can still improve the embedding distances by introducing reliable short-cut edges and drag related models together. In other words, the combined kernel can better capture the intrinsic structure of the data.



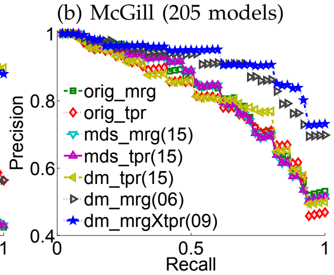
(a) Poser (1020 models)



(b) McGill (205 models)



(c) Poser (510 models)



(d) McGill (103 models)

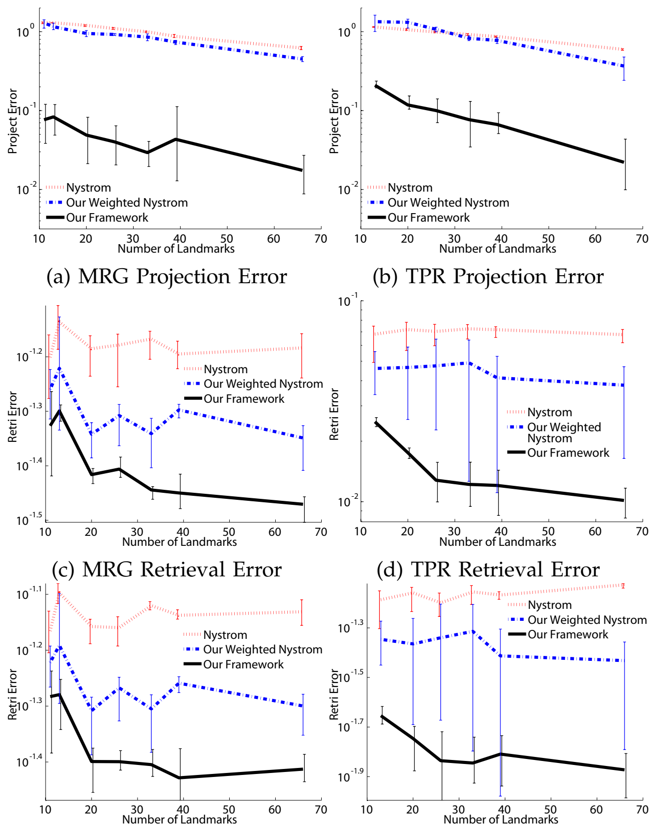
Fig. 10. Retrieval performance using our parameter settings. MRG and TPR are applied on the Poser and McGill datasets. They are compared to retrieval performance after applying MDS and Diffusion Map.

With reduced databases shown in Fig. 10(c) and 10(d), we originally expected that more data should lie on disconnected manifolds, degrading retrieval performances. Although there is a drop in performance, the drop is slight. Our conclusion is that since the features involved in articulated geometry model retrieval are usually high-dimensional, these data are better analyzed by manifold learning techniques.

6.2 Nyström Alignment and Retrieval Error

Fig. 11 and 12 (both in logarithmic scale) compare our framework with Nyström extension on the McGill and the Poser databases. We first randomly sample 33% of the data as out-of-samples (testing data) and build a database using the remaining 66% of the data (training data). The two figures are produced by finding 20 sets of landmarks, computing the PR curves on the best 5 sets, and then computing the projection and retrieval errors. The retrieval error is defined as: $\|orig\_pr - pr\|$, where $orig\_pr(i)$ and $pr(i)$ are the i^{th} precision values of the average PR curves. $orig\_pr$ is the retrieval performance when there is no separation of training and testing data and the embedding is computed based on the eigensolver embedding (ground truth). pr is the retrieval performance using Nyström extension or our framework when training and testing data are separated. We use the retrieval error as a measure because the traditional retrieval framework uses Nyström extension for both database creation and out-of-sample extension. Our framework uses eigensolver embedding for database creation but Nyström extension and alignment for out-of-sample extension. To evaluate whether the retrieval performance is preserved, we measure the retrieval errors.

Fig. 11(a), 11(b), 12(a) and 12(b) show the projection errors due to samples in the databases (training data of both landmarks and non-landmarks). Fig. 11(c), 11(d), 12(c) and 12(d) show the retrieval errors using samples in the databases. Fig. 11(e), 11(f), 12(e) and 12(f) show the retrieval errors using those out-of-samples (testing data). *Nyström* refers to the Nyström

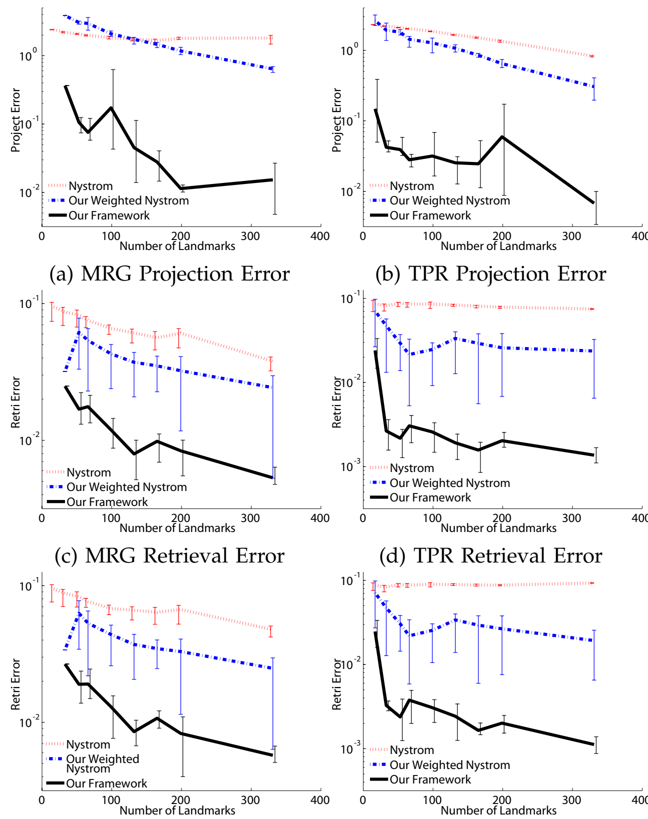


(a) MRG Projection Error (b) TPR Projection Error

(c) MRG Retrieval Error (d) TPR Retrieval Error

(e) MRG Out-of-Sample Retri Err (f) TPR Out-of-Sample Retri Err

Fig. 11. Comparing the projection and retrieval errors when using Nyström, Density-Weighted Nyström and our approach on the McGill dataset.



(a) MRG Projection Error (b) TPR Projection Error

(c) MRG Retrieval Error (d) TPR Retrieval Error

(e) MRG Out-of-Sample Retri Err (f) TPR Out-of-Sample Retri Err

Fig. 12. Comparing the projection and retrieval errors when using Nyström, Density-Weighted Nyström and our approach on the Poser dataset.

extension method (Eq. 5). *Weighted Nyström* refers to the Density-Weighted Nyström method with our proposed distribution adjustment (Section 5.1). Both of them are integrated into the traditional framework (Fig. 4 left). *Our Framework* refers to our retrieval framework (Section 5). Fig. 11 and 12 also show the maximum, minimum and average values.

In summary, Density-Weighted Nyström with our distribution adjustment performs better than Nyström extension. This shows that our proposed adjustment allows Density-Weighted Nyström to be used to approximate the Diffusion Map. Our retrieval framework performs even better in almost all situations on the McGill database. Our observation is that Nyström techniques usually work on large datasets. For example, [5] applies the Density-Weighted technique on 6000 images and compares the first 3 eigenvectors. Here, we only have 205 models in the McGill database and our retrieval is operating at dimension 5 (MRG) and 14 (TPR). We believe that this causes the approximations to be sensitive to the distribution of data and the choices of landmarks. This also explains the observation that our method generally performs better in the Poser database, as the Poser database is larger and its models are shown to lie on manifold. This suggests that our framework performs better in approximating diffusion embedding in these situations.

6.3 Time Comparison

Table 1(a) compares the online retrieval time per query on the Poser database. The *TPR* and *MRG* columns show the sequential search times using the two methods. TPR is metric, and VP-tree (a distance-based indexing technique) can be used. The *VP-tree* column shows the retrieval time of using VP-tree. However, due to the high dimensionality, the method soon approaches brute-force when $k \approx 50$. The actual retrieval time is even higher than brute-force, due to the overhead of tree searching when $k > 50$. Since MRG is non-metric, indexing techniques cannot be used. The retrieval time for a query is long as it involves sequential scanning of the whole database.

Our retrieval framework can incorporate both methods, whether metric or non-metric, with the assumption that the feature space lies on manifolds. If we take half of the database as landmarks, it is roughly 1.4 (for TPR) or 1.8 (for MRG) times faster than the corresponding sequential search, depending on the accuracy needed, as shown in the *TPR+Ours* and *MRG+Ours* columns. The speedup is mainly due to the reduced number of distance computations between the query and landmarks. As a spatial indexing technique, e.g., kd-tree, is extremely fast (when $d \leq 20$), the time spent on k-nn search is so small that it can be neglected compared to the time spent on evaluating the similarity measure. When our retrieval framework is applied on the two methods (MRG and TPR), the computational costs are roughly constant as the required number of nearest neighbors increases. In addition, our method does not suffer from the curse of dimensionality because of the reduced dimension, and consistently requires less time than the indexing approach even when k is small. The VP-tree indexing and distance computation is implemented with C++, while our proposed framework is implemented in Matlab. All experiments were performed on an Intel(R) 2.67GHz i7 Quad Core PC.

Table 1(b) shows the preparation time for the Poser database in our experiment. In the *Parameter Search* phase, *Embed* refers to the time to embed pairwise distances into diffusion embedding of dimension 3-20. It includes the time to select the parameters σ and t. We also sparsified the kernel by setting value $\leq 1e^{-8}$ to zero, because this does not affect empirically the resulting embedding and retrieval performance. *Retrieve* refers to the time to find dimension d for the best retrieval performance. In the *Landmark Search* phase, *Landmark* refers to the time taken by the K-means clustering process, generation of landmarks, eigenmode switching, sign-flip, thin plate spline alignment and eigen-precomputation. *Projection* refers to the time to compute projection errors in Fig. 12a and 12b. All these are offline processing times and are shown here for completeness. These steps are required in our framework in order to gain online retrieval speedup.

TABLE 1
Time Comparison.

k-NN	TPR	VP-tree	TPR+Ours	MRG	MRG+Ours
1	1.034	0.477	0.705	3.189	1.726
5		0.832	0.705		1.726
10		0.903	0.705		1.726
50		1.052	0.705		1.727
100		1.157	0.706		1.727
500		1.195	0.709		1.730
1020		1.193	0.713		1.734

(a) Online Retrieval Time (in seconds)

Method	Parameter Search		Landmark Search	
	Embed	*Retrieve*	*Landmark*	*Projection*
TPR	1327.3	321.88	173.87	5453.81
MRG	2055.9	585.44	179.63	3826.57

(b) Offline Preparation Time (in seconds)

7 Conclusion and Future Work

In this paper, we have pointed out that existing graph or bag-based matching methods cannot handle articulated geometry models with similar skeleton, as they are projected on nearby manifolds. Increasing the number of geometric features may increase the distances among these manifolds but the intra-cluster variance is usually larger than inter-cluster distances. Retrievals using these methods may fail especially when the database contains many similar-skeleton models. Our idea here is to apply manifold learning techniques to exaggerate inter-cluster distances. To handle large databases, we have also adapted the Density-Weighted Nyström extension for the computation of the Diffusion Map and used correspondence analysis to define a retrieval framework to reduce Nyström extension errors. We have shown with a number of experiments that the proposed framework improves retrieval accuracy and speed.

Our method assumes that the model features lie on separate manifolds and thus separable in the high dimensional space. When the model features do not lie on separate manifolds, the proposed heuristic method for computing the DM parameters may no longer separate these manifolds well. We solve this by introducing shortcut edges using different similarity measures. In the future, we would like to explore other manifold learning techniques, and to develop more representative features to address this issue. Our method also assumes that a true embedding is available, and we take the eigensolver embedding as the true embedding. As the data size reaches the limit of the hardware requirement (e.g., memory), it may be difficult to obtain a full eigensolver embedding. One possibility is to use Density-Weighted Nyström extension on as many landmarks as possible and take this as the true embedding. Given a sufficient number of landmarks, our Density-Weighted Nyström method approaches the eigensolver embedding, and our framework can be applied for faster online query.

Though our current focus is 3D articulated models, we believe that the framework is applicable to a wider class of applications (e.g., non-articulated models and time-series, including video and motion captured data), which involve high-dimensional features. As we have shown that features of articulated geometry models lie on manifolds, one interesting question is whether these features can be used to build a statistical model. We are planning to compare the implicit optimization with our study here.

Acknowledgements

The authors would like to thank the anonymous reviewers for their helpful and constructive comments on this paper. The work described in this paper was partially supported by a SRG grant from City University of Hong Kong (Project Number: 7002576). Gary Tam was supported by an EPSRC research studentship during his study at Durham University.

References

[1] M. Hilaga, Y. Shinagawa, T. Kohmura, and T. Kunii, "Topology matching for fully automatic similarity estimation of 3d shapes," in *Proc. ACM SIGGRAPH*, 2001, pp. 203–212.

[2] G. Tam and R. Lau, "Deformable model retrieval based on topological and geometric signatures," *IEEE Trans. VCG*, vol. 13, no. 3, pp. 470–482, 2007.

[3] R. Coifman and S. Lafon, "Diffusion maps," *Applied and Computational Harmonic Analysis*, vol. 21, no. 1, pp. 5–30, 2006.

[4] S. Lafon and A. Lee, “Diffusion maps and coarse-graining: A unified framework for dimensionality reduction, graph partitioning, and data set parameterization,” *IEEE Trans. PAMI*, vol. 28, no. 9, pp. 1393–1403, 2006.
[5] K. Zhang and J. Kwok, “Density-weighted nyström method for computing large kernel eigensystems,” *Neural Computation*, vol. 21, no. 1, pp. 121–146, 2009.
[6] M. Kazhdan, T. Funkhouser, and S. Rusinkiewicz, “Rotation invariant spherical harmonic representation of 3d shape descriptors,” in *Proc. EG SGP*, 2003, pp. 156–164.
[7] R. Osada, T. Funkhouser, B. Chazelle, and D. Dobkin, “Matching 3d models with shape distributions,” in *Proc. IEEE Conf. on Shape Modeling and Applications*, 2001, pp. 154–166.
[8] J. Tangelder and R. Veltkamp, “A survey of content based 3d shape retrieval methods,” in *Proc. IEEE Conf. on Shape Modeling and Applications*, 2004, pp. 145–156.
[9] A. Ion, N. Artner, G. Peyre, S. Marmol, W. Kropatsch, and L. Cohen, “3d shape matching by geodesic eccentricity,” in *Proc. CVPR Workshops*, 2008, pp. 1–8.
[10] R. Liu, H. Zhang, A. Shamir, and D. Cohen-Or, “A part-aware surface metric for shape analysis,” *Computer Graphics Forum*, vol. 28, no. 2, 2009.
[11] M. Reuter, F. Wolter, and N. Peinecke, “Laplace-beltrami spectra as ”shape-dna” of surfaces and solids,” *Computer-Aided Design*, vol. 38, no. 4, pp. 342–366, 2006.
[12] R. Rustamov, “Laplace-beltrami eigenfunctions for deformation invariant shape representation,” in *Proc. EG SGP*, 2007.
[13] R. Ohbuchi and T. Furuya, “Distance metric learning and feature combination for shape- based 3d model retrieval,” in *Proc. ACM Workshop on 3D Object Retrieval*, 2010, pp. 63–68.
[14] M. Ovsjanikov, A. Bronstein, M. Bronstein, and L. Guibas, “Shape google: a computer vision approach to isometry invariant shape retrieval,” in *Proc. Workshop on Nonrigid Shape Analysis and Deformable Image Alignment*, 2009, pp. 320–327.
[15] A. Bronstein, M. Bronstein, M. Ovsjanikov, and L. Guibas, “Shape google: geometric words and expressions for invariant shape retrieval,” *ACM Trans. on Graphics*, 2011.
[16] T. Tung and F. Schmitt, “Augmented reeb graphs for content-based retrieval of 3d mesh models,” in *Proc. IEEE Conf. on Shape Modeling and Applications*, 2004, pp. 157–166.
[17] S. Biasotti, S. Marini, M. Spagnuolo, and B. Falcidieno, “Sub-part correspondence by structural descriptors of 3d shapes,” *Computer-Aided Design*, vol. 38, no. 9, pp. 1002–1019, 2006.
[18] Y. Rubner, C. Tomasi, and L. Guibas, “The earth mover’s distance as a metric for image retrieval,” *Int’l Journal of Computer Vision*, vol. 40, no. 2, pp. 99–121, 2000.
[19] M. Ruggeri and D. Saupe, “Isometry-invariant matching of point set surfaces,” in *Proc. EG Wkshp. on 3D Object Retrieval*, 2008.
[20] J. Tierny, J. Vandeborre, and M. Daoudi, “Reeb chart unfolding based 3d shape signatures,” in *Proc. Eurographics*, 2007.
[21] S. Lafon, “Diffusion maps and geometric harmonic,” PhD Thesis, Yale University, 2004.
[22] M. Demirci, R. van Leuken, and R. Veltkamp, “Indexing through laplacian spectra,” *Computer Vision and Image Understanding*, vol. 110, no. 3, pp. 312–325, 2008.
[23] H. Sundar, D. Silver, N. Gagvani, and S. Dickinson, “Skeleton based shape matching and retrieval,” in *Proc. IEEE Conf. on Shape Modeling and Applications*, 2003, pp. 130–139.
[24] M. Belkin and P. Niyogi, “Laplacian eigenmaps for dimensionality reduction and data representation,” *Neural Comput.*, vol. 15, no. 6, pp. 1373–1396, 2003.
[25] J. Tenenbaum, V. de Silva, and J. Langford, “A global geometric framework for nonlinear dimensionality reduction,” *Science*, vol. 290, no. 5500, pp. 2319–2323, 2000.
[26] S. Roweis and L. Saul, “Nonlinear dimensionality reduction by locally linear embedding,” *Science*, vol. 290, no. 5500, pp. 2323–2326, 2000.
[27] X. He, “Incremental semi-supervised subspace learning for image retrieval,” in *Proc. ACM Multimedia*, 2004, pp. 2–8.
[28] T. Cox and M. Cox, *Multidimensional Scaling*. Chapman & Hall, London, 1994.
[29] H. Wang, S. Yan, T. Huang, and X. Tang, “Maximum unfolded embedding: formulation, solution, and application for image clustering,” in *Proc. ACM Multimedia*, 2006, pp. 45–48.
[30] H. Sahbi, P. Etyngier, J. Audibert, and R. Keriven, “Manifold learning using robust graph laplacian for interactive image search,” in *Proc. CVPR*, 2008, pp. 1–8.
[31] S. Chopra, R. Hadsell, and Y. LeCun, “Learning a similarity metric discriminatively, with application to face verification,” in *Proc. CVPR*, 2005, pp. 539–546.
[32] J. Platt, “Fastmap, metricmap, and landmark mds are all nyström algorithms,” in *Proc. Int’l Workshop on Artificial Intelligence and Statistics*, 2005, pp. 261–268.
[33] C. Fowlkes, S. Belongie, F. Chung, and J. Malik, “Spectral grouping using the nystrom method,” *IEEE Trans. PAMI*, vol. 26, no. 2, pp. 214–225, 2004.
[34] N. Gershenfeld, *The Nature of Mathematical Modeling*. Cambridge University Press, 1999.
[35] J. Shi and J. Malik, “Normalized cuts and image segmentation,” *IEEE Trans. PAMI*, vol. 22, no. 8, pp. 888–905, 2000.
[36] R. Liu and H. Zhang, “Segmentation of 3d meshes through spectral clustering,” in *Proc. Pacific Graphics*, 2004, pp. 298–305.
[37] H. Zhang, O. van Kaick, and R. Dyer, “Spectral methods for mesh processing and analysis,” in *EG State-of-the-art Report*, 2007, pp. 1–22.
[38] V. Jain and H. Zhang, “Robust 3d shape correspondence in the spectral domain,” in *Proc. IEEE Conf. on Shape Modeling and Applications*, 2006, pp. 118–129.
[39] F. Bookstein, “Principal warps: Thin-plate splines and the decomposition of deformations,” *IEEE Trans. PAMI*, vol. 11, no. 6, pp. 567–585, 1989.
[40] S. Belongie, J. Malik, and J. Puzicha, “Shape matching and object recognition using shape contexts,” *IEEE Trans. PAMI*, vol. 24, no. 4, pp. 509–522, 2002.
[41] J. Ham, D. D. Lee, and L. K. Saul, “Semisupervised alignment of manifolds,” in *Proc. Int’l Workshop on Artificial Intelligence and Statistics*, 2005.
[42] K. Siddiqi, J. Zhang, D. Macrini, A. Shokoufandeh, S. Bouix, and S. Dickinson, “Retrieving articulated 3-d models using medial surfaces,” *Machine Vision Applications*, vol. 19, no. 4, pp. 261–275, May 2008.

Gary K.L. Tam received his B.Eng. (first-class honors), M.Phil. and Ph.D. degrees from Hong Kong University of Science and Technology, City University of Hong Kong, and Durham University, respectively. In 2004, he worked as an instructor in the City university of Hong Kong. He is currently a research associate in the Cardiff University, United Kingdom.

Rynson W.H. Lau received a B.Sc. degree from University of Kent and a Ph.D. degree from University of Cambridge. He was on the faculty of Durham University and Hong Kong Polytechnic University. He is now with City University of Hong Kong. Rynson serves on the Editorial Board of Computer Animation and Virtual Worlds, Int’l Journal of Virtual Reality, and IEEE Trans. on Learning Technologies. He has served as the Guest Editor of a number of journal special issues, including *ACM Trans. on Internet Technology*, *IEEE Trans. on Multimedia*, *IEEE Trans. on Vis. and Computer Graphics*, and *IEEE Computer Graphics & Applications*. In addition, he has also served in the committee of a number of conferences, including Program Co-chair of *ACM VRST 2004*, *ICEC 2007*, *ACM MTDL 2009*, *IEEE U-Media 2010*, and Conference Co-chair of *CASA 2005*, *ACM VRST 2005*, *ICWL 2007*, *ACM MDI 2009*, *ACM VRST 2010*.

Project Link:
http://www.cs.cityu.edu.hk/~rynson/projects/retrieval/geomretrieval.html