Deformable Model Retrieval Based on Topological and Geometric Signatures

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

Abstract—With the increasing popularity of 3D applications such as computer games, a lot of 3D geometry models are being created. To encourage sharing and reuse, techniques that support matching and retrieval of these models are emerging. However, only a few of them can handle deformable models, that is, models of different poses, and these methods are generally very slow. In this paper, we present a novel method for efficient matching and retrieval of 3D deformable models. Our research idea stresses using both topological and geometric features at the same time. First, we propose Topological Point Ring (TPR) analysis to locate reliable topological points and rings. Second, we capture both local and global geometric information to characterize each of these topological features. To compare the similarity of two models, we adapt the Earth Mover Distance (EMD) as the distance function and construct an indexing tree to accelerate the retrieval process. We demonstrate the performance of the new method, both in terms of accuracy and speed, through a large number of experiments.

Index Terms—Deformable geometry models, model matching/retrieval, geometry model processing.

1 Introduction

The recent rapid development of computer graphics (CG) animation and 3D games stimulates a dramatic growth in the number of 3D models available on the Internet. Many online 3D repositories store and share hundreds or even thousands of models. Most of them categorize models into groups to facilitate searching. However, because there are discrepancies between text annotation and model content, it may still be difficult for users to locate suitable models. Like other media such as audio, images, and videos, there is a need for an accurate and efficient content-based 3D model search engine. Content-based methods typically comprise two main processes: *feature extraction* and *feature matching*. Feature extraction concerns the use of compact features to represent a model, whereas feature matching computes the similarity of the extracted features using some distance functions. Since feature matching is time consuming when the database is large, some indexing techniques may be applied to speed up the retrieval process.

There is a substantial amount of work devoted to matching and retrieving rigid geometry models efficiently and accurately. Princeton University has developed a search engine [7] where benchmarking is also available [25]. The method presented here, however, targets another type of model: the deformable model, that is, models with similar skeletons but different postures. There are many tools and methods for creating these models. Notable examples include [13] and [24]. To support deformable models, a retrieval system should be able to classify, for example, a sitting human model as similar to a standing one. There are only a few methods proposed to do this. They typically extract graphlike features to represent a model and local geometric features to describe each node of the graph. However, graph matching is computationally expensive.

In this paper, we propose a new matching methodology, emphasizing accuracy and speed. It computes both topological and geometric features to represent a model and uses both features together for feature matching. To accelerate the matching process, we intentionally throw away all skeletal graph information and convert the matching problem to an Earth Mover Distance (EMD) formulation, which measures the energy transfer between two signatures. The contributions of this work can be summarized as follows:

- To avoid using slow graph matching algorithms and to keep the feature size small, we propose to extract topological points and rings as features to represent the 3D models. This representation is compact and can significantly reduce the matching cost.
- To discriminate models with similar or dissimilar skeletons, we stress using both local and global geometric features. These features help discriminate different model groups that have similar skeletons, like girls and babies. (We refer to these models as *similar-skeleton models* hereafter.) Existing methods use only local geometric features for matching. They work best to discriminate model groups with dissimilar skeletons, like dogs and men. (We refer to these models as *dissimilar-skeleton models* hereafter.)
- We formulate the feature-matching problem as a flow and transportation problem and propose a new similarity measure based on EMD. As this similarity measure is a metric, we can build a vantage point (VP)-tree to answer retrieval queries efficiently by k-nearest

• *The authors are with the Department of Computer Science, University of Durham, E-Science Building, Science Laboratories, South Road, Durham, County Durham, DH1 3LE, United Kingdom.*
E-mail: {g.k.l.tam, rynson.lau}@durham.ac.uk.

Manuscript received 8 Aug. 2005; revised 11 June 2006; accepted 1 Nov. 2006; published online 8 Jan. 2007.
For information on obtaining reprints of this article, please send e-mail to: tvcg@computer.org, and reference IEEECS Log Number TVCG-0102-0805.
Digital Object Identifier no. 10.1109/TVCG.2007.1011.

neighbor search. To the best of our knowledge, our indexing scheme is the first to support matching of both topological and geometric features in one pass.

The rest of this paper is organized as follows: Section 2 summarizes the existing 3D model matching and retrieval techniques. Section 3 presents an overview of our method. Sections 4 and 5 discuss how we extract topological and geometric features, respectively. Section 6 presents our feature-matching scheme. Section 7 evaluates the performance of our method through a number of experiments. Section 8 briefly concludes this paper.

2 Related Work

Classical 3D retrieval methods can be categorized into four approaches: geometry-based, transform-based, image-based, and topology-based. The first three methods can only handle nondeformable models, whereas the fourth can handle deformable models.

2.1 Methods for Nondeformable Geometry Models

Geometry, transform, and image-based approaches focus on retrieving nondeformable 3D models only. The geometry-based approach concerns properties related to the shape and size of a model. In general, methods of this approach can be classified into three types: methods based on extracting physical properties [12], [5], methods based on computing histograms or some distribution functions [18], [20], [10], and methods based on computing energy for morphing a model [2], [28], [30]. The transform-based approach analyzes 3D models in a different feature domain. Transformation functions used include Fourier Transform [29], Wavelets Transform [21], and Zernike Transform [17]. Funkhouser et al. [7], Kazhdan et al. [11], and Novotni and Klein [17] propose Spherical Harmonic for extracting rotation-invariant features. The image-based approach captures features from 2D image views of a 3D model [1], [19].

Generally, the geometry-based approach is efficient and easy to implement, but its matching accuracy is usually lower than the other two approaches. The transform-based approach has several advantages such as supporting multiresolution analysis and having improved accuracy with the recent development in concentric spherical harmonic. A major advantage of the image-based approach is its independence from 3D data representation. However, it typically has a large feature size and, hence, high matching cost.

2.2 Methods for Deformable Geometry Models

The topology-based approach is the only approach that supports matching of deformable models through analyzing the model with skeletal or topological information. As this approach is the focus of our work, we discuss these methods in detail here. In [9], the Multiresolution Reeb Graph (MRG) is proposed. It first partitions a model into nodes using integral geodesic at different resolutions. Unlike Euclidean distance, geodesic measures distances on the surface and is not affected by model deformation. Thus, integral geodesic indicates how far a point is from the surface center. (We discuss this further in Section 4.1.) MRG then constructs an MRG tree by analyzing the adjacency of each node in the current and lower/higher resolutions. In each node, it uses area and length as geometric features. To match two MRG trees, a heuristic graph-matching algorithm is applied in a coarse-to-fine manner, starting from the root nodes of the two trees and traversing down the trees following the child nodes with maximum similarity. When all high-resolution nodes are exhausted, the matching process traces back to the lowest resolution nodes again. All similarity values computed are added up as the final similarity value.

In [23], a voxel thinning method is proposed to extract the skeleton from a voxelized model. In each skeletal node, the radial distribution of edges is preserved for local shape matching. To speed up the query process, a topological feature vector is generated for each skeletal graph as an index to the database. Nearest neighbor search is then applied for model retrieval. To further verify the correctness of the retrieved models, an enhanced maximum cardinality minimum weight bipartite matching algorithm is used. Instead of using the skeletal graph, [26] analyzes models based on the component graph. A model is first split by mesh decomposition with each component node described by one primitive. An optimal error-correcting subgraph isomorphism algorithm is then applied to match two component graphs.

In [8], a pose-oblivious shape signature is proposed, which supports deformable models. The shape descriptor is a 2D histogram of two functions: local diameter and integral geodesic (centricity). Experiments showed that the new method could produce interesting retrieval results.

In summary, topology-based methods handle deformable models using skeletal information. However, several research issues have still not been explored yet. First, although most of these methods work well in discriminating *dissimilar-skeleton* models, none of them consider the issue of discriminating *similar-skeleton* models as they use only local geometric features. Second, due to the large feature size or the use of slow graph matching techniques [26], these methods are generally slow in practice and do not scale well to large databases. Third, although [23] proposes an indexing scheme for large databases, it may still suffer from the accuracy problem when answering nearest neighbor queries as it separates topological matching and geometric matching into two processes. To improve the recall rate, it needs to return a large number of models in the first pass, causing a performance penalty to the indexing scheme.

To address these problems, we proposed in [27] to use both topological and geometric features simultaneously. It extracts bounded regions as topological features using saddle critical points from two source points. As will be discussed later, these critical points may be unstable or extraneous, leading to stability problems. Here, we improve the work significantly to address this problem by proposing to use topological points and rings as features. To support fast retrieval, we further extend the matching algorithm to support indexing search with both topological and geometric features in a single pass.

3 Method Overview

The focus of our method is on the use of both topological and geometric features at the same time. We extract two types of topological features: *topological points* and *topological rings*. A topological point is defined as the salient point located at a protrusion tip, and a topological ring is defined as the border that separates two significant components in a model.

To capture topological points to represent protrusion tips, we first derive our algorithm from a skeleton extraction technique Level Set Diagram (LSD) [14]. However, LSD suffers from two problems: extraneous critical points [16] and slicing direction [15]. To alleviate these problems while remaining fast and automatic, we propose two solutions: *Critical Point Analysis* and *Topological Point Selection*, which are described in detail in Sections 4.3 and 4.4, respectively. These two steps produce the validated maximum critical points, referred to as topological points. To reduce computation time, we further discuss how we select the minimum critical points (source points) in Section 4.5. Our method also use topological rings as features, which are first discussed in [15]. However, this method has its own limitations that make it impractical for our use. To extract reliable topological rings to represent joint locations, we propose *Topological Ring Extraction* in Section 4.6. We name our approach for extracting both topological points and rings as *Topological Point Ring* (TPR) analysis.

After obtaining all topological features (points and rings), we extract geometric features to characterize each of them. There are two kinds of geometric features in our method: local and global features. Local features include normalized integral geodesic and effective area. They are used to characterize the locations and importance of a topological feature, as will be discussed in Sections 5.1 and 5.2, respectively. Global geometric features are used to capture the surface information of a model. They help discriminate *similar-skeleton* models like girls and babies, as will be discussed in Section 5.3. Hence, our model signature is defined by a collection of topological features and each of them is characterized by a number of geometric features.

We formulate the matching of two models as the energy transfer between two signatures by adapting the EMD, which computes the minimum energy required to transform one signature into another. We define our metric distance function for the EMD matching framework in Section 6. Since the function is a metric, we can construct a fast indexing scheme by building a VP-tree. Such an indexing scheme can support both topological and geometric nearest neighbor searches in one pass.

4 Topological Point Ring (TPR) Analysis

In this section, we first briefly introduce integral geodesic and LSD, which are fundamental to our method. We then discuss the two problems of using LSD to extract maximum critical points as the topological points and propose our methods to tackle the two problems. Finally, we discuss how topological rings can be extracted reliably based on these topological points.

Fig. 1. Integral geodesic on a surface. The brighter region is closer to the surface center, whereas the darker regions are farther away from the surface center.

4.1 Integral Geodesic

Geodesic and integral geodesic are basic to our method. As mentioned earlier, geodesic is the shortest distance between two points on a surface. Hilaga et al. [9] first suggest the use of integral geodesic, which is defined on a surface as $G(q) = \int_{p \in S} g_q(p) \partial S$. Given a vertex q, integral geodesic is the integral of all geodesics g measured from q to all vertices p on a surface S. In general, integral geodesic gives a small scalar value if vertex q is near to the center of the mesh (brighter region in Fig. 1) and a larger scalar value if q is located away from the center (darker region in Fig. 1). Hence, integral geodesic indicates how far a vertex is from the points that have minimum integral geodesic. Note that a point with minimum integral geodesic is not the *center of mass* of the model; the *center of mass* can be considered as the minimum integral Euclidean distance of a point set. In the following discussion, we refer to the points having minimum integral geodesic as the *surface centers*. Fig. 1 shows the integral geodesic of a surface.

4.2 Level Set Diagram

The LSD [14], which is based on the Morse theory, is a skeleton extraction technique. The Morse theory describes how the differential geometry of a surface algebraically relates to the topology. LSD applies the theory on polyhedral surfaces to extract the skeletons. It uses geodesics as the Morse function to build a scalar field on the surface, which is then used to extract three kinds of critical points (minima, saddles, and maxima) by tracing the geodesic wavefront. A geodesic wavefront originates from a minimum point, meets itself at a saddle point, and closes at a maximum point. These critical points form an LSD tree with the minimum, saddle, and maximum points forming the root, the internal nodes, and the leaf nodes of the tree, respectively. LSD also defines a level set $C(l_s)$ as a polygonal contour of the same level l_s on the surface, where l_s is a scalar field value and s is the origin of the field. An edge (v_i, v_j) is called a cross-edge if it passes through level set l_s satisfying the condition: $g_s(v_i) < l_s < g_s(v_j)$, where $g_s(v)$ is the scalar field at vertex v obtained by calculating the geodesic from a minimum (source) point s. LSD uses the Dijsktra algorithm to approximate the geodesic distance.

4.3 Critical Point Analysis

As mentioned, our topological features are composed of topological points and rings. We choose the maximum critical points from LSD as the topological points because the location of maximum critical points matches the definition of topological points discussed in Section 3.

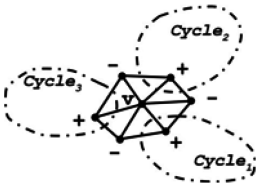


Fig. 2. $C(l_s)^+$ and its three cycles, which are shown as dotted lines.

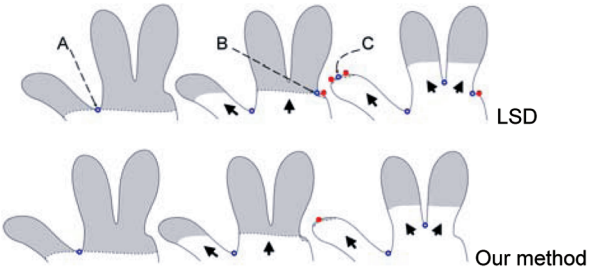
Though it is easy to apply LSD, there are two problems that hinder us from using it directly here: extraneous critical points and slicing direction.

Shortest path algorithms usually suffer from getting extraneous critical points when they are applied on meshes. As LSD is based on computing shortest path distances (geodesics), it also suffers from this problem. According to [16], extraneous critical points may result from noise, precision errors, or the fact that geodesic distance is not a good Morse function. Though [16] provides a method to find the optimal number of critical points, it is not fully automatic, making it less suitable for use in a 3D model search engine. To alleviate this problem, we propose an efficient method based on LSD. We observe that extraneous points arise during the registration of saddles, producing critical points that are very close to each other. As such, we apply a proximity-filtering step before a saddle is registered. We refer to the modified LSD as Critical Point Analysis.

Before we register a vertex v as a saddle in LSD, we approximate the level set $C(l_s)$ by defining a vertex set $C(l_s)^+ = \cup v_j$ such that $l_s < g_s(v_j)$ for all cross-edges (v_i, v_j) related to $l_s = g_s(v)$. The set contains the unvisited vertices adjacent to the geodesic wavefront in LSD. It can be obtained by a depth first search algorithm on the Dijsktra heap. We do not compute $C(l_s)$ because of speed concerns. As v is a saddle candidate, the geodesic wavefront meets itself at v; so, the vertex set $C(l_s)^+$ can be split into a collection of cycles depending on the sign change around v, $C(l_s)^+ = Cycle_1 \cup Cycle_2 \ldots \cup Cycle_n$. Fig. 2 shows an example where $C(l_s)^+$ has three cycles.

TABLE 1
Summary of Critical Point Analysis during Saddle Registration

No. of valid cycles	Explanation	Action
> 1	Branching occurs at the current vertex if there is more than 1 valid cycle. This is a normal case in LSD (case A of Fig 3).	Register as normal saddle.
1	No branching actually occurs – a degenerate case. If the vertex in the cycle is ignored, LSD does not consider the current vertex as a critical point (case B of Fig 3).	Disallow saddle registration. Disallow all vertices in $C(l_s)^+$ to become critical points.
0	No branching occurs – not a good saddle. However, it may be a good maximum critical point as all the surrounding vertices have scalar field less than the current vertex if vertices in $C(l_s)^+$ are ignored (case C of Fig 3).	Register the saddle candidate as maximum critical point and disallow all vertices in $C(l_s)^+$ to become critical points.



White / Shaded Area: Visited / unvisited region
Dashed Line: Cycles under consideration
Frontier between White and Shaded Areas: Geodesic Wavefront

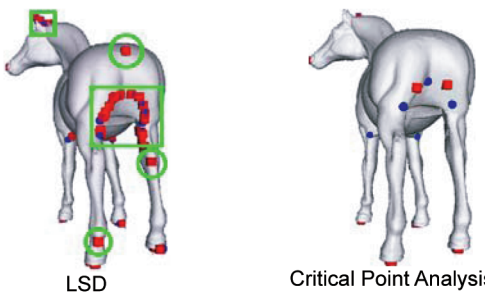
Fig. 3. Selection of saddles in Critical Point Analysis.

In our filtering step, we consider a cycle, $Cycle_i$, as valid if and only if the number of vertices inside is greater than a number n, where n indicates how strong the filtering step is. If the number is small, it allows a smaller distance between adjacent critical points. In our case, we choose $n = 1$ because we do not want to miss small features for our feature-matching step. Then, by considering the number of valid cycles around a saddle candidate, we further decide if the registration is a success or not. Table 1 and Fig. 3 explain the considerations and subsequent actions. Fig. 4 shows an example to compare LSD and *Critical Point Analysis*. The two diagrams are generated using geodesic as the scalar field.

4.4 Topological Point Selection

Another problem of LSD is the slicing direction problem. When a different source point is chosen, a different set of critical points may be extracted [15]. Consider Fig. 4; LSD identifies a source point at the horse nose and extracts many critical points from the back of the model. However, if the source point were located at the bottom of one of the rear legs, LSD would extract critical points from the front neck of the horse. To tackle this stability problem, we introduce the idea of using multiple source points and derive *Topological Point Selection* as a process for selecting valid topological points.

Suppose that we have n source points, $s_1 \ldots s_n$, on the surface. We apply *Critical Point Analysis* on each of these n source points to obtain n critical points sets, each consisting of three kinds of critical points: $CriticalSet_{s_i} = Max_i \cup Saddle_i \cup Min_i$. Max_i and $Saddle_i$ are the sets of maximum



Cube: Maximum or minimum
Sphere: Saddle
Green Region: Many critical points found close to each other

Fig. 4. Comparison of LSD and our method.

and saddle points, respectively. The minimum point $Min_i = s_i$ is the source point.

Our *Topological Point Selection* then identifies valid topological points by counting the number of different maximum points in a region. First, for each maximum point m in Max_i, we search for the number of maximum critical points nearby. The search radius is set to the geodesic distance measuring from point m to the nearest saddle which may exist in any $Saddle_i$. If the number of maximum points in the search region is higher than $n/2$, that is, more than half of the source points produce maximum points in the region, we consider it as a valid maximum point. We repeat this for all unvisited m in Max_i, $i = 1 \ldots n$. Since there may be many maximum critical points in a region, we arbitrarily choose one of them as the topological point. Note that the actual location of a topological point is not important as it is used for feature representation only.

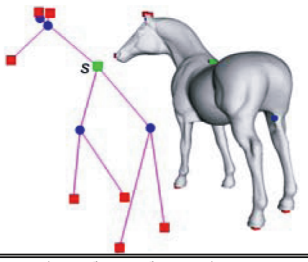
4.5 Source Point Selection

To perform Topological Point Selection, a number of source points must be selected. We have found from experiments that three source points, if selected appropriately, are sufficient to identify all valid topological points with minimal computational cost. We choose the two furthest points in a 3D mesh as the source points. The first point can be found by running Dijkstra on an arbitrary point on the model to obtain the point with maximum geodesic. The second point can then be determined by applying Dijkstra again on the first point to obtain another point with maximum geodesic [14]. However, since they are located at the far end of a model, LSD may still favor a particular direction and miss some critical points. Thus, we choose the third source point near the center of a mesh. A good choice is the surface center that we have discussed earlier in Section 4.1.

In [9], an approximation method is proposed to find the surface center, but it is too slow for a search engine as it samples at least 120 points on the surface. We observe that most deformable models are not perfectly symmetric after deformation and, usually, there is only one point that corresponds to the minimum integral geodesic. To speed up the process, we apply a hierarchical search to locate the point. (Note that our method still works even if there is more than one such point in a mesh as we only need to find a reference point.) In our hierarchical search, we first split the surface into many patches. For each patch, we calculate the integral geodesic at the patch center. We then identify the patch with the smallest integral geodesic. We split it again into many subpatches and calculate the integral geodesic at each subpatch center. We apply this strategy recursively until the change in the smallest integral geodesic is less than a threshold or the patch area is too small. Fig. 5 shows the topological points obtained from Topological Point Selection and the LSD tree constructed using Critical Point Analysis with the surface center s being the source point.

4.6 Topological Ring Extraction

The term "Topological Ring" is first mentioned in [15]. Given some initial points, the method applies topological expansion of a 1-ring neighborhood. When frontiers of different topological expansions collide, a branching is identified. Mortara and Patane [15] defines the borders of these frontiers as topological rings. However, as its objective is on skeleton extraction, the topological rings produced are not suitable to be used as features here for two reasons. First, the topological expansion in [15] depends on a 1-ring neighborhood only. To extract topological rings reliably for use as features, a regular tessellation of the mesh surface is required. Second, the method processes all source points at the same time and, thus, the locations of topological rings greatly depend on the differences in branch lengths. For general 3D models, these two requirements may be difficult to satisfy. Hence, the method cannot ensure consistent recovery of topological rings.



Cube: Maximum (topological point) or minimum
Sphere: Saddle **S:** Source Point
Link: Relationship between critical points

Fig. 5. Selection of topological points.

Here, we propose our Topological Ring Extraction to address this problem. Our method features a multisource point approach. It is modified from the Dijkstra algorithm, which approximates geodesic distances. Since our topological expansion is based on shortest path growing instead of a 1-ring neighborhood, it does not require regular tessellation of the mesh surface. In addition, different source points are given different initial values. This allows more stable extraction of topological rings with no regard for branch lengths. Our method comprises three stages: initialization, shortest path algorithm, and termination. The initialization stage defines different initial values for different topological points. Our shortest path algorithm then traces the geodesic wavefronts from these topological points using the corresponding initial values. When executing the algorithm, different wavefronts merge and new wavefronts are formed, which become the topological rings. When all frontiers merge into one, the algorithm terminates. The details of these three stages are discussed as follows.

4.6.1 Initialization

In this stage, we determine an initial value for each topological point such that topological rings may collide near joint locations. We assign a smaller initial value to a topological point that is far away from the mesh center and a larger value to one that is near the mesh center. This initial value is the starting time of the shortest path algorithm for that point. Since all topological points are located far away at protrusion tips, these initial values ensure that the shortest path algorithm is always moving toward the center. We use the *surface center* to approximate the mesh center. Because integral geodesic is a good measure of the relative

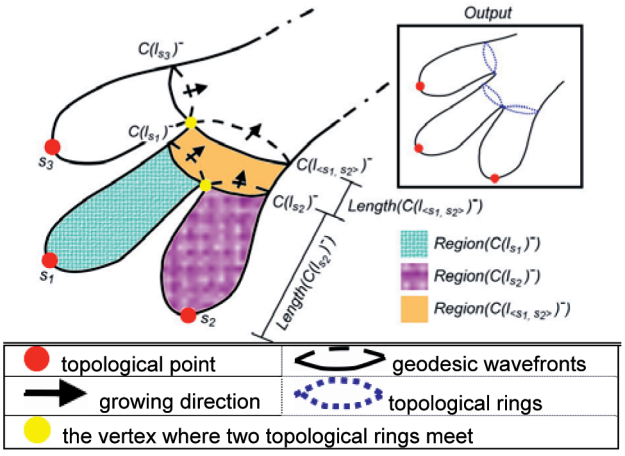


Fig. 6. Shortest path growing and validation with three topological points.

distance from the *surface center*, o, we compute the initial value of a topological point m based on the following:

$$MAX_{q\in S}g_o(q) \times \frac{MAX_{q\in S}G(q) - G(m)}{MAX_{q\in S}G(q) - MIN_{q\in S}G(q)}. \quad (1)$$

This initial value of m is calculated by the maximum geodesic $MAX_{q\in S}g_o(q)$ measured from o, which is the longest possible distance that the shortest path algorithm travels and is interpolated by integral geodesic $G(m)$.

4.6.2 The Shortest Path Algorithm

After defining the initial values for the topological points, we put these points in the heap of a Dijsktra algorithm. By using Dijsktra, the topological expansion is, in effect, a shortest path algorithm. During the execution, the vertex with the smallest initial value is removed from the heap one at a time and its neighbors are updated. Hence, we may consider that different geodesic wavefronts grow from different topological points and are moving toward the mesh center. When executing the algorithm, if two geodesic wavefronts meet, a merge of geodesic wavefronts occurs and a new wavefront is formed.

Without loss of generality, we consider here a typical case where two geodesic wavefronts meet at vertex v. We define two vertex sets $C(l_{s_a})^-$ and $C(l_{s_b})^-$ to represent them, where l_{s_a} and l_{s_b} are the two level sets that originated from two distinct source points s_a and s_b. If we let $g_{s_a}(v)$ and $g_{s_b}(v)$ be the scalar fields that originated from s_a and s_b, respectively, we have $g_{s_a}(v) = g_{s_b}(v)$ at vertex v. We also define $C(l_s)^- = \bigcup v_i$ such that $l_s > g_s(v_i)$ for all cross-edges (v_i, v_j) related to l_s, where $l_s = g_s(v)$ and s is the source point. $C(l_s)^-$ refers to the vertices that form the geodesic wavefronts at v and are stored in the heap of the Dijsktra algorithm. We further define $C(l_{<s_a,s_b>})^-$ as the resulting geodesic wavefront of merging $C(l_{s_a})^-$ and $C(l_{s_b})^-$ as shown in Fig. 6. Note that, in practice, $g_{s_a}(v)$ may differ slightly from $g_{s_b}(v)$ at v as the initial values are only approximations. However, the small difference will not affect the shortest path algorithm.

We mentioned earlier that topological rings are registered when geodesic wavefronts meet. However, redundant topological rings may sometimes be created. For example, in Fig. 6, we would expect to have three rings located at the three joints between the arm and the fingers as shown inside the square box. Hence, $C(l_{<s_1,s_2>})^-$ is redundant. To detect this case, we check if a geodesic wavefront is a valid topological ring by measuring the length of the region formed. We first define *Region*() and *Length*() as follows:

$$Region(C(l_{<s_a,s_b>})^-) = \left\{ q \,\middle|\, \begin{matrix} g_{s_a}(q) < l_{<s_a,s_b>} \\ or \\ g_{s_b}(q) < l_{<s_a,s_b>} \end{matrix} \right\} - Region(C(l_{s_a})^-) - Region(C(l_{s_b})^-), \quad (2)$$

where $Region(C(l_s)^-) = \{q | g_s(q) < l_s\}$ for all q in S. Here, we define a region associated with a geodesic wavefront as all the vertices that have scalar field g less than the level set at the wavefront except those already registered in other regions. The scalar field should be measured according to all the possible source points s_a and s_b. Since a wavefront may be originated from a merge of two rings, (2) is defined as a recursive function. The length of a region is defined as the minimum difference between the level set values of the current geodesic wavefront and each of the originating topological rings:

$$Length(C(l_{<s_a,s_b>})^-) = Minimum\{(l_{<s_a,s_b>} - l_{s_a}), (l_{<s_a,s_b>} - l_{s_b})\}. \quad (3)$$

To check whether a geodesic wavefront is a valid topological ring, we simply check if the length of the region is longer than a given threshold. All valid geodesic wavefronts are marked as topological rings.

4.6.3 Termination of the Algorithm

The algorithm terminates when all geodesic wavefronts have merged into one. However, we note that, when we have visited all saddle points obtained from Critical Point Analysis using the surface center as the source point, there should be no more branching left and the algorithm may end. The unvisited vertices or vertices that are visited but not yet included in any region are grouped into one final region (FR). Fig. 7 shows some example models with various topological rings extracted using our algorithm.

5 Geometric Feature Extraction

After TPR analysis, we obtain a set of topological points and rings together with a set of regions. These topological points and rings are located at protrusion tips and articulated joints. They provide skeletal information of the model, independent of model deformation. We characterize each of these topological features with three types of geometric information: Normalized Integral Geodesic, Effective Area, and Geometric Surface Vector.

5.1 Normalized Integral Geodesic—Spatial Information

Integral geodesic is a function defined over the surface to indicate how far a point is from the surface center. It maps a point to a scalar value and is thus a good feature to describe the spatial location of a topological point. To generalize this

Fig. 7. Topological rings (borders between different colored regions).

function to any topological feature (point or ring), we need to consider the case of the topological ring as well. To compute the integral geodesic for a ring, we interpolate the value from the integral geodesic of the surface center and the value of one of the originating topological points of the ring. Since a ring may come from many originating topological points, we use the one that is furthest away from the ring, which is the ancestor topological point as its distance from each vertex on the ring has a smaller deviation. The interpolation requires two distance values: from the ring to the surface center and to the ancestor topological point. To compute these distances, we use geodesics with respect to a vertex set (vs). Such distance, which is denoted as g_{vs}, can be calculated by Dijsktra with all the vertices in the ring as source points. The generalized Integral Geodesic $G'(U)$ of topological feature U is as follows:

$$G'(U) = \begin{cases} \text{if } U \text{ is a topological point : } \\ \quad G(U) \\ \text{if } U \text{ is a topological ring : } \\ \quad \frac{g_{vs}(o,U)\times G(w)+g_{vs}(w,U)\times G(o)}{g_{vs}(o,U)+g_{vs}(w,U)}, \end{cases} \tag{4}$$

where $G(v)$ is the integral geodesic of point v. $g_{vs}(o,U)$ and $g_{vs}(w,U)$ are the shortest distances measured from topological ring U to surface center o and to ancestor topological point w, respectively. Finally, we calculate the Normalized Integral Geodesic as follows:

$$G'_{norm}(U) = \frac{G'(U) - MIN_{q\in S}G(q)}{MAX_{q\in S}G(q) - MIN_{q\in S}G(q)}. \tag{5}$$

5.2 Effective Area—Weights of Importance

We note that the importance of a topological ring located in a finger, for example, should be smaller than that located in the leg. This is intuitive as removing a leg from a 3D model gives a larger perceptual impact than removing a finger. Hence, we approximate the importance of topological features by distributing the local surface areas among the adjacent topological features. We denote such a redistributed area as the Effective Area.

Fig. 8. Models divided into 20 bands from topological rings (dashed lines).

To simplify our discussion, we first define some abbreviations. A Protrusion Region (**PR**) is a region bounded by a topological point and a topological ring. A Segment Region (**SR**) is a region bounded by topological rings only. An **FR**, as mentioned in Section 4.6.3, is the final extracted region. We consider two cases in our method:

- ***PR***. Simply divide the **PR** surface area into two and associate half to the point and half to the ring.
- ***SR and FR***. Distribute the local surface area to the adjacent topological rings in proportion to $Region(C(l_s)^-)$ computed for each adjacent ring $C(l_s)^-$.

Note that:

$$\begin{aligned} &\sum EffectArea(U) \\ &= \frac{\sum Area(PR) + \sum Area(SR) + Area(FR)}{Area(S)} = 1. \end{aligned} \tag{6}$$

5.3 Geometric Surface Vector—Surface Distribution

In order to better discriminate *similar-skeleton* models, we capture additional geometric information to describe the global surface change. We consider three global geometric features: curvature, area, and average distance. We construct three vectors from these features by first dividing the model into many bands according to their geodesic distances from a given topological feature. Again, since geodesic is calculated on the surface, the resulting feature vector is stable toward mesh deformation. As an example, we divide two dog models shown in Fig. 8 into geodesic bands relative to a topological ring located at one of the legs. Bands of the same color indicate that they are within the same geodesic interval from the ring. We can see that, although the two dogs have different poses, the locations of the color bands are similar.

Sometimes, a single band may be composed of several segments. For example, some color bands in Fig. 8 may have segments at different locations like the body, the limbs, and the tail. We apply depth first search to locate all segments of each band. We then use the surface area of each segment to compute the weighted average to be the final value of that band. In our implementation, we use Gaussian curvature to be the curvature information. The average distance of a segment is defined as the average Euclidean distance from its center of mass to all vertices in the segment. In general, area and curvature are used to capture the global surface change, whereas average distance measures the thickness of individual segments.

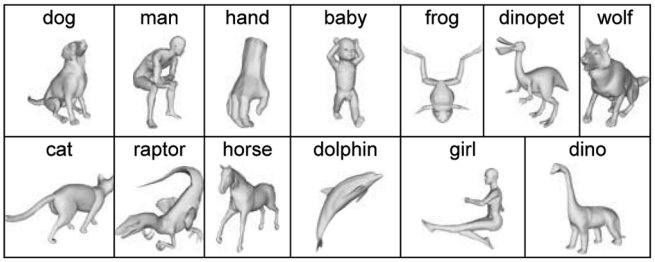


Fig. 9. The 13 model groups in our model database.

6 Feature Matching

With a signature for each model, that is, a set of topological features (points or rings) each described by three types of geometric features, we may compare the similarity of different models based on matching the signatures.

6.1 Simliarity Measure

To compare the similarity of two signatures, we propose to use EMD [22] as a distance measure. EMD computes the minimum energy required to transform one point set to another. Each point in the set has a mass (weight) and requires some energy to transfer to another location. In our approach, we consider a topological feature as an EMD point and define Effective Area as weight. To describe the energy transfer between two EMD points, we further define a distance function $Dist()$ based on geometric features as follows:

$$\begin{aligned} Dist(U_1, U_2) = {} & W_1 \times |G'_{norm}(U_1) - G'_{norm}(U_2)| \\ & + W_2 \times L_{2,norm}(K(U_1), K(U_2)) \\ & + W_3 \times L_{2,norm}(A(U_1), A(U_2)) \\ & + W_4 \times L_{2,norm}(H(U_1), H(U_2)), \end{aligned} \tag{7}$$

where G'_{norm} is the Normalized Integral Geodesic. K, A, and H are the geometric surface vectors representing curvature, area, and average distance, respectively, and they implicitly capture different branch arrangements relative to a topological feature. Hence, G'_{norm}, K, A, and H together describe the spatial location of the topological feature. W_1, W_2, W_3, and W_4 are ratios such that $W_1 + W_2 + W_3 + W_4 = 1$. We use these weights to adjust the relative importance of G'_{norm}, K, A, and H. By using EMD, we can now avoid slow graph matching algorithms by converting the matching problem to a flow and transportation problem.

6.2 Indexing Scheme

In content-based retrieval, we typically expect the system to return a given number of objects that are most similar to a given query—k-nearest neighbor search (k-NNS). To speed up the retrieval process, it is common to apply an indexing scheme. Here, we discuss how our similarity measure can be used for indexing retrieval.

Since our model signature is not a single k-dimensional vector, but a set of topological features characterized by many geometric features, we need to use a distance-based indexing method. We use a VP-tree as it is such a method and has been reported to have good performance over others [6]. A VP-tree stores all the indexing data points in a tree. By comparing from the root nodes and level by level down the tree, it may return the nearest neighbors of a given query. A distance-based indexing method generally requires a distance function that satisfies metric properties. Our method is based on the EMD framework, which can be proven a true metric if it satisfies the following properties under EMD formulation [22]:

- The sum of all feature weightings for each model should be the same.
- The ground distance function used by EMD must be a metric.

Our algorithm satisfies the first property because we use the normalized *Effective Area* as the weight, as shown in (6). It is also clear that $Dist()$, shown in (7), is a metric because it only makes use of ratio combination of metric functions.

TABLE 2
Mean Similarity of Dissimilar-Skeleton Models

	man	frog	dolphin
man	1	0.414	0.002
frog	0.414	1	0.008
dolphin	0.002	0.008	1

7 Experimental Results

To evaluate the performance of the proposed retrieval method for deformable models, we discuss a number of experiments here. We have constructed a database from 150 models for these experiments. To test the invariant properties of our method in rotation and scaling, we create three additional sets by rotating the 150 models against the xy-axis, random scaling between (1.0, 2.0], and rotating by the yz-axis plus random scaling to produce a total of 600 models. We then manually categorize these models into 13 groups as shown in Fig. 9. Each group consists of similar models but at different postures. All the experiments presented here are performed on a PC with a Pentium 4 2.4-GHz CPU and 1-Gbyte RAM.

7.1 Performance on Model Discrimination

Tables 2 and 3 show some matching results, using all models and normalized by maximum and minimum work done. We can see that our method can distinguish models based on their skeletons and shapes. In Table 2, man, frog, and dolphin have dissimilar skeletons. Our method can discriminate them as the similarity values among different model groups are relatively small. In Table 3, man, girl, and baby are model groups with similar skeletons. Our method again can discriminate them as the similarity values among different model groups, but are still comparatively small;

TABLE 3
Mean Similarity of Similar-Skeleton Models

	man	girl	baby
man	1	0.733	0.553
girl	0.733	1	0.458
baby	0.553	0.458	1

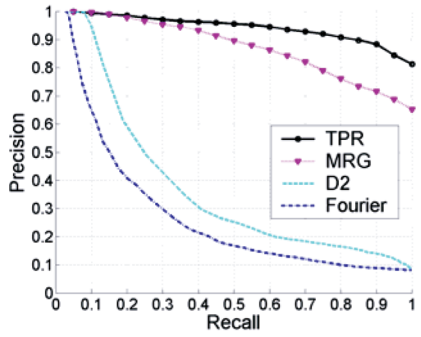


Fig. 10. Performance comparison of TPR with other methods.

although, they are slightly larger than those in Table 2. These two sets of results match human intuition well.

7.2 Performance Comparison with Nontopology-Based Methods

Fig. 10 shows the precision-recall graph of our method compared with some nontopology-based methods. It shows that our method outperforms the geometry-based D2 method [20] (feature size: 72) and the transform-based Fourier method [29] (feature size: 21). From the plot, we may conclude that our method is capable of handling deformable models, whereas the D2 and Fourier methods are incapable as their precisions drop dramatically when the recalls rise over 0.1.

7.3 Performance Comparison with MRG (Topology-Based Method)

Topology-based methods are generally difficult to implement as they require specific skeletal tree construction and graph matching techniques. To study the performance of the new method as compared to existing ones, we have implemented the MRG method [9] for comparison. We have chosen to implement MRG because both MRG and TPR make use of geodesic distance, and we have all the necessary information to reimplement MRG. In our experiments, we compare the performance of TPR with MRG in terms of accuracy and speed.

7.3.1 Accuracy Comparison

The similarity matrix can be used to serve as a measure of matching performance among topology-based methods. Because of the large number of models in our database and due to page limitation, Fig. 11 only shows the mean similarity matrices of TPR and MRG. Each cell in the matrix indicates the similarity of two model groups—the darker a cell is, the higher the similarity of the two corresponding groups. The similarity values shown are normalized by the maximum and minimum values of the corresponding groups.

From the similarity matrices, we have the following observations:

1. The diagonal lines of both matrices give the darkest color. This means that both TPR and MRG perform equally well in identifying models of the same group.

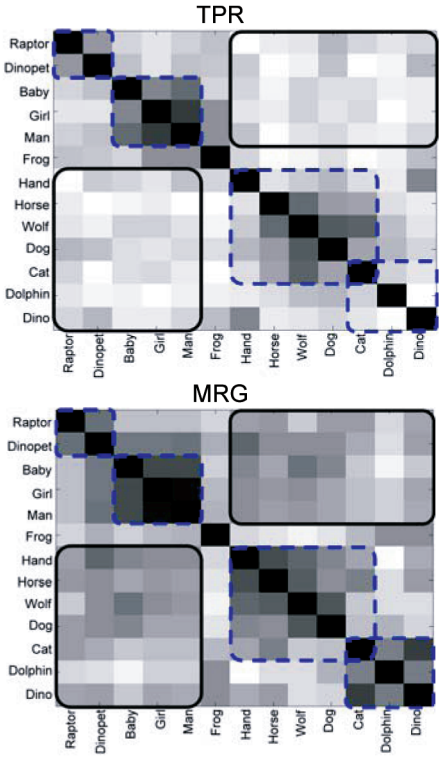


Fig. 11. Mean similarity matrices for TPR and MRG.

2. TPR gives lower similarity values than MRG in the two regions circled by solid lines. These regions show the similarity among different model groups with dissimilar skeletons. This means that TPR can discriminate dissimilar-skeleton models better.
3. TPR also gives lower similarity values in the four regions circled by dashed lines. These regions show the similarity among different model groups with similar skeletons. We can see that TPR can discriminate hand from horse, wolf, dog, and cat with higher similarity contrast. Although MRG considers cat to be similar to dolphin and dino, TPR can discriminate them better and considers cat to be similar to dog and wolf, which matches human inspection. This means that TPR can discriminate similar-skeleton models better.

To explain these observations, we may analyze the features used by TPR and MRG. First, though our database contains many models of different postures, both methods can identify models of the same groups with highest similarity. This can be explained by the fact that they are both topology-based methods.

Second, TPR gives a higher similarity contrast than MRG to dissimilar-skeleton models. This can be explained by the use of topological and geometric features in TPR. In MRG, the models are partitioned into regular intervals and matched. Since intervals are regular, it cannot tell how important a branch or limb is when compared with others. To match two nodes, only local geometric features like area and length are used in MRG.

In our third observation, TPR can discriminate similar-skeleton models better. For MRG, the similarity measure is

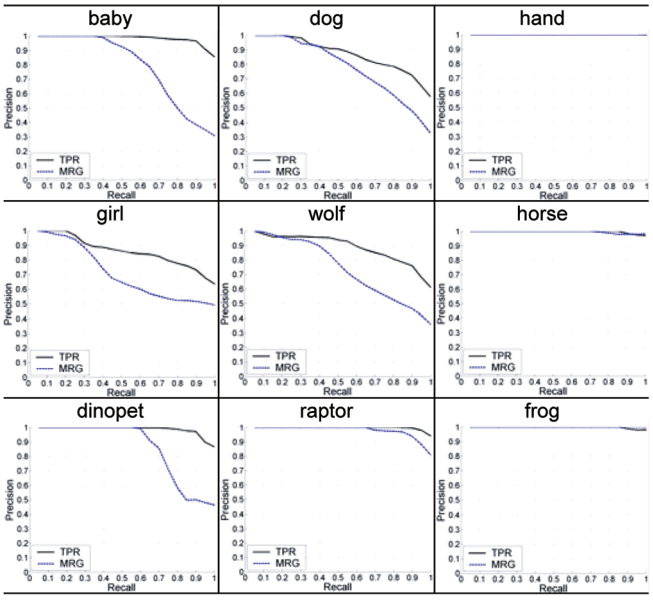


Fig. 12. Precision-recall graphs for some of the model groups.

based on matching multiresolution reeb graphs. With similar-skeleton models, the differences can only be captured at the lowest level of the graphs using local geometric features, area, and length. These features are local and do not represent the overall shape of the model. As a result, MRG is less effective on these models. Although TPR represents similar-skeleton models by the same number of topological features, it captures the overall shape of the models by the global geometric features and weights. For example, dog, wolf, cat, horse, and dino are four-legged animals with tails, dolphin has four side fins and a back fin, and the hand has five fingers. It is difficult to discriminate them if we consider only topological and local geometry information, as in MRG.

Although the mean similarity matrix shows the relative similarity among model groups, it cannot show the actual retrieval performance. If the range of maximum and minimum similarity values within one particular model group overlaps with those of the others, the precision-recall values may drop due to the increase in mismatches. This is a general phenomenon when the database contains many model groups. To compare the actual retrieval performance of TPR with MRG, we have plotted their mean precision-recall values. In Fig. 10, we can see that TPR and MRG have similar performance when recall is below 0.3. When recall is above 0.3, the precision of MRG begins to drop more significantly compared with that of TPR. To further compare the performance of the two methods, we have also plotted the precision-recall graph for each model group. Fig. 12 shows some of them. We observe that TPR performs better than MRG for model groups like dog, wolf, raptor, dinopet, baby, girl, man, dino, and dolphin or equally well as MRG for model groups like hand, frog, cat, and horse.

From these graphs, we notice that TPR outperforms MRG particularly on similar-skeleton models such as canines (dog, wolf), cannibal dinosaurs (raptors, dinopet), and humans (baby, girl, man). These models have similar skeletons that cause MRG to drop in performance. By also considering the global geometric features, TPR performs

TABLE 4
Complexity Analysis of TPR

Topological Feature Extraction	
Selection of Source Points	$(3+\psi)\times O(n\log n+e)$
Critical Point Analysis	$3\times\left[O(n\log n+e)+\lambda_s\times O(n_v+n_e)\right]$, $n_v+n_e<<n$
Topological Point Selection	$R\times O(n\log n+e)$
Initialization Stage (Ring Extraction)	$\mu\times O(n\log n+e)$
Shortest Path Algorithm (Ring Extraction)	$O(n\log n+e)$
Geometric Feature Extraction	
Spatial Location (Integral Geodesic)	$\gamma\times O(n\log n+e)$
Area Weighting, Area, Curvature, AvgDistance	$O(n)$
Geodesic Bands + Segments (DFS)	$(\mu+\gamma)\times\left[O(n)+O(n+e)\right]$
Overall Feature Extraction	$(7+R+\psi+\mu+\gamma)\times O(n\log n+e)$

Notations:

μ, γ:	Numbers of topological points and rings found, respectively
ψ:	Number of integral geodesics calculated by hierarchical partitioning. On average, $\psi\approx 20$
n, e:	Numbers of vertices and edges in the models, respectively
λ_s:	Number of saddles encountered in Critical Point Analysis
n_v, n_e:	Numbers of vertices and edges visited during DFS traversal in *Critical Point Analysis*, respectively
R:	A constant for Topological Point Selection. In our experiment, $R\approx 2$

significantly better. For example, in the case of baby and girl, the arms, legs, and bodies of girl are relatively longer than those of baby, and in the case of dog and wolf, the bodies and ears of wolf are fatter and sharper, respectively, than those of dog. All of these differences affect the weights and distributions of area, curvature, and average distance in TPR. We also notice that both TPR and MRG perform well on hand, frog, cat, and horse. This indicates that the range of maximum and minimum similarity values of one model group does not overlap with those of the other model groups. This is because their shapes are comparatively unique in our database. As a result, both TPR and MRG perform equally well on them.

7.3.2 Speed Comparison

We have also compared the time complexity of different processes between TPR and MRG as follows:

Feature Extraction: Table 4 shows all the steps and computation complexities of the feature extraction process. First, we apply LSD heuristics and hierarchical partitioning to find three source points using the Dijsktra algorithm. Second, we apply *Critical Point Analysis* on these three source points to identify feature points of the model. Third, we apply *Topological Point Selection*, which will stop as soon as the search radius is reached. Hence, only boundary vertices of each region may be visited more than once. The selection process thus visits most of the vertices, and its complexity is slightly higher than $O(n\log n+e)$, but bounded by $R\times O(n\log n+e)$. *Topological Ring Extraction*

TABLE 5
Time Analysis of TPR and MRG on Feature Extraction

	TPR	MRG
Average number of vertices / faces per model	9241 / 18478	9241 / 18478
Average number of integral geodesic calculated (per model)	50	131
Average time for all geodesic calculation (per model)	27.51s	72.64s
Number of feature captured	30	555
Average total time required for feature extraction per model	45.75s	75.23s

TABLE 6
Time Analysis of TPR and MRG on Feature Matching

	TPR	MRG
Average number of topological features (per model)	30 points and rings	555 r-nodes
Average total time for matching one model	1ms	16ms
Average total time for one query (similarity matrix calculations only)	0.6s	9.6s
Total time for our retrieval experiments using all 600 models as input queries (database parsing, retrieval score calculation and other overheads included)	529s	5840s

is a modified Dijsktra algorithm, and its complexity is the same as Dijsktra. For integral geodesic calculation, it is $(\mu+\gamma)\times O(n\log n+e)$. Since $(7+R)$ is a constant, the overall complexity of the algorithm depends on the number of geodesic calculations, which is $\psi+\mu+\gamma$.

Similar to TPR, MRG also requires the computation of integral geodesics for interval partitioning. It first samples a large number of seeds regularly over the model surface. It then computes the integral geodesics and interpolates the values over other vertices. In order to obtain a good approximation for interval partitioning, the number of seeds required is usually over 130 as shown in Table 5. TPR is comparatively much more efficient as it does not require a large number of seeds. Further, we limit the geodesic calculations to topologically important locations only as they dominate the overall feature extraction time. Hence, the number of geodesic calculations, which is $\psi+\mu+\gamma$, has an average value of 50. This significantly speeds up the whole process. Table 5 compares the performance of TPR and MRG. We can see that TPR is nearly two times faster than MRG to do the geodesic calculations. However, the overall feature extraction time of TPR is only about 40 percent faster. This is mainly due to the higher computational cost of computing the global geometric features.

Feature Matching: We apply the EMD to compare the features of two models. A theoretical computation analysis on the complexity of EMD is difficult as it is based on the simplex algorithm. However, according to [22], if EMD is formulated as a bipartite graph problem with signatures of the same size, the time complexity is roughly $O(n^3\log n)$, where n is the number of topological features. As a comparison, the overall complexity of MRG is $O(n_r\times(m_r+n_r))$, where m_r and n_r are the numbers of r-nodes of the two matching models. Hence, TPR has a higher complexity. However, as shown in Table 6, matching one model using TPR is 15 times faster than MRG because TPR has a much smaller number of topological features, n_t. On average, $n_t=30$ for TPR and n_t depends on the topology of the model. For MRG, the number of r-nodes, $n_r=555$.

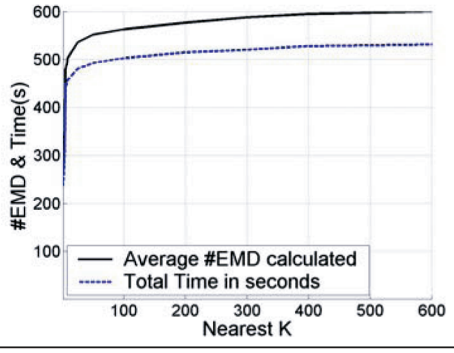
7.4 Performance of the Indexing Scheme

We have created a VP-tree with two fanouts. It stores at most two data points in each node. To build the indexing tree for our existing database, it takes 778.7 s. To carry out the retrieval test, we use all models in the database as input queries and vary k for nearest neighbor search. Table 7 shows the results for the retrieval experiment.

The diagram in Table 7 shows the total time of the k-Nearest Neighbor Search (using all 600 models as input queries) against k (the number of returned models) in the indexing scheme. From Table 7, we can see that if we perform a similar experiment as in Table 6 (that is, $k=600$), the total time required (531.2 s) is roughly the same as that in Table 6 (529.1 s). However, as most users typically like to retrieve only a few models that are most relevant to a given query, the indexing scheme would certainly speed up the retrieval process. In the extreme case, if a user just wants to find the best match, the retrieval system can handle 1-nearest neighbor search in 0.39 s on average, which is 44 percent of the original full matching time of TPR without indexing or 4 percent of the matching time of MRG. Table 7 also shows that the total computation time is proportional to the total number of EMD operations.

TABLE 7
Summary of K-Nearest Neighbor Search

Indexing			Average query time		
k	#EMD	Total time	TPR (k-NNS)	TPR	MRG
1	261	233.56s	0.39s	529s /600 =0.88s	5840s /600 =9.73s
2	282	250.75s	0.42s		
3	315	284.06s	0.47s		
4	357	322.95s	0.54s		
5	479	439.50s	0.73s		
6	483	444.00s	0.74s		
7	489	448.64s	0.75s		
8	492	450.13s	0.75s		
9	504	458.48s	0.76s		
10	505	459.16s	0.77s		
25	536	480.86s	0.80s		
600	600	531.22s	0.89s		



7.5 Summary

From the above experiments, we may conclude that TPR outperforms MRG in both accuracy and speed. However, TPR has some limitations. First, TPR currently cannot support nonmanifold, nonclosed models, or works poorly on models with genus number greater than zero. This is because *Critical Point Analysis* is modified from LSD and LSD performs poorly on such models. Second, because TPR considers only neighboring vertices within the same geodesic wavefront in *Critical Point Analysis*, the features may still be affected by noise if they appear in finely tessellated regions. Since our current focus is only on accuracy and speed, we have not explored multiresolution or part matching.

Due to the unavailability of source codes and the difficulties in implementing topology-based methods, we have only implemented MRG for comparison. However, our method also has advantages over the others. First, our method avoids slow subgraph isomorphism matching and is expected to be more efficient. Second, most existing methods do not consider discriminating *similar-skeleton* models. Their features are relatively local and may not be adequate to distinguish these models. Third, our similarity measure is a metric. By using spatial database techniques, our algorithm supports an indexing tree that encapsulates both topological and geometric features in one pass. It outperforms [23], as we have discussed in Section 2.

Currently, we are targeting a faster and more accurate method for feature extraction. One approach is to adapt from mesh segmentation methods. However, since most segmentation methods focus on segmenting meshes in a very accurate way, they generally have a higher computational cost and are not suitable for use in search engines. Another direction is to stabilize critical points by choosing an optimal scalar field [4]. One of the implementations that we can consider is to fix all maxima obtained from our Critical Point Analysis as a Dirichlet boundary condition [3] and calculate harmonic scalar field instead of geodesic.

8 Conclusion

In conclusion, we have introduced a novel and efficient method for retrieving 3D deformable models. Unlike classical topology-based approaches, we propose to use topological points and rings to describe a 3D model. By using additional global geometric features and weights to describe the importance of features, the matching can be modeled as a flow and transportation (EMD) problem. This opposes traditional methods that require skeleton or graph matching algorithms for matching topological entities. Our experimental results show that the new method outperforms MRG [9] in both accuracy and speed. In addition, since our similarity measure is a metric function, an indexing tree can be constructed that encapsulates both topological and geometric information in one pass.

Acknowledgments

The authors would like to thank the anonymous reviewers for their helpful and constructive comments on this paper.

References

[1] D. Chen, X. Tian, Y. Shen, and M. Ouhyoung, "On Visual Similarity Based 3D Model Retrieval," *Proc. Eurographics,* Sept. 2003.

[2] T. Dey, J. Giesen, and S. Goswami, "Shape Segmentation and Matching with Flow Discretization," *Proc. Workshop Algorithms Data Structures,* pp. 25-36, 2003.

[3] S. Dong, S. Kircher, and M. Garland, "Harmonic Functions for Quadrilateral Remeshing of Arbitrary Manifolds," *Computer Aided Geometry Design,* vol. 22, no. 5, pp. 392-423, 2005.

[4] S. Dong, P.-T. Bremer, M. Garland, V. Pascucci, and J. Hart, "Spectral Surface Quadrangulation," *ACM Trans. Graphics,* vol. 25, no. 3, pp. 1057-1066, 2006.

[5] M. Elad, A. Tal, and S. Ar, "Content Based Retrieval of VRML Objects—An Iterative and Interactive Approach," *Proc. Eurographics (EG) Multimedia,* pp. 97-108, 2001.

[6] A. Fu, P. Chan, Y. Cheung, and Y. Moon, "Dynamic VP-Tree Indexing for N-Nearest Neighbor Search Given Pair-Wise Distances," *Very Large Data Bases J.,* 2000.

[7] T. Funkhouser, P. Min, M. Kazhdan, J. Chen, A. Halderman, D. Dobkin, and D. Jacobs, "A Search Engine for 3D Models," *ACM Trans. Graphics,* vol. 22, no. 1, pp. 83-105, Jan. 2003.

[8] R. Gal, A. Shamir, and D. Cohen-Or, "Pose-Oblivious Shape Signature," *IEEE Trans. Visualization and Computer Graphics,* vol. 13, no. 2, pp. 261-271, 2007.

[9] M. Hilaga, Y. Shinagawa, T. Kohmura, and T. Kunii, "Topology Matching for Fully Automatic Similarity Estimation of 3D Shapes," *Proc. ACM SIGGRAPH '01,* pp. 203-212, 2001.

[10] M. Kazhdan, B. Chazelle, D. Dobkin, T. Funkhouser, and S. Rusinkiewicz, "A Reflective Symmetry Descriptor for 3D Models," *Algorithmica,* 2003.

[11] M. Kazhdan, T. Funkhouser, and S. Rusinkiewicz, "Rotation Invariant Spherical Harmonic Representation of 3D Shape Descriptors," *Proc. Symp. Geometry Processing,* 2003.

[12] M. Kazhdan, T. Funkhouser, and S. Rusinkiewicz, "Shape Matching and Anisotropy," *Proc. ACM SIGGRAPH '04,* Aug. 2004.

[13] L. Kobbelt, S. Campagna, J. Vorsatz, and H. Seidel, "Interactive Multi-Resolution Modeling on Arbitrary Meshes," *Proc. ACM SIGGRAPH '98,* pp. 105-114, Aug. 1998.

[14] F. Lazarus and A. Verroust, "Level Set Diagrams of Polyhedral Objects," *Proc. ACM Symp. Solid Modeling and Applications,* 1999.

[15] M. Mortara and G. Patane, "Affine-Invariant Skeleton of 3D Shapes," *Proc. Int'l Conf. Shape Modeling and Applications,* 2002.

[16] X. Ni, M. Garland, and J. Hart, "Fair Morse Functions for Extracting the Topological Structure of a Surface Mesh," *Proc. ACM SIGGRAPH '04,* Aug. 2004.

[17] M. Novotni and R. Klein, "3D Zernike Descriptors for Content Based Shape Retrieval," *Proc. ACM Symp. Solid Modeling and Applications,* June 2003.

[18] R. Ohbuchi and T. Takei, "Shape-Similarity Comparison of 3D Shapes Using Alpha Shapes," *Proc. Pacific Graphics Conf. '03,* Oct. 2003.

[19] R. Ohbuchi, M. Nakazawa, and T. Takei, "Retrieving 3D Shapes Based on Their Appearance," *Proc. ACM SIGMM Workshop Multimedia Information Retrieval,* Nov. 2003.

[20] R. Osada, T. Funkhouser, B. Chazelle, and D. Dobkin, "Matching 3D Models with Shape Distributions," *Proc. Int'l Conf. Shape Modeling and Applications,* pp. 154-166, May 2001.

[21] E. Paquet and M. Rioux, "Content-Based Access of VRML Libraries," *Proc. Multimedia Information Analysis and Retrieval Conf.,* pp. 20-32, 1998.

[22] Y. Rubner, C. Tomasi, and L. Guibas, "The Earth Mover's Distance as a Metric for Image Retrieval," *Int'l J. Computer Vision,* vol. 40, no. 2, pp. 99-121, Nov. 2000.

[23] H. Sundar, D. Silver, N. Gagvani, and S. Dickinson, "Skeleton Based Shape Matching and Retrieval," *Proc. Int'l Conf. Shape Modeling and Applications,* May 2003.

[24] R. Sumner, M. Zwicker, C. Gotsman, and J. Popović, "Mesh-Based Inverse Kinematics," *Proc. ACM SIGGRAPH '05,* 2005.

[25] P. Shilane, P. Min, M. Kazhdan, and T. Funkhouser, "The Princeton Shape Benchmark," *Proc. Int'l Conf. Shape Modeling and Applications,* June 2004.

[26] A. Tal and E. Zuckerberger, "Mesh Retrieval by Components," Technical Report CCIT-475, EE-2004, Faculty of Electrical Eng., Technion, Mar. 2004.

[27] G. Tam, R. Lau, and C. Ngo, "Deformable Geometry Model Matching by Topological and Geometric Signatures," *Proc. 17th Int'l Conf. Pattern Recognition (ICPR '04)*, pp. 910-913, Aug. 2004.
[28] J. Tangelder and R. Veltkamp, "Polyhedral Model Retrieval Using Weighted Point Sets," *Proc. Int'l Conf. Shape Modeling and Applications*, pp. 119-129, 2003.
[29] D. Vranic and D. Saupe, "3D Shape Descriptor Based on 3D Fourier Transform," *Proc. EURASIP Conf. Digital Signal Processing for Multimedia Comm. and Services (ECMCS '01)*, pp. 271-274, Sept. 2001.
[30] M. Yu, I. Atmosukarto, W. Leow, Z. Huang, and R. Xu, "3D Model Retrieval with Morphing-Based Geometric and Topological Feature Maps," *Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, 2003.

Gary K.L. Tam received the BEng degree in computer science from the Hong Kong University of Science and Technology in 2002 with first-class honors. In the same year, he was selected as one of the 23 Youth Telecom Ambassadors in the "ITU Telecom Asia 2002 Youth Forum" in Hong Kong. In 2004, he received the MPhil degree from the City University of Hong Kong. After graduation, he worked as an instructor in the Department of Computer Science of the same university for about two years. He is currently a PhD student of the University of Durham, England. His research mainly focuses on computer graphics and geometry processing. He is a student member of the IEEE.

Rynson W.H. Lau received the BSc degree in computer systems engineering with first class honors from the University of Kent in 1988 and the PhD degree in computer graphics from the University of Cambridge in 1992. He is currently a professor at the University of Durham. Prior to his current appointment, he taught at the City University of Hong Kong from 1998 to 2006 and the Hong Kong Polytechnic University from 1993 to 1998. He serves on the editorial board of *Computer Animation and Virtual Worlds*. He has served as the guest editor of a number of journal special issues, including *ACM Transactions on Internet Technology*, *IEEE Transactions on Multimedia*, *IEEE Transactions on Visualization and Computer Graphics*, *Presence*, and *IEEE Computer Graphics and Applications*. He has also served on the conference committee of a number of conferences, including program cochair of ACM VRST '04, ICAT '06, and conference cochair of ACM VRST '05 and CASA '05. He is a senior member of the IEEE.