A Method for Content-Based 3D Model Retrieval by 2D Projection Views

Hang XIAO [1] , Xiubin ZHANG [1]
1: School of Electronic, Information and Electrical Engineering
Shanghai Jiaotong University
No. 1954, Huashan Road, Shanghai
P.R. China
xiaohang@sjtu.edu.cn, http://www.sjtu.edu.cn

Abstract: - An effective approach for 3D model retrieval is to represent models by a set of projection views . In this paper, we propose a flexible framework for view-based 3D model retrieval. The framework provides users the ability to configure three components: the number of views, the type of views, and the 2D shape descriptors used to distinguish views. Because the three components are closely related to the performance of a view-based 3D model retrieval system and usually application-dependent, it is necessary to make them configurable. To achieve this goal, the framework employs a new view polyhedron positioning algorithm and provides an adaptable interface to its 3D shape feature extraction module. We build a demo 3D model retrieval system based on the framework. Experiment shows that the performance of the demo system is satisfactory.

Key-Words: - 3D model retrieval, information retrieval, 3D shape analysis, model feature extraction, view-based models

1 Introduction

The development of computer graphics has made great progress over the last few decades. One aspect of the progress is that 3D models can be generated much easier than before with new 3D modelling tools and digitalizing technologies. As 3D models are proliferating, it is necessary to have a system for 3D shape retrieval.

Many 3D shape retrieval approaches have been proposed recent years. Among them are view-based approaches, which distinguish 3D models by comparing multiple 2D projection images of the models. The following two features of view-based approaches make them important to 3D model retrieval. 1) Although it seems that view-based approaches do not use the 3D geometry information fully, they achieve better retrieval performance than other approaches [1]. 2) Because view-based retrieval algorithms work in image space instead of object space, they are not sensitive to noise and model degeneracy.

The main shortcoming of view-based descriptors is their high computational complexity. Another problem of view-based approaches lies in what type of views (silhouette or depth map?) should be taken and which 2D shape descriptor (Fourier descriptor or Zernike moments?) should be adopted. Depending on different application requirements, there may be different choices [2].

In this paper, we propose a flexible framework for view-based 3D model retrieval, which provides multiple-level retrieval precision and computational complexity. It is also extensible for different types of views and 2D shape descriptors.

The rest of the paper is organized into the following sections: Some related work is summarized in Sec. 2. Then, we describe our framework system and provide some implementation details in Sect. 3. Sect. 4 presents a demo 3D model retrieval system based on the framework and the experimental results of it. Finally, we draw the conclusions in Sect. 5.

2 Related Works

The main idea of view-based 3D shape descriptor comes from the fact that two 3D objects are similar, if they look similar from all viewing angles. Thus, almost all view-based retrieval approaches use a “view polyhedron” to encompass the 3D model, and viewpoints are placed on its vertices. We will discuss three typical view-based 3D model retrieval systems in the following.

Funkhouser et al [3] have a view-based 3D shape descriptor in their 3D model search engine to provide a 2D sketch query interface. The descriptor uses 13 views defined by a bounding cube. Viewpoints are taken at the center of three faces, the

four top corners, and the middle of six edges of that cube. The approach uses too few numbers of views, so its retrieval precision is relatively low.

Chen et al [4] propose the Light Field Descriptor (LFD) for 3D model retrieval. A LFD is a set of images rendered from 10 vertices of a dodecahedron. To be robust against rotation, a model can be represented by a number of LFDs which are evenly distributed around it. Their system gets an outstanding retrieval performance, but it is very time and computation consuming at the same time, because a large number of views need to be computed and compared. Moreover, due to only the simple silhou ettes are extracted as views, Chen's 3D shape descriptor is not very capable of discriminating some concave geometries. For instance, it can not tell a sphere from a sphere with a pit on the surface. If depth maps are used as views, such as the Ohbuchi's system [5], this problem can be solved.

Ohbuchi et al [5] present a 3D model retrieval system using a view polyhedron with 42 vertices. The depth maps of 3D models are taken as views and a region-based 2D shape descriptor is employed to distinguish them. But this system adopts a non-regular view polyhedron, so the rotation alignment of 3D models can not be done efficiently and effectively.

As above discussed, each of these systems has its advantage and disadvantage. In different applications, there are different strategies to deal with these issues. With the intention of allowing the needs of diverse users and applications, we follow Chen's Light Field Descriptor idea, but take the configurability of system into consideration to design a flexible view-based 3D model retrieval framework.

3 View-Based 3D Model Retrieval Framework

The overall structure of our framework is illustrated in Fig. 1.

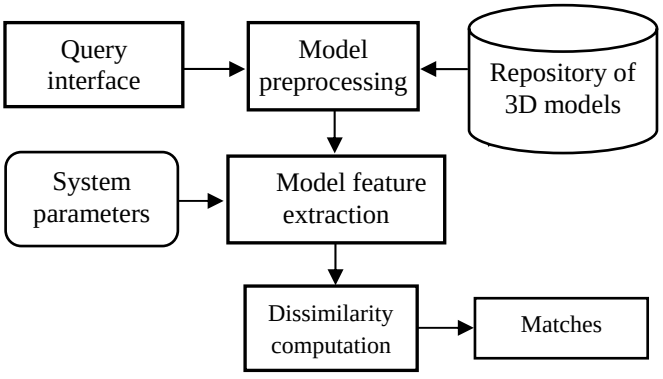


Fig. 1 Structure diagram of the view-based 3D model retrieval framework

Like most 3D model retrieval systems, its main parts include Query interface, Model preprocessing, Model feature extraction and Dissimilarity computation [6]. What makes it different is that the " Model feature extraction" module is configurable, which means it can be changed through specifying system parameters by users. We will elaborate on these parts in the following subsections.

3.1 Query Interface and Model Preprocessing

The query interface receives a 3D model submitted by users. The queried model follows the same processing flow as models in the repository until their dissimilarity is computed. Before extracting shape feature from 3D models, a preprocessing step for translation and scale normalization is need. This step ensures that 3D models are fitly circumscribed by the view polyhedron. Rotation normalization is not included in this step. Two models are implicitly aligned in rotation when their dissimilarity are compared, i.e. rotation alignment takes place in the "Dissimilarity computation" module.

3.2 Model Feature Extraction

The shape feature of 3D models is extracted at the "Model feature extraction" module in Fig. 1, which is the heart of our framework.

We employ the dodecahedron as the view polyhedron. To be robust against model rotation and get higher retrieval precision, more views of a 3D model need to be taken. When 20 views[1] that a single dodecahedron provides is not enough, a set of view polyhedrons with different rotating angle is applied to a model. The framework is designed to be able to use different number of view polyhedrons. The variability of the number of view polyhedrons enables multiple-level retrieval precision and computation complexity. Users can set a proper number according to their specific application. For example, many models are aligned with Cartesian axes when they were built (This may be explained by that people don't like posing an object random).

1 If the view is reflective symmetric (for example, silhouette), there are only 10 different views produced for a 3D model, because the views projected from two opposite vertices on the dodecahedron are identical.

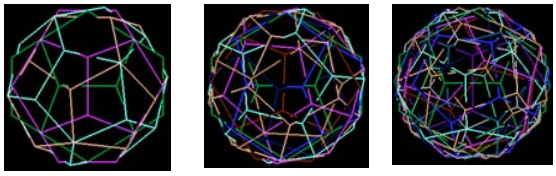
In these cases, we can use a smaller number of view polyhedrons, for models generally fall into several fixed poses.

When multiple view polyhedrons are applied, they need to be evenly distributed around the model to cover every viewing angle. Chen's system [4] makes use of Turk's random points relaxation algorithm [7] to generate evenly distributed LFDs(view polyhedrons). Our model retrieval framework employs a simple but effective algorithm to position view polyhedrons. Suppose that we apply N view polyhedrons to a model. Each view polyhedron is rotated around x-, y- and z-axis by an offset angle. The offset angle for the i-th view polyhedron is:

$$Offset\_angle_i = \frac{108^\circ}{N} \times (i-1) \quad (1)$$

$$i = 1,2,\ldots,N$$

Due to the circular symmetry of the dodecahedron, the viewing angle range is 0 to 180-degree. Formula. 1 divides the viewing angle range into equal parts, each of them being taken by a view polyhedron. Fig. 2 shows the distribution of view polyhedrons generated by our algorithm.



(a) (b) (c)

Fig. 2. The distribution of view polyhedrons with different N. (a) N = 4 (b) N = 8 (c) N = 12

When view polyhedrons are ready, we can take views from a 3D model. Silhouette may be the simplest and the most common type of views. Other possible kinds of views include depth map, normal map or thickness map, etc. Theoretically, these maps are more helpful for 3D shape discriminating, since they contain more geometry information than silhouette. As above mentioned, depth map has been already explored by some researchers. We expect that other kinds of maps will also have its utility for view-based 3D model retrieval.

Effectively measuring the shape dissimilarity between views is crucial for 3D shape discrimination. Fortunately, in the past decades, CBIR (Content-Based Image Retrieval) community has already brought out a lot of 2D shape descriptors for image retrieval [8]. Just like the 3D shape discrimination, 2D shape discrimination is also a non-trivial work. There is no 2D shape descriptor suited for all cases. Using a combination of different 2D shape descriptors can get better retrieval performance. Moreover, because the framework can use different kinds of maps as views, there is need for different 2D shape descriptors to distinguish them. Thus, our framework provides an interface to assemble different 2D shape descriptors. Users can try, test and combine different descriptors to find out the best ones suited for their application.

3.3 Dissimilarity Computation

To compute the dissimilarity between two 3D models is to compare the dissimilarity between corresponding views of them. We use the algorithm proposed by Chen [4] to do this job. When computing the dissimilarity, rotation alignment of two 3D models is done at the same time. Chen's algorithm computes the dissimilarity distance between two view polyhedrons of the two 3D models by compare their views in pairs subject to view consistency constrain. If multiple view polyhedrons are employed, each view polyhedron of a model is compared with each of another and the minimal dissimilarity distance between all view polyhedrons is taken as the dissimilarity distance between the two 3D models.

4 Experimental Results

We implement a 3D model retrieval system based on the proposed framework. The demo system uses 4 view polyhedrons, silhouette as views, and employs two 2D shape descriptors to distinguish them: the Geometric Moment Descriptor (GMD) and the Polar Radius Fourier Transform [9].

GMD is a kind of feature based on region. Let $f(x,y)$ be a 2D digital image with its size being $M \times N$. The $(p+q)$ rank moment is defined as following:

$$M_{pq} = \sum_{i=0}^{M} \sum_{j=0}^{N} f(i,j) i^p j^q, \; p,q = 0,1,\cdots \quad (2)$$

To be invariant to position, we can use the central moment, that is to say, calculating the moment with the condition that the centroid of the object in the image is the coordinate origin. Thus the central moment can be defined as:

$$m_{pq} = \sum_{i=0}^{M} \sum_{j=0}^{N} f(i,j)(i-x_c)^p (j-y_c)^q \quad (3)$$

$$p,q = 0,1,\cdots$$

(x_c, y_c) is the centroid. If we unitize the central moments according to area by substituting $m_{pq} \Big/ M_{00}^{(p+q)/2+1}$ for m_{pq} itself, the obtained ones have the character of invariance to scale. There are 7 moments as:

$$u_1 = m_{20} + m_{02}$$

$$u_2 = (m_{20} - m_{02})^2 + 4m_{11}^2$$

$$u_3 = (m_{30} - 3m_{12})^2 + (3m_{21} - m_{03})^2$$

$$u_4 = (m_{30} + m_{12})^2 + (m_{21} + m_{03})^2$$

$$u_5 = (m_{30} - 3m_{12})(m_{30} + m_{12})[(m_{30} + m_{12})^2 - 3(m_{21} + m_{03})^2] + (3m_{21} - m_{03})(m_{21} + m_{03})[3(m_{30} + m_{12})^2 - (m_{21} + m_{03})^2]$$

$$u_6 = (m_{20} - m_{02})[(m_{30} + m_{12})^2 - (m_{21} + m_{03})^2] + 4m_{11}(m_{30} + m_{12})(m_{21} + m_{03})$$

$$u_7 = (3m_{21} - m_{03})(m_{30} + m_{12})[(m_{30} + m_{12})^2 - 3(m_{21} + m_{03})^2] + (m_{30} - 3m_{12})(m_{21} + m_{03})[3(m_{30} + m_{12})^2 - (m_{21} + m_{03})^2]$$

The 7 moments are invariant to the image translation, rotation and scaling. We can constitute a feature vector of an image using these 7 moments.

Polar Radius Fourier Transform is based on the contour of an image.

$$a_n = \frac{1}{N} \sum_{i=0}^{N-1} r_i \exp(-\frac{j2\pi ni}{N}) \quad (4)$$

$$n = 0, 1, \cdots, N-1$$

We choose the first several coefficients as the high frequency part is not steady and changes drastically when the contour has even a slight difference. To get invariance for scaling, we choose the following list as feature vector:

$$[\frac{|a_1|}{|a_0|}, \frac{|a_2|}{|a_0|}, \cdots, \frac{|a_{21}|}{|a_0|}] \quad (5)$$

We use the 3D model database of Princeton Shape Benchmark [1] as our test database. The "precision and recall" curve is used to evaluate the performance of our demo system. Our demo system is compared with two other competing 3D shape descriptors: the LFD [4] and the D2 Shape Distribution [10]. As shown in Fig. 3, our demo system outperforms the D2 Shape Distribution significantly, but is inferior to the LFD. This can be explained by two reasons: 1) our demo system uses fewer views than the LFD system. We set 4 view polyhedrons in the demo system, while the LFD system uses 10 view polyhedrons. 2) Our demo system adopts a simple region-based 2D shape descriptor. We employ GMD for the demo system, while the LFD system uses Zernike moments as one of its 2D shape descriptor. Generally, Zernike moments perform better than GMD.

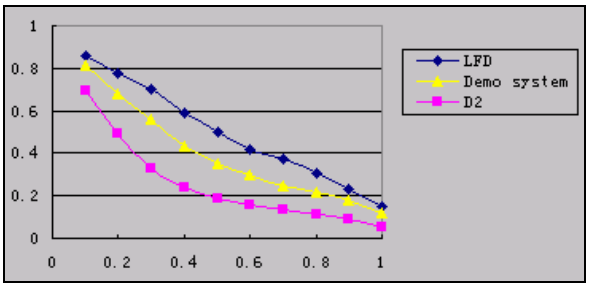


Fig. 3. Precision-recall plot of the LFD, D2 and demo system

Although losing some performance, our demo system is less computation consuming than the LFD system by employing fewer views and a simple 2D shape descriptor. The experiment was done on a PC with a Pentium IV 2.0GHz CPU, 256M memory. Table. 1 shows the average feature extraction time and dissimilarity computation time of the two systems. Our demo system is 42% faster on feature extraction and 44% faster on dissimilarity computation than the LFD system.

Table 1. Feature extraction time and dissimilarity computation time

	Feature extraction	Dissimilarity computation
LFD	7.75s	0.004131s
Demo system	3.23s	0.001824s

5 Conclusions

In this paper, a flexible framework for view-based 3D model retrieval is proposed. Using the framework, users can build a 3D model retrieval system easily according to their specific application. The framework is also very useful for researchers to observe the tradeoff between computational complexity and retrieval precision, and to test different types of views and 2D shape descriptors.

In the future wok, we will test different type of views and 2D shape descriptors using the framework. We will also explore the three configurable components on a variety of 3D model databases, expecting to find proper configurations for different classes of 3D models.

Acknowledgements

We would like to thank the Princeton Shape Retrieval and Analysis Group for providing the Princeton Shape Benchmark. This work is supported by a grant from the key programs of Shanghai Municipal Information Commission (No. 040xx920008).

References:

[1] Shilane P., Min P., Kazhdan M., Funkhouser T., The Princeton Shape Benchmark. *Proc of the International Conference on Shape Modeling. Genova, Italy*, 2004, pp. 167-178.
[2] Suzuki M. T., Shibata T., A pattern matching technique for detecting similar 3D terrain segments, *WSEAS Transactions on Information Science and Applications*, Vol. 2, No. 2, February, 2005, pp. 177-182.
[3] Funkhouser T., Min P., Kazhdan M., Chen J., Halderman A., Dobkin D., Jacobs D., A Search Engine for 3D Models. *ACM Transactions on Graphics*, Vol.22, No.1, 2003, pp. 83-105.
[4] Chen D. Y., Tian X. P., Shen Y. T, Ouhyoung M., On Visual Similarity-Based 3D Model Retrieval. *Computer Graphics Forum*, Vol.22, No.3, 2003, pp. 223-232.
[5] Ohbuchi R., Nakazawa M., and Takei T., Retrieving 3D Shapes Based on Their Appearance. *Proceedings of the 5th ACM SIGMM International Workshop on Multimedia Information Retrieval*, 2003, pp. 39-46.
[6] Huang C. H., Wang C. S., Shih T. K., 3D human kinematical motion retrieval system, *WSEAS Transactions on Information Science and Applications*, Vol. 4, No. 5, May, 2007, p 901-908.
[7] Turk G., Generating Textures on Arbitrary Surfaces Using Reaction-Diffusion. *Computer Graphics*, Vol.25, No.4, 1991, pp. 289-298.
[8] Chevalier S., Geoffrois E., Preteux F., 2D Markovian modeling for character recognition and segmentation, *WSEAS Transactions and Communications*, Vol. 4, No. 9, September, 2005, pp. 915-920.
[9] Zhang D. S., *Image Retrieval Based on Shape*. Phd dissertation, Monash University, 2002.
[10] Osada R., Funkhouser T., Chazelle B., Dobkin D., Matching 3D Models with Shape Distributions. *Proceedings of the International Conference on Shape Modeling and Applications*, 2001, pp. 154–166.