3D Model Searching

A Seminar Report
Submitted in partial fulfilment of
the requirements for the award of the degree of

Master of Technology
in
Computer Science and Engineering

by

Febin P Jacob
M105105

Department of Computer Science & Engineering
College of Engineering Trivandrum
Kerala - 695016
2010-11

College of Engineering Trivandrum
Department of Computer Science & Engineering

Certified that this Seminar Report entitled

3D Model Searching

is a bonafide record of the seminar presented by

Febin P Jacob
M105105

in partial fulfilment of
the requirements for the award of the degree of
Master of Technology
in
Computer Science & Engineering

Mr. Rameez Mohammed A.
Guide
Lecturer
Dept.of Computer Science & Engineering

Dr. M.S. Rajasree
Professor and
Head
Dept.of Computer Science & Engineering

Acknowledgement

I would like to express my sincere gratitude and indebtedness to my guide **Mr Rameez Mohammed**, Lecturer, Department of Computer Science And Engineering for his valuable guidance and encouragement during the course of preparation for this seminar.

I am thankful to **Dr. M. S. Rajasree**, Head of the Department of Computer Science and Engineering, **Shine S.**, P.G Co-ordinator, **Sreelekshmi R.**, Asst. Professor and **Anwar A.**, Asst. Professor and Project Co-ordinator for their valuable help and support.

I also acknowledge my gratitude to other members of faculty in the Department of Computer Science And Engineering and all my friends for their whole hearted cooperation and encouragement.

Above all, I am thankful to God Almighty.

Febin P Jacob

Contents

Abstract

Three dimensional shape searching refers to the process of shape based retrieval of 3D models from a large database by determining the similarities among 3D shapes. 3D models are extensively used in the movie industry, video game industry, architecture industry, medical industry, and in the science and engineering community. Three dimensional model databases used around the web is growing at a fast speed which makes the problem of 3D model searching significant. Time and effort that is involved in creating a high quality 3D model is high. Considerable amount of resources could be saved if these existing models are reused. However, finding a model is not easy since most online models are scattered across the web on repository sites, project sites, and personal homepages. The main difficulties that arise when developing a 3D model search engine are the selection of query interface and computational representations of the 3D model. The query interface should be simple enough that even novice users could use it. The representation using the shape descriptors should be efficient such that the computation and comparison could be done correctly. Since the search engine is an interactive system, a matching method has only limited time to run. One of the principal challenges faced in the area of shape matching is that in many applications, a model and its image under a similarity transformation are considered to be the same. Thus, the challenge in comparing two shapes is to find the best measure of similarity over the space of all transformation. So a rotation invariant shape representation of 3D shape descriptors is needed to perform the search effectively.

1 Introduction

Over the last decade, tools for acquiring and visualizing 3D models have become integral components of data processing in a number of disciplines, including medicine, chemistry, architecture and entertainment. With the proliferation of these tools, an explosion in the number of available 3D models has occured. These developments are changing the way people think about 3D data. For years, a primary challenge in computer graphics has been how to construct interesting 3D models. Now the key question shifted from "how do we construct them?"to "how do we find them?".

An important question then is how people will search for 3D models. Of course, the simplest approach is to search for keywords in filenames, captions, or context. However, this approach can fail: (1) when objects are not annotated (e.g. "B19745.wrl"), (2) when objects are annotated with unspecific or derivative keywords (e.g. "yellow.wrl"or "sarah.wrl"), (3) when all related keywords are so common that the query result contains a flood of irrelevant matches (e.g. searching for faces i.e., human not polygonal), (4) when relevant keywords are unknown to the user (e.g. objects with misspelled or foreign labels), or (5) when keywords of interest were not known at the time the object was annotated.

In these cases, shape-based queries will be helpful for finding 3D objects. Shape can also be used to discriminate between similar objects. Query interfaces based on 3D sketches and 2D sketches can be used in combination with text interfaces to yield a better search result.

2 System Overview

The execution of the system proceeds in four steps: crawling, indexing, querying and matching. Crawling and indexing takes a lot of time. So these steps are done off-line. Querying and matching is done for each user query at run time. The main research issue at the heart of this system is how to provide shape based query interfaces and matching methods that enable easy and efficient retrieval of 3D models from a large repository. The organization of the system is shown in Figure 1.

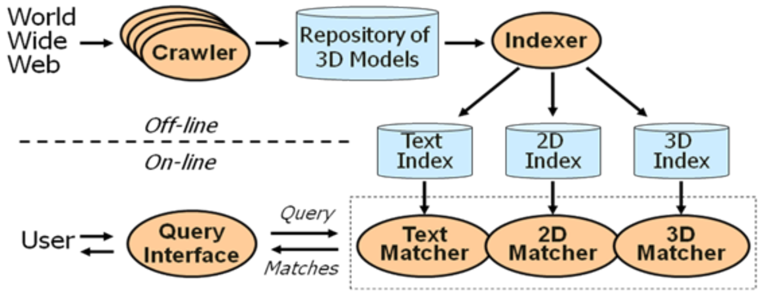

Figure 1: System Overview

The following text provides an overview of each step and highlights its main features.

1. **Crawling:** Crawling is the process by which the 3D models that are scattered across the web are collected to build the database. 3D data still represents a very small percentage of the web, and high quality models represent an equally small percentage of all 3D data. So a focused crawler that incorporates a measure of 3D model quality into its page rank should be used for crawling the web.

2. **Indexing:** Indexing is the process of computing the indices to retrieve 3D models efficiently based on text and shape based queries. This is done using a 3D shape descriptor based on spherical harmonics that is descriptive, efficient to compute, robust to model degeneracies, and invariant to rotations.

3. **Querying:** Querying allows the user to search interactively for 3D models. Query methods based on text keywords, 2D sketching, 3D sketching and model matching can be used.

4. **Matching:** Matching is the process by which the shape descriptor of the query model is found and compared with the shape descriptors of models in the database to fetch the matching models.

3 Challenges

Selection of the query interface is important because the traditional text based query may not return the desired result if the models in the web are poorly annotated. Thus shape based query interfaces are used which utilizes other attributes of a 3D model like shape and appearance. Shape based query interfaces like 2D sketching and 3D sketching can represent those details of 3D models which are very difficult to represent by words.

The main challenge in supporting these 3D shape-based similarity queries is to find a computational representation of shape (a shape descriptor) for which an index can be built and geometric matching can be performed efficiently. Generally speaking, the following properties are desirable for a shape descriptor. It should be,

- quick to compute
- concise to store
- easy to index
- invariant under similarity transformations
- insensitive to noise and small extra features
- independent of 3D object representation, tessellation, or genus
- robust to arbitrary topological degeneracies
- discriminating of shape differences at many scales

4 Retrieval Algorithm

As in many database retrieval applications, the retrieving algorithms for matching 3D shapes are motivated by two principal concerns. First, the algorithm needs to be discriminating. This means that it should be effectively distinguishing between different classes of shapes and returning those models in the database that most closely approximate the ones that a user would want. Second, the algorithms need to be efficient in both space and time. In particular, since many of the existant repositories index thousands, or even tens of thousands of models, the stored representation of a 3D model needs to be compact and the retrieval time needs to be fast enough to return results in real time. In practice, addressing the run-time efficiency requrement is done with the assistance of a shape descriptor. The shape descriptor is an abstraction of the 3D model, capturing salient shape information in a structure that is well-suited for comparison.

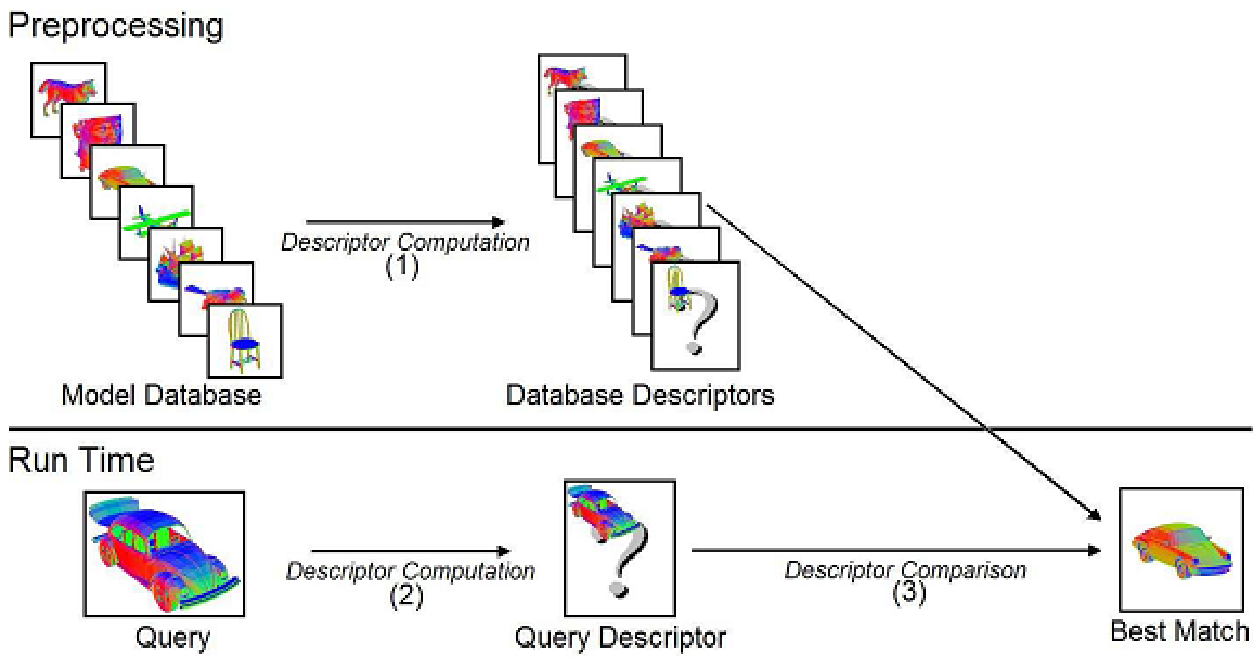

Figure 2: Retrieval Algorithm

The retrieval algorithm shown in Figure 2 works in two phases: the preprocessing phase and the runtime phase. In the preprocessing phase the shape descriptor computation is done for each model in the model database. At the runtime phase when user submits a query, shape descriptor computation is done for the submitted query and it is then compared with all other descriptors in the database using an matching method and the best matches are fetched.

5 Matching Methods

The query interfaces for 3D search can be based on text, 3D sketch or 2D sketch. The query interface may use the combination of these to improve the search results.

5.1 Text Matching Method

Searching based on text keywords is by far the most common type of query interface available on the web. A representative text document is created for each 3D model in our database, using several potentially relevant text sources such as model filename, model file contents, descriptive text of the hyperlink to the model file, full URL path to the model file, web page content, web page context and WordNet synonyms and hypernyms. User entered text keywords are matched to representative text documents, one for each 3D model in the database and best matches are fetched. Since it compares the text keywords, it is fast and best suited for an interactive search. But the precision of the search is less because most of the 3D models found on the web are poorly annotated.

5.2 3D Shape Descriptor Computation and Matching Method

The user gives a 3D sketch as the query to the search engine and search engine computes the shape descriptor of the 3D sketch. The computed 3D shape descriptor is compared with the preprocessed shape descriptors of the 3D models in the database. 3D sketching programs which are simple may not be efficient. Efficient 3D sketching require high levels of skill and consumes lot of time which makes it unsuitable for interactive search. So the selection of sketching programs should be done with utmost care.

Another option is to upload a 3D model file and compare the shape descriptor computed from this model file with the shape descriptors in the database. 3D file formats like VRML2.0, PLY, Wavefront OBJ can be used for this purpose. Since the shape descriptor computation from a 3D model file is easier searching can be done faster.

Two shape descriptors are compared by computing the Euclidian distance between them. Computation of the spherical harmonics shape descriptor is shown in Figure 3.

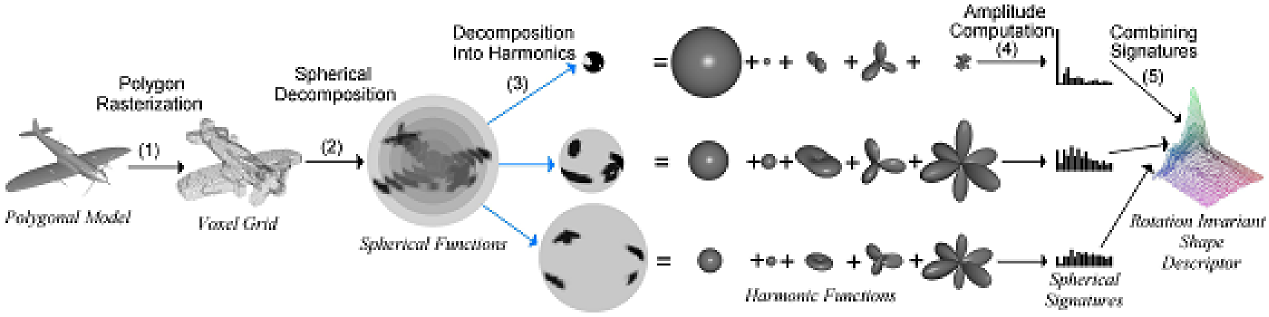

Figure 3: Computing the spherical harmonics shape descriptor.

The main steps for computing a spherical harmonics shape descriptor for a set of polygons are

1. First, the polygonal surfaces is rasterized into a $2R*2R*2R$ voxel grid, assigning a voxel a value of 1 if it is within one voxel width of a polygonal surface, and assigning it a value of 0 otherwise 1. To normalize for translation and scale, the model is moved so that the center of mass lies at the point (R, R, R), and scale it so that the average distance from non-zero voxels to the center of mass is $R = 2$.

2. The voxel grid is treated as a (binary) real-valued function defined on the set of points with length less than or equal to R and express the function in spherical coordinates. By restricting to the different radii a collection of spherical functions $f0, f1, ..., fR$ is obtained.

3. Using spherical harmonics, each function fr is expressed as a sum of its different frequencies.

4. Noting that the different irreducible representations are fixed under rotation, and noting that rotations do not change the L2 norm of functions, a rotation invariant signature for fr is defined.

5. Combining these different signatures over the different radii, a two-dimensional rotation invariant spherical harmonics descriptor for the 3D model is obtained, with the value at index $(r0, m0)$ corresponding to the length of the $m0^{th}$ frequency of the restriction of f to the sphere with radius $r0$.

5.3 2D Shape Descriptor Computation and Matching Method

Since it is very difficult for an average user to create a 3D shape query from scratch, a 2D free-form sketching interface is provided. User may create 2D sketches from three different views and these sketches are converted into boundary descriptors and compared with the stored boundary descriptors of models that are present in the database. Since drawing the front, top and side views are easier for a novice user, it is popular than 3D query interface. Combination of side and corner views gives the best search results among the 2D matching methods. Two descriptors are compared by computing the Euclidian distance between them. Computation of the shape descriptor for boundary contours is shown in Figure 4.

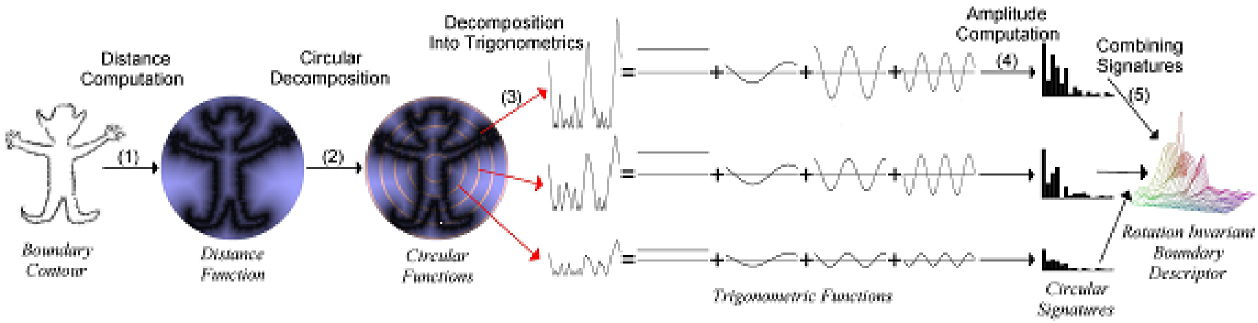

Figure 4: Computing the shape descriptor for boundary contours.

The main steps for computing shape descriptor boundary contours are

1. Compute the distance transform of the boundary contour.
2. Obtain a collection of circular functions by restricting to different radii.
3. Expand each circular function as a sum of trigonometric functions.
4. Using the fact that rotations do not change the amplitude within a frequency, define the signature of each circular function as a list of the amplitudes of its constituent trigonometrics.
5. Finally, combine these different signatures to obtain a 2D signature for the boundary contour.

6 Query Processing Performance

The response time a user experiences is the sum of the time it takes for the following operations:

1. Connecting to the web server and sending query data.
2. Executing a CGI script on the web server (which connects to and has to wait for the matching server).
3. Processing and matching of the query on the matching server.
4. Returning the results to the user.
5. Rendering the results web page on the user's machine.

The time taken in steps (1) and (4) depends on the available bandwidth between the user's machine and our web server, step (5) depends on the performance of the user's machine. Step (2) adds an estimated overhead of about 1-2 seconds. The accurate timings for step (3) can be calculated. The time between when a query arrives at the web server and when its results are ready is about 0.4 seconds on average. The time required for the text matching is 0.22 seconds which is the fastest matching method while the time required for 3D shape matching is 3.2 seconds which makes it the slowest matching method.

7 Conclusion

The most popular searching method among all the above given methods is the text query interface, because every user is used to that. The performance of text matching method is comparatively less because of the poor quality of text annotation of web models. The performance of 2D shape matching method is also less. But it shows better discrimination of the shape differences as the number of 2D sketches given by the user increases. 3D shape matching method outperforms the text matching method. But the main disadvantage of 3D matching method is that an average user will not be able to draw 3D sketches very effectively. The quality of search results can be increased by combining text with the 3D and 2D query interfaces. Combined text and 3D shape based matching methods will give the best precision for the search.

References

[1] M. Kazhdan J. Chen A. Halderman D. Dobkin T. Funkhouser, P. Min and D. Jacob. "A search engine for 3d models". *ACM Transactions on Graphics*, Vol. V(No. N, 10 202002).

[2] Thomas Funkhouser Michael Kazhdan and Szymon Rusinkiewicz. "Rotation invariant spherical harmonic representation of 3d shape descriptors". *Eurographics Symposium on Geometry Processing*, 2003.

[3] Princeton 3D Model Search Engine. *http://shape.cs.princeton.edu.*

[4] Princeton Shape Benchmark. *http://shape.cs.princeton.edu/benchmark.*