Non-Rigid 3D Model Retrieval Based on Topological Structure and Shape Diameter Function

Yiyu Hong
Dept. of Copyright Protection
Sangmyung University
Seoul, Korea
e-mail: hongyiyu@cclabs.kr

Jongweon Kim
Dept. of Contents and Copyright
Sangmyung University
Seoul, Korea
e-mail: jwkim@smu.ac.kr

***Abstract*—With the increasing popularity of 3D technology like 3D printing, 3D modeling, etc., there is a growing need for searching similar models on the Internet. Subsequently, matching non-rigid shapes has become an active research field in computer graphics. In this paper, we present an efficient and effective non-rigid model retrieval method based on topological structure and Shape Diameter function (SDF). The integral geodesic distances are first calculated for each vertex on a mesh to construct the topological structure. Next, each node on the topological structure is assigned a local volume, which is calculated using the Shape Diameter function. Finally, we utilize the Hungarian algorithm to measure similarity between two non-rigid models. Experimental results on the latest benchmark (*SHREC' 15 Non-rigid 3D Shape Retrieval*) demonstrate that our method works well compared to the state-of-the-art.**

Keywords-non-rigid model retrieval; integral geodesic distance; shape diameter function; the Hungarian algorithm;

I. INTRODUCTION

The rapid development of 3D technology (3D printing, 3D scanning, 3D modeling, etc.) and computer networks have naturally led to more and more 3D models being widely used in many fields. Considering that designing and creating a 3D model is not that simple, retrieving 3D accurately and quickly from a huge database is becoming more and more necessary.

In the beginning of 3D shape retrieval, most efforts were focused on retrieval methods for rigid 3D models. However, in recent years, retrieval methods for non-rigid 3D models, which may require more shape analysis, have been an active research area in computer graphics. As shown in Fig. 1, non-rigid 3D models indicate that, with different poses or articulations, the human and hand models in each row are in the same category.

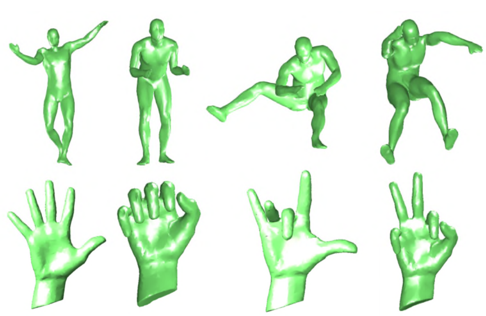

Figure 1. Non-rigid models

For the purpose of comparing two non-rigid models appropriately, shape descriptors are required to be invariant to non-rigid bending and articulations. In this paper, we utilize two characteristics on non-rigid models to measure dissimilarity between two non-rigid models. The first characteristic is geodesic distance and path, which means shortest distance and path between two vertices on the mesh surface. As we can see in Fig. 2 (a), the distance and path on the mesh between two pose-deformed models are nearly unchanged. The second characteristic is local volume on the corresponding position between two non-rigid models. In Fig. 2 (b), the color on the models indicates local volume which we calculated by SDF [1]. We can see the local volume on the corresponding positions are very similar.

II. RELATED WORK

During the past few years, many algorithms [2-7] have been proposed for 3D shapes retrieval. Generally, existing methods can be divided into mainly two types: retrieval methods for rigid and non-rigid 3D models. For rigid model retrieval, there were algorithms based on 2D views, spectral transformation, topology and statistic, etc. For more details about these algorithms, we refer readers to [9]. The second type is retrieve approaches for non-rigid models that can be seen as an extension of algorithms for rigid model retrieval.

Figure 2. (a) Geodesic path (b) Local volume

The extension requires extracted features from models to be isometry-invariant. For example, Lian et al. in [4] extended a 2D-view based rigid model retrieval method [3], which they had proposed before, to work on non-rigid model retrieval by first utilizing Multidimensional Scaling on the 3D model to get its bending invariant representation. Readers can refer to [10] for a good comparison of methods for non-rigid 3D shape retrieval.

One intuitive approach for non-rigid 3D model retrieval is to compare the topological structure and the corresponding geometric features between two non-rigid 3D models. Hilaga et al. [2] presented Multi-resolution Reeb Graph which is a topology construction method based on geodesic distance and reed graph theory. The topology matching used coarse-to-fine strategy to search the node pairs that give the maximum similarity. However, these kind of topology construction and similarity measure algorithms need to satisfy many conditions which cannot achieve good performance. Sfikas et al. [5] proposed a conformal factor guided topological structure construction algorithm. Nevertheless, conformal factor is mainly based on curvature, which can be easily affected by geometric noise.

Gal [6] proposed a 2D histogram based pose-oblivious shape signature which combines two scalar functions defined on the surface of a 3D model. The first function called as local-diameter function can measure local volume of a 3D model. In the following study [1], they did a little modification on this function and renamed it as SDF used in consistent mesh partitioning and skeletonisation. The second function is called centricity function, which measures the integral geodesic distances for the whole 3D model.

Inspired by the papers mentioned above, we propose here an efficient and effective approach for non-rigid 3D model retrieval, which is largely based on two pose invariant features: geodesic distance and SDF.

III. METHOD DESCRIPTION

A. Construction of Topological Structure

Our algorithm for construction of topological structure needs four steps. First, integral geodesic distances are calculated for every vertex on the mesh. Second, we extract vertices that reside on tips of protrusions and vertex on the center of surface using integral geodesic distances. Third, the protrusion tips and vertex on the surface center are connected by finding shortest geodesic paths. Finally, we sample points on the geodesic paths to extract topological nodes. Fig. 3 illustrates the overall topology construction process. We will discuss the process in detail below.

Integral geodesic distances were first proposed by Hilaga et al. [2] and their discrete case can be defined as following:

$$IGD(p) = \sum_{q \in S} g(p,q) \qquad (1)$$

where $g(p,q)$ denotes the shortest geodesic distance between vertex p and q. So $IGD(p)$ means integral of all geodesic distances from p to all vertices q on a surface S. In our approach, all geodesic distances and paths are computed by fast marching method [8]. Fig. 3 (b) shows color-coding of integral geodesic distances of the model. Generally, the vertex which has minimum integral geodesic distance would reside on the center of surface, and the vertices that farther from the center of surface would have larger scalar value of integral geodesic distance. Using this property of integral geodesic, we could extract vertices on the tips of protrusions by measuring whether the scalar value of integral geodesic distance of a vertex is the local maxima within a radius of geodesic neighborhood [7]. In our implementation, the radius of geodesic neighborhood is set as $\sqrt{0.08 * area(S)}$. In Fig. 3 (c), the blue point and the red points represent the surface center and extracted protrusion tips respectively.

To construct the topological structure simply and effectively, we found that connecting protrusion tips and surface center on the mesh surface can approximately represents the topology of a model without any complex process (Fig. 3 (c)). The connection can be easily done by finding the shortest paths from each protrusion tip to the surface center using fast marching method [8]. In Fig. 3 (c), the black line represents the shortest paths. For better presentation, we show the topological structure alone in Fig 3 (d). Every path from protrusion tip to surface center, we call *topological path* in this paper.

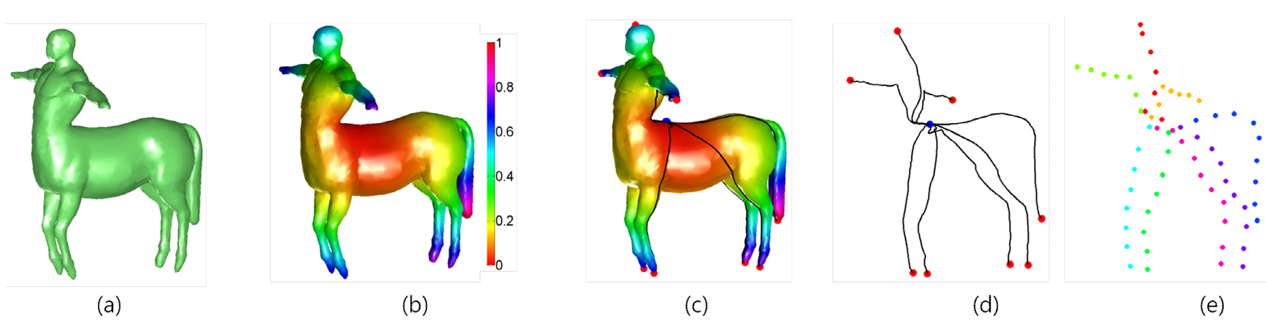

Figure 3. Overall topology construction process (a) Original model (b) Color-coding of integral geodesic distance of the model (c) Shortest paths from each protrusion tip to surface center (d) Topological structure (e) Selected topological nodes.

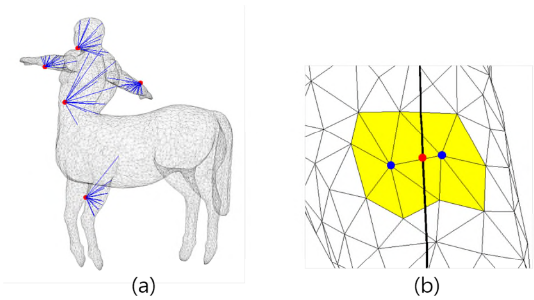

Figure 4. (a) Cone-shaped rays sent to the inside of the mesh (b) Faces related to SDF values assignment.

After constructing the topological structure, we select the topological nodes which can represents corresponding sub-part of the 3D model. On every topological path, we choose points from protrusion tip to surface center in a certain geodesic distance interval T_n, and regard it as topological nodes. In Fig. 3 (e), the points with the same color represent selected topological nodes on the same topological path. We define the topological nodes on the same topological path with order from the protrusion tip to surface center as a *topological string*.

B. SDF values Assignment

Shapira et al. in [1] have introduced the SDF, which is a scalar function defined on the mesh surface to measure the local shape's volume of a 3D mesh. For a given face on a mesh, the SDF send cone-shaped rays (Fig. 4 (a)) from the centroid of a face to its normal-opposite side (inward direction to the mesh). The length of the rays can be calculated by checking the ray-mesh intersections. Finally, the scalar value of the SDF for the face is the weighted average of all ray's lengths.

In our implementation, SDF is computed using a cone of angle 120° with 30 rays. We do not calculate SDF values for every face on the mesh, we only need to care about faces which are nearby the topological nodes. As shown in Fig. 4 (b), assume that the red point is a topological node which we selected on a topological path (thick black line), and then we find the one vertex-ring faces (yellow faces) of the two blue points which construct the edge where the topological node resides on. Subsequently, the topological node is assigned the average SDF values of these faces.

After calculating all SDF values for topological nodes, in order to be compatible with 3D meshes in different scales and resolution, the SDF values are normalized as follows:

$$nsdf(TN) = \frac{sdf(TN)}{\sqrt{area(S)}} \tag{2}$$

where $sdf(TN)$ and $nsdf(TN)$ denote the original SDF value and normalized SDF value for topological node respectively, and S denotes surface of the mesh. Instead of using the logarithmized version [1], we normalize the SDF values by dividing them by the root area of the mesh. Because we do not calculate SDF values for every face on the mesh, as mentioned before, to reduce computation time, it may be that the max SDF value and the min SDF value could be different between two non-rigid models in the same class.

C. Matching Approach

For matching approach, we first calculate all dissimilarity distance among topological strings with node-by-node SDF values between two 3D models. Next, the Hungarian algorithm is utilized to find a "minimum matching". The Hungarian algorithm is a combinatorial optimization algorithm that solves the assignment problem. Our matching approach is similar to [5], but different in penalizing method.

For calculating the dissimilarity between two topological strings, if two topological strings have the same number of topological nodes, the dissimilarity can be simply calculated by averaging the difference between the corresponding SDF scalar values. If two topological strings have different lengths, we first append the shorter topological string with its last topological node to have same length as the longer one. Then we penalizing these appended values by putting weights. Let p and q be two topological strings, and let $p[i].sdf$ denotes the SDF value of the ith topological node start from protrusion tip on p. Assuming that p has more topological nodes than q, the dissimilarity between two topological strings is defined as:

$$Dis(p,q) = \frac{\left(\begin{array}{c} \sum_{k=1}^{len(q)} |p[k].sdf - q[k].sdf| + \\ \sum_{l=len(q)+1}^{len(p)} |p[l].sdf - q[len(q)].sdf| \\ \times w_{l-len(q)} \end{array} \right)}{len(q)} \tag{3}$$

$$w_t = 1 + t \times \alpha, \quad t = 1, \dots, len(p) - len(q) \tag{4}$$

where len denotes the number of topological nodes in a topological string. w denotes the penalizing weights. Fig. 5 illustrates the comparison between two topological strings. In our experiments, $\alpha = 0.2$ yields good retrieval result.

Let M and N be two 3D models. Assuming they have m and n topological strings respectively, after comparing each topological string in M with each in N, we can get a $m \times n$ matrix filled with the dissimilarity values, which are calculated using equation (3). To apply the Hungarian algorithm, the dissimilarity matrix is required to be a square

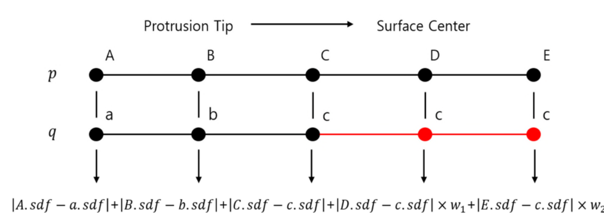

Figure 5. Comparison between two topological strings (p, q) with SDF values, where $len(p) > len(q)$

matrix. In the case of $m = n$, the Hungarian algorithm can be directly applied. And if $m \neq n$ we pad the rows (or columns) of the dissimilarity matrix with mean of existing values of the columns (or rows). Assuming that $m > n$, we can define a $m \times m$ dissimilarity matrix as $Dism(i, j)$, where $1 \leq i \leq m,\ j \leq n$ have the dissimilarity values of topological strings. The padding procedure is mathematically formulated as follows:

$$Dism(i,j) = \sum_{u=1}^{n} Dism(i,u)/n, \qquad (5)$$
$$1 \leq i \leq m, n < j < m, m > n$$

After applying the Hungarian algorithm, it will return the "minimum matching" indexes. The final dissimilarity value between the two models is the average of the indexed value of the dissimilarity matrix.

IV. EXPERIMENTAL RESULTS

In this section, we evaluate the retrieval performance of the proposed algorithm and compare it with other state-of-the-art methods. We carry out experiments on the datasets of the *SHREC' 15 Non-rigid 3D Shape Retrieval* [11]. The datasets contain 1200 deformable models, classified into 50 classes, each with 24 models. The retrieval accuracy is evaluated by the following five quantitative measures [9]:

- Nearest Neighbor (NN): The percentage of best matches that belong to the query's class.
- First Tier (FT) and Second Tier (ST): The percentage of models belonging to the query's class that appear within the top $(K-1)$ and $2(K-1)$ matches respectively, where K is the number of models in the query's class.
- E-measure: A composite measure of the precision and recall for a fixed number (32) of retrieved models.
- Discounted Cumulative Gain (DCG): A statistic that weights correct results near the front of the list more than correct results later in the ranked list.

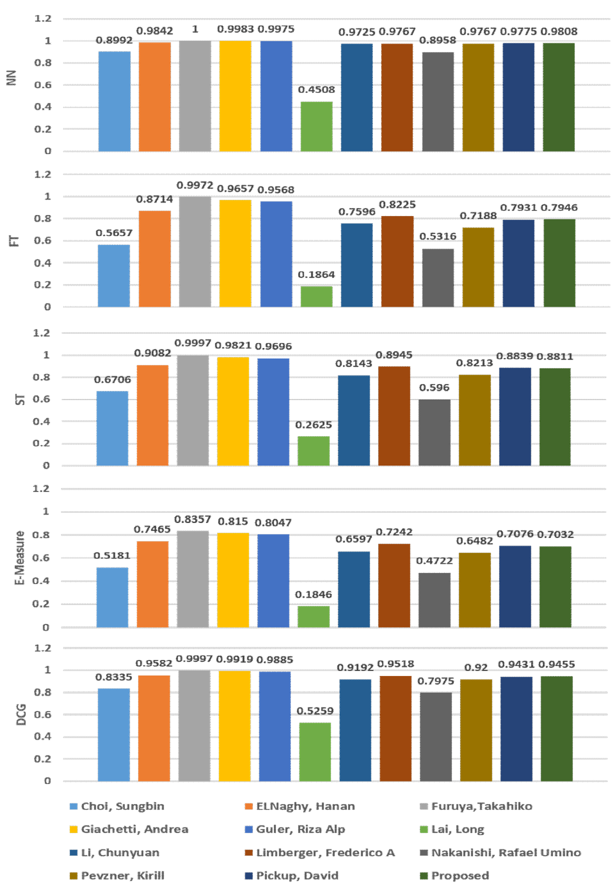

Figure 6. Comparative results based on the five standard measures for the *SHREC' 15 Non-rigid 3D shape retrieval* dataset.

All metrics above are in the range [0,1] and higher values indicate better retrieval results. For more details about the metrics, we refer readers to [9].

We implemented the proposed algorithm in Matlab on a personal computer with a 3.60 GHz i7-4790 CPU, 8GB DDR3 memory. As the calculation of geodesic distances is computationally expensive, we first use QSlim [12] to simplify mesh with 1500 faces and it takes only around 3 seconds for topological structure construction and corresponding SDF values calculation of a mesh by adopting parallel computation with 4 cores. For mesh matching which uses the Hungarian algorithm, it takes around 2 milliseconds for comparing between two meshes. The proposed algorithm was evaluated on the datasets with parameters: $T_n = \sqrt{0.0025 * area(S)},\ \alpha = 0.2$

As we can see from Fig. 6, our method obtains competitive results among the 11 contestants. There are only two contestants writing about their running time, Giachetti's HAPT algorithm needs 3 min on average for extracting feature map of the tested dataset, and Limberger's algorithm needs 18 seconds to compute three local descriptors on a model. Moreover, our topological structure and SDF value based descriptor is compact, which only need less than 2000 bytes.

V. CONCLUSION

In this paper, we developed an efficient and effective method for the retrieval of non-rigid 3D models mainly based on geodesic distance and Shape Diameter function, which are two pose-invariant features on the mesh surface. The experiment on the *SHREC' 15 Non-rigid 3D Shape Retrieval* shows that our method is competitive against state-of-the-art. Furthermore, our method has the advantage of low complexity to implement, fast running time and small data storage space for descriptors.

ACKNOWLEDGMENT

This research was supported by Institute for Information & communications Technology Promotion (IITP) funded by the Ministry of Science, ICT and Future Planning (No. R0126-15-1024).

REFERENCES

[1] L. Shapira, A. Shamir, and D. Cohen-Or, "Consistent mesh partitioning and skeletonisation using the shape diameter function," The Visual Computer, vol. 24, pp. 249-259, 2008.

[2] 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.

[3] Z. Lian, A. Godil, and X. Sun, “Visual similarity based 3D shape retrieval using bag-of-features,” Proc. SMI’10, pages 25–36, 2010.

[4] Z. Lian, A. Godil, X. Sun, and H. Zhang, “Non-rigid 3D shape retrieval using multidimensional scaling and bag-of-features,” Proc. ICIP 2010, pages 3181–3184, 2010.

[5] K. Sfikas, T. Theoharis, and I. Pratikakis, “Non-rigid 3D object retrieval using topological information guided by conformal factors,” The visual Computer, vol. 28, pp. 943-955, 2012.

[6] 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.

[7] A. Agathos et al., “3D articulated object retrieval using a graph-based representation,” The Visual Computer, vol 26, pp. 1301–1319, 2010.

[8] J. Sethian and R. Kimmel, “Computing geodesic paths on manifolds”, Proc. of Natl. Acad. Sci. vol. 95, no. 15, pp. 8431–8435, 1998.

[9] P. Shilane, P. Min, M. Kazhdan and T. Funkhouser, “The Princeton Shape Benchmark,” Proc. SMI’04, pages 167-178, 2004.

[10] Z. Lian et al., “A comparison of methods for non-rigid 3D shape retrieval”, Patten Recognition, vol. 46, no. 1, pp. 449-461, 2013.

[11] Z. Lian et al., “SHREC’15 Track: Non-rigid 3D Shape Retrieval,” Proc. Eurographics Workshop on 3D Object Retrieval, pp. 107-120, 2015.

[12] M. Garland, Qslim Simplification Software, Available from: http://www.cs.cmu.edu/~./garland/quadrics/qslim.html [retrieved: July, 2016].