Ryutarou Ohbuchi, Masatoshi Nakazawa, Tsuyoshi Takei, Retrieving 3D Shapes Based On Their Appearance, accepted for the Proc. 5th ACM SIGMM Workshop on Multimedia Information Retrieval (MIR 2003), Berkeley, Californica, USA, November 2003.

Retrieving 3D Shapes Based On Their Appearance

Ryutarou Ohbuchi
University of Yamanashi
4-3-11 Takeda, Kofu-shi
Yamanashi-ken, Japan, 400-8511
+85-55-220-8570
ohbuchi@acm.org

Masatoshi Nakazawa
University of Yamanashi
4-3-11 Takeda, Kofu-shi
Yamanashi-ken, Japan, 400-8511
+85-55-220-8570
mnakazawa@ysk.co.jp

Tsuyoshi Takei
University of Yamanashi
4-3-11 Takeda, Kofu-shi
Yamanashi-ken, Japan, 400-8511
+85-55-220-8570
f8058@kki.yamanashi.ac.jp

ABSTRACT
In this paper, we propose an algorithm for shape-similarity comparison and retrieval of 3D shapes defined as polygon soup. One of the issues in comparing 3D shapes is the diversity of shape representations used to represent these "3D" shapes. While a solid model is well-defined and is easier to handle, others such as polygon soup poses many problems. In fact, a polygon soup 3D model most often does not define a 3D shape, but merely an illusion of "3D shape-ness" by its collection of independent polygons, lines, and manifold meshes. The most significant feature of our 3D shape similarity comparison method is that it accepts polygon soup and other ill-defined 3D models. Our approach is to use the rendered appearance only of the model as the basis for shape similarity comparison. Our method removes scale and positional degrees-of-freedom by using normalization, and the three rotational degrees of freedom by using a combination of discrete sampling of solid angles and a rotation-invariant 2D image similarity comparison algorithm. Evaluation experiments showed that, despite its simplicity, our approach worked quite well despite its simplicity.

Categories and Subject Descriptors
[I.3.5] Computational Geometry and Object Modeling] Curve, surface, solid, and object representations, [H.3.1 Content Analysis and Indexing] Abstracting methods, [I.4.8 Scene Analysis] shape

General Terms
Algorithm, Experimentation.

Keywords
Three-dimensional models, shape similarity search, polygon soup, geometric modeling, polygonal mesh, depth map.

1. INTRODUCTION
Proliferation of 3D models prompted development of the technology for effective content-based search and retrieval of three-dimensional (3D) models. A 3D model could be searched by textual annotation by using a conventional text-based search engine. This approach wouldn't work in many of the application scenarios. The annotations added by human beings depend on culture, language, age, sex, and other factors. It is also extremely difficult to describe by words shapes that are not in the well-known shape or semantic categories. It is thus necessary to have a content-based search and retrieval systems for 3D models that are based on the features intrinsic to the 3D models, most important of which is the shape [18, 21, 1, 8, 19, 20, 22, 4, 9, 16, 12, 6, 24, 25, 2, 11, 13, 17, 26, 5, 14, 15].

Current focuses in the research of shape similarity search of 3D models are the development of robust, concise, yet expressive *shape features*, and the development of *dissimilarity comparison* methods that are efficient and conform well to the human notion of shape similarity. One of the difficulties in shape feature extraction is that there are *diverse shape representations* and file formats, many of which are "*ill-defined*" and are mutually incompatible. For example, surface curvature, volume, and other mathematically nice properties can't be computed for a typical VRML model, which is an example of the *polygon-soup* 3D shape representation. Also, the very definition of similarity is the issue; for example, are two human figures, one with extended arms and the other with folded arms similar?

Diverse, mutually incompatible, and often "ill-defined" shape representations are the major cause of difficulty in finding effective shape descriptor for shape similarity comparison. Some of the existing shape similarity comparison methods assumed well-defined manifolds or even solid models, often targeting 3D CAD models [20, 9, 6, 12, 25, 2, 11]. Such mathematically nice properties as volume (e.g., [2]), surface curvature (e.g., 25), or such topological features as Reeb graph [6] can be computed for these "well-defined" shape representations.

Others tried to cope with ill-defined shape representations [18, 21, 19, 16, 13, 26, 5, 14, 15]. Majority of 3D models are not 3D solids, but a collection of disconnected discretised surfaces attempting to create an impression of 3D solid-ness. Each one of these surfaces may or may not be manifolds meshes, and what appears to be a surface may actually be a collection of numerous independent polygons. Many of the differential geometry operators can't be applied to these fragmented surfaces without major repairs that often require human intervention (e.g., [25]). Surfaces are often inconsistently oriented, or not oriented at all.

In this paper we propose a shape similarity comparison algorithm designed for ill-defined model representations, most notable of which is the polygon soup model. The method is *appearance-based*, in that the shape descriptor for a 3D shape is a set of multiple-view 2.5D images (i.e., depth images) created from the

3D model. Comparison of shape features is performed by using the 2D image similarity comparison algorithm by Zhang [27]. As far as the appearance of the model can be rendered as the surface depth image, e.g., by using the Z-buffer algorithm, the model can be compared. Our retrieval experiments using a 1213 model database showed that the method performed quite well despite its simplistic, rather brute force approach. Our proposed method outperformed the D2 shape function by Osada et al [16, 17], as well as our two previously published methods [13, 14], and is roughly comparable to our most recent method [15] based on 3D alpha-shapes.

The remaining part of this paper is organized as follows. We first review the previous work in the next section. We then present our appearance based shape similarity comparison method in Section 3, followed by its experimental evaluation results in Section 4. We conclude the paper with summary and future work in Section 5.

2. PREVIOUS WORK

There are four major steps in shape-based retrieval of 3D models from a 3D model database.

(1) Query formation: Form and present a query specifying a 3D shape or a 3D scene including multiple shapes.

(2) Feature extraction: Extract feature vectors from the model to be used for shape similarity (more often, dissimilarity) computation. Shape representation of the target 3D models influences the shape features that can be employed.

(3) Dissimilarity computation: Compute dissimilarity value between shapes. Usually, the dissimilarity values are expected to reflect human judgments.

(4) Retrieval: Efficiently retrieve the models having the lowest dissimilarity values form the database.

In this section, we mostly review the item (2) above, the feature extraction method. Shape features can be either geometrical or (surface) topological. Topological features require more or less well-defined models to compute the features. Geometrical features may be used for both well-defined (e.g., solid or manifold) and ill-defined (e.g., polygon soup) models.

3D geometrical features for well-defined models: Most of the other features are of geometrical nature. Some of the previous methods target more or less well-defined shape representations. Mukai et al [11] targets Constructive Solid Geometry (CSG) models, and exploits the CSG tree to decompose a shape into sub-components. However, their shape similarity is based on component-based geometrical similarity. Keim's method [8] accepted voxel model as inputs and performed voxel based similarity matching. Novotni [12] converted a boundary representation (B-rep.) model into voxel representation for a voxel-based shape similarity matching. The method by Corney et al [2] also assumed B-rep. solid as its input and employed, among others, ratios of surface and volume, for example, for a coarse shape matching. Zaharia et al [25] assumed a manifold surface and used a histogram of Gaussian curvature of the surface as the shape feature. As many models are not manifolds, Zaharia et al converted, with human intervention, models having topological flaws into well-defined manifold models.

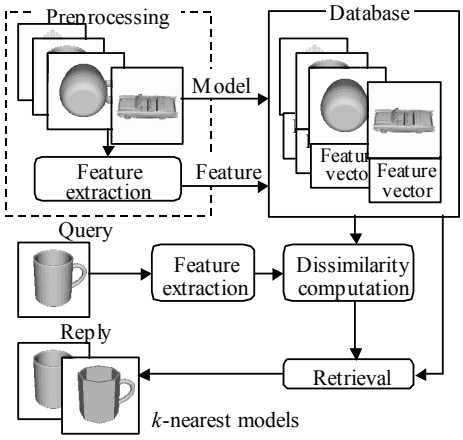

Figure 1. A shape similarity search database for 3D shapes.

3D geometrical features for polygon-soup models: Others tried to match shape represented using not-so-well-defined shape representations, e.g., polygon soup.

Some of the previous methods employed pose normalization prior to shape matching. Paquet et al [18] employed, after pose normalization, a set of geometrical features as well as color and other properties for their shape similarity search. Suzuki et al [21, 22] computed, after pose normalization, distribution of vertices in the uniformly subdivided axis-aligned grid. Zaharia [25] employed a 3D Hough transformation as the shape feature, after pose normalization. Both [22] and [26] took advantage of symmetries of their shape features so that their pose normalization can be simplified. Elad et al [4] also normalized pose and computed various moments from the points generated randomly on the surface. The method by Ohbuchi et al [13] first normalizes pose by using moments. Then it computes several inertial properties along the principal axis of the model. These and other methods that require pose normalization could run into trouble if pose normalization fails.

Other shape features targeting polygon soup are by themselves invariant to rigid, similarity or other transformations without requiring pose normalization [1, 16, 24, 17, 5, 14, 15].

Osada et al [16, 17] proposed a set of shape features that are invariant to rigid body transformation. The best performing one, the *D2 shape function*, is a 1D histogram of Euclidian distances between randomly selected pairs of points located on the model's surface. The points are generated at random location on each polygon. Their methods are quite robust, and for its simplicity, performed quite well both in terms of computational cost and performance.

Ohbuchi et al [14] improved Osada's D2 by adding mutual orientation of surfaces to the histogram, making the histogram 2D having both distance and angle axes. The shape descriptors, called *Angle Distance* (*AD*) and *Absolute Angle Distance* (*AAD*) histograms, significantly outperformed the Osada's D2. Ohbuchi recently proposed [15b] a 3D multiresolution approach to shape similarity comparison based on the *3D alpha shapes* [3]. They evaluated the effectiveness of their multiresolution approach by combining it with the AD shape descriptor above, calling the combination *alpha-multiresolution AAD* (AMR-AAD) shape descriptor. The experiment showed that the AMR-AAD

significantly outperformed the AD (and D2) shape descriptor. In this paper, we will compare the performance of the method proposed in this paper with those of the D2, the AAD, and the AMR-AAD.

Funkhouser, Min, and others developed a shape similarity search database for 3D models available for use via a sophisticated Web-based interface that accepts 2D sketches, 3D sketches (using Teddy [7]), and 3D example models [5, 10]. To compute their shape descriptor, they first scan-converts polygons of a 3D model into voxel buffer of size 64^3 in a Cartesian coordinate system. The shape descriptor is the coefficients of spherical harmonics approximating, at the concentric shells, the voxel distribution in the voxel buffer. Their method outperformed several other methods, including those by Osada [16, 17] and Ankerst [1]

3D topological features: Topological features have an advantage of being insensitive to position or orientation of 3D models [6, 20, 9]. Hilaga's method [6] captures surface topology of the shape, along with certain geometrical features. Hilaga's features are more or less invariant against both local and global geometrical transformations. For example, a pair of human figures with bent and stretched limbs will show very high similarity. However, to compute topology of surfaces or surfaces patches, the target models must be well-defined, e.g., solid models or at least oriented manifold models, so that such mathematically sound properties as surface curvature and volume can be computed.

3. AN APPEARANCE BASED SHAPE COMPARISON ALGORITHM

Overall structure of our 3D shape similarity search database system is shown in Figure 1. Actually, this diagram is generic enough so that it applies to a large portion of 3D shape similarity search system that employs 3D model as its query. The database stores the models as well as their shape feature, or *shape descriptor*, that are computed prior to the query. As a 3D model is given as a query, its shape feature is computed and compared with those stored in the database. The models having the smallest dissimilarity values among the features are retrieved and presented to the user. Typically, the system presents the user with *k* closest models to the query.

As the query is presented as an example 3D model, the system computes 3D shape similarity feature, or 3D shape descriptor. Figure 2 illustrates the steps taken to compute the shape descriptor. Our 3D shape similarity comparison method compares 3D models based on their appearance using depth images (or, 2.5D images). The system first normalizes the model size and places it at the origin. Then it computes a set of depth-buffer images of the 3D model that are viewed from 42 viewpoints in order to approximately and discretely cover all the possible view aspects of the model. The system then computes a shape feature vector per viewpoint. The shape feature for each view is a rotationally invariant generic Fourier Descriptor for 2D images developed by Zhang et al [27]. The set of 42 feature vectors comprises the shape descriptor of the model, which is called the *Multiple Orientation Depth Fourier Descriptor (MODFD)* in this paper. Dissimilarity computation among a pair of MODFDs is performed by computing the distance of all the possible combinations of 2 sets of feature vectors. That is, in the case of 42 feature vectors per shape descriptor, 42^2=1764 distance computations are done.

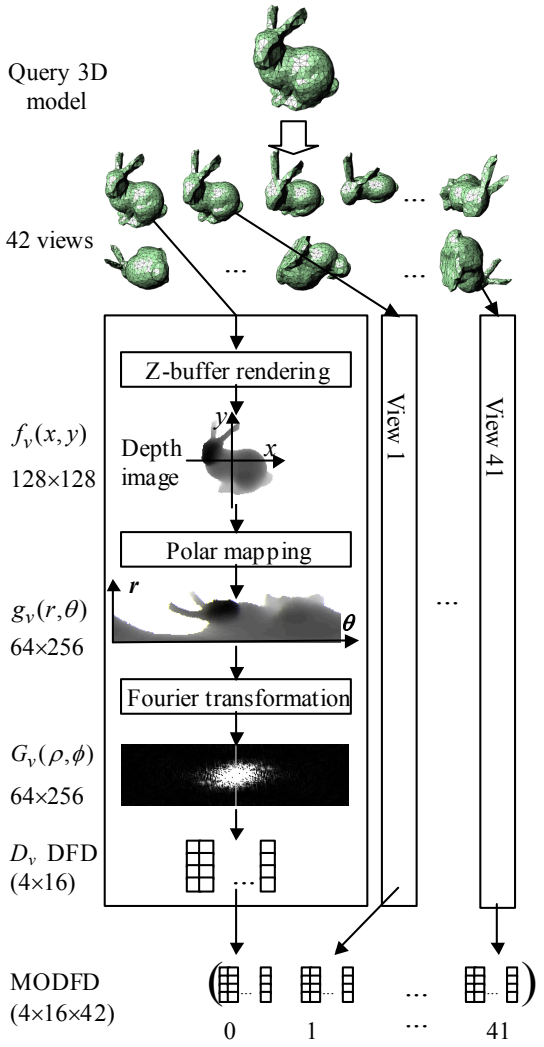

Figure 2. Computing the MODFD shape descriptor.

A characteristic of our MODFD shape descriptor is that it is very tolerant of not-so-well-defined models, i.e., non-solid models, models having multiple connected components, non-manifold models, etc. Essentially, the MODFD can be computed for any 3D model that can be rendered reasonably by using the Z-buffer algorithm. Another characteristic is that our shape descriptor does not require pose normalization. Consequently, we don't have to worry about matching failures due to failed pose normalization.

3.1 Computing the Shape Descriptor

We assume that the models to undergo similarity transformation. Then, a 3D model for which a shape descriptor is computed has 1 uniform scaling, 3 positional and 3 rotational DOF. We remove the first two, the 1 uniform scaling and 3 positional DOF by translating the barycenter of the model to the coordinate origin and uniformly scaling the model so that the model fits within a unit bounding sphere. Of the remaining 3 rotational DOF, 2 are dealt with by discretely and approximately enumerating all the possible view orientations about coordinate origin. The remaining 1 DOF, which is the "roll" of the camera, is removed by using a

rotation-invariant feature for 2D images developed by Zhang et al [27].

To normalize the position of the model, the center G of the model is computed as the midpoint of the minimum and maximum values of the vertex coordinates (i.e., axis-aligned bounding box). The size of the model is normalized to fit inside a unit sphere centered about the coordinate origin by uniform scaling. To scale, the coordinate values of the model is divided by the maximum of the distance to the vertices from the coordinate origin.

We then generate *depth* or *z*-value image $f_v(x,y)$ of the model from multiple viewpoints v that are equally spaced on the unit sphere. In our current implementation, there are 42 such views. We subdivided the *icosahedron* once by using the *Loop*'s subdivision scheme to generate the 80 faceted polyhedron having 42 vertices. Then 42 cameras are placed at each vertex looking at the coordinate origin (Figure 3). As mentioned before, we need not care about the camera's *roll* angle as we use the rotation invariant generic Fourier Descriptor [Zhang02].

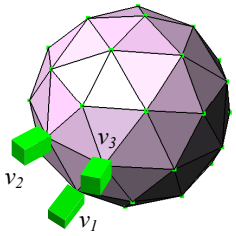

Figure 3. 42 cameras are placed at the 42 positions to generate depth (range) images..

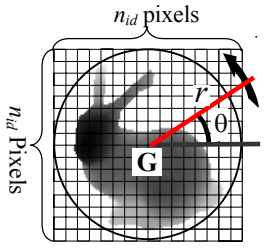

Figure 4. Computing the polar mapped version of the depth images.

Each depth images is generated using the orthographic projection with the bounding box (in the camera coordinate) set to tightly cover the unit bounding-sphere. The resulting *z* values have the range [0,1], in which 0 and 1 correspond, respectively, to the back and the front clipping planes. Using depth images has several advantages; a depth image (a 2.5D images) captures a part of 3D shape of the model, albeit from a viewpoint, and is not affected by lighting. Figure 5a shows an example of the standard flat-shaded image of a bunny model, and Figure 5c shows the corresponding depth image $f_v(x,y)$.

The last remaining DOF is the camera roll angle. We employ Zhang's *generic Fourier descriptor* [27] for similarity matching of 2D images for its rotation invariance and good matching performance. Using Zhang's method, our algorithm first maps the depth image $f_v(x,y)$ in the *x*-*y* Cartesian coordinate system into another image $g_v(r,\theta)$ in the *r*-θ polar coordinate system using a polar map (Figure 4). This converts the rotational DOF in the *x*-*y* Cartesian coordinate system into the translational DOF in the *r*-θ coordinate system. The image $g_v(r,\theta)$ is then Fourier transformed into $G_v(\rho,\phi)$. We use a standard Fast Fourier Transform algorithm for the computation.

$$G(\rho,\phi)=\sum_r\sum_\theta f(r,\theta)\exp\left[j2\pi\left(\frac{r}{R}\rho+\frac{\theta}{T}\phi\right)\right] \qquad (1)$$

The shape feature vector *Depth Fourier Descriptor (DFD)* D_v of a model for a viewpoint v is the low frequency part $m\times n$ coefficients of the 2D Fourier spectrum, defined by the following equation;

$$D_v=\left\{\frac{|G(0,0)|}{area},\frac{|G(0,1)|}{|G(0,0)|},\ldots,\frac{|G(0,n)|}{|G(0,0)|},\ldots,\frac{|G(m,0)|}{|G(0,0)|},\ldots,\frac{|G(m,n)|}{|G(0,0)|}\right\} \qquad (3)$$

This method by Zhang takes advantage of the translational invariance of the Fourier transform to remove the effects of rotation of the images. By using the low frequency component only of the Fourier transform, small variations in 3D model shapes, small variation in appearance (e.g., due to rotation) and noise in the images are ignored, using only the significant shape feature of the 2D images for the similarity (not exact) matching.

A shape descriptor for a model is a set of p=42 such DFD feature vectors, each of which is generated from depth images $f_i(x,y)$ taken from the viewpoint i, $i=1,2,...,p$. We call the set *Multiple Orientations DFD (MODFD)*.

In our current implementation, we generate the depth image for the viewpoint i, $f_i(x,y)$ with the resolution of 128×128 (i.e., $I_d=128$), which is converted to the polar mapped image $g_i(r,\theta)$ of resolution $64_{[r]}\times256_{[\theta]}$. The 2D Fourier transformed image $G_i(\rho,\phi)$ has the resolution of $64_{[\rho]}\times256_{[\phi]}$. We used only the lower-frequency part of the $64_{[\rho]}\times256_{[\phi]}$ 2D spectra for shape matching. This is because the shape similarity comparison that included the higher frequency component was too sensitive to the minute shape variations and hence lowered retrieval performance in our experiment. In the experiments that follow, we used the DFD feature vector of size $4_{[r]}\times16_{[\theta]}$ (i.e., m=3, n=15).

Figure 5 shows examples of DFD computation. Figure 5a shows the bunny model using a flat-shaded rendring. Figure 5b shows the bunny rotated 60 degree about the roll axis. Figure 5c and Figure 5d shows examples of the depth images $f_i(x,y)$ and $f_i'(x,y)$, in which the darker colours mean smaller *z* values. Both images are of the same bunny model; the $f_i'(x,y)$ used the same model as the $f_i(x,y)$ except that the model is rotated. Figure 5e and Figure 5f are the polar mapped images $g_i(r,\theta)$ and $g_i'(r,\theta)$ of the depth images $f_i(x,y)$ and $f_i'(x,y)$, respectively. It can be seen that the image $g_i'(r,\theta)$ is a translated image, in axis θ, of the image $g_i(r,\theta)$. Despite the rotation, which produced very different depth images $f_i(x,y)$ and $f_i'(x,y)$, the Fourier transformed images $G_i(\rho,\phi)$ (Figure 5g) and $G_i'(\rho,\phi)$ (Figure 5h) are almost identical, especially in the low-frequency part of the images

The dissimilarity between a pair of 3D models are computed as the distance between their respective MODFDs. Let $\mathbf{x}=(X_{i,j})$ and $\mathbf{y}=(Y_{i,j})$, $i=1,2,...,p$ and $j=1,2,...,q$, be the MODFDs of the models *X* and *Y*. Here, p=42 is the number of views, hence the number of DFDs per MODFD, and $q=64=4\times16$ is the dimension of each DFD vector. The distance $D(\mathbf{x},\mathbf{y})$ between the models *X* and *Y* is computed as below.

Figure 5a. The bunny model (flat shaded).

Figure 5b. The bunny model rotated by 60 degree about the roll axis.

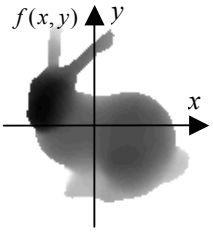

Figure 5c. A depth image $f_i(x,y)$ for the viewpoint i of the bunny model.

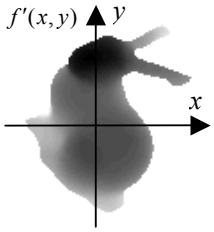

Figure 5d. A depth image $f_i'(x,y)$ of the rotated bunny model.

Figure 5e. $g(r,\theta)$ (Polar mapped $f(x,y)$)

Figure 5f. $g'(r,\theta)$ (Polar mapped $f'(x,y)$)

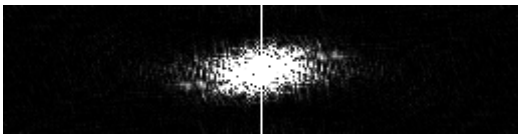

Figure 5g. $G(\rho,\phi)$ (Fourier transform of $g(r,\theta)$).

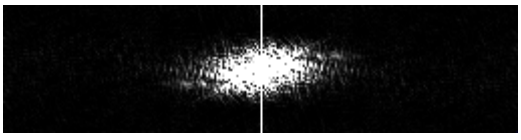

Figure 5h. $G'(\rho,\phi)$ (Fourier transform of $g'(r,\theta)$).

Figure 6 shows a quadrant, 32×128 coefficients, of the 64×256 Fourier coefficients $G_i(\rho,\phi)$. A small, low-frequency subset of the coefficients contains most of the power. As noted we currently use $4\times16=64$ low-frequency coefficients for as the DFD.

(1) Compute the L1 norms between an *i*th DFD X_i of the model X and every DFDs Y_i, $i=1,2,...,p$ of the model *Y*. The smallest of the computed distances is the distance $d(X_i,Y)$ of the DFD X_i of the model *X* to the model Y (Equation (4)).

$$d(X_i,Y)=\min_{1\le k\le q}\left(\sum_{j=1}^{q}\left|X_{i,j}-Y_{k,j}\right|\right) \tag{4}$$

(2) Repeat the computation p times for every DFDs X_i of the model *X*. The average of $d(X_i,Y)$ over $i=1,2,...,p$, $D(\mathbf{x},\mathbf{y})$, is the distance between the models *X* and *Y*.

$$D(\mathrm{x},\mathrm{y})=\frac{1}{P}\sum_{i=1}^{p}\left(d(X_i,Y)\right) \tag{5}$$

The search through the database is a linear time process, computing the distance $D(\mathbf{x},\mathbf{y})$ between the query model X and every model Y in the database. The database retrieves t models having the t smallest distances.

4. EXPERIMENTS AND RESULTS

We implemented the shape similarity matching method described in the previous section using C++ on the Linux operating system. We used the Mesa-GL software renderer to generate the depth image.

4.1 Experimental Methods

Performance evaluation for 3D shape similarity search is not an established art. The issues include many incompatible shape representations (voxel enumeration, polygon soup, solid, etc.), many file formats (VRML 97, 3DS, poly, etc.), lack of standardized test database, and the performance metric.

Unlike shape similarity search of 2D images, there is no established test database. Most researchers must contend with a few hundred free 3D mesh models collected from the Internet. We manually collected 1213 free 3D models that fall in the "polygon soup" category from the Internet. We then converted their format to that of VRML 97 using various conversion tools so that our parser would accept them.

To measure performance, we classified the 1213 models into 35 categories based on the judgment of two adult male persons. Of 1213 models, 861 are categorized into one of 34 "*known*" categories. The "known" categories includes such shape and/or semantic categories as "Car", "Lamp", "Chair", "Office-chair", "Humanoid", "4-legged animal", "Plane1", "Head", and "Mug". There is 35th category named the "Other", which included remaining 352 had-to-classify models (e.g., a pretzel-like model of a knot, etc.) In the experiments below, we queried models in the known categories only. If a query using a model from a known category retrieved models from the "Other" category, those models from the "Other" category are considered as failures.

Obviously, performance figures would depend very much on the model database and the categories used for the experiment. Changes in the models contained in the database or changes in the categorization will produce results different from those reported below.

We used several different performance measures to objectively evaluate our shape similarity matching methods; the *First Tier (FT)*, the *Second Tier (ST)*, and *Nearest Neighbor (NN)* match percentages, as well as the *recall-precision* plot.

Recall and precision are well known in the literature of content-based search and retrieval. Approximately, recall represents the completeness of the retrieval and precision represents the accuracy or "purity" of the retrieval. Let C be the number of relevant models in the database, that are, the number of models of the class Q to which the query belongs. Let N be the number of relevant models that are actually retrieved in the top A retrievals. Then, recall and precision are defined as follows;

$$\text{Recall}=\frac{N}{C}, \quad \text{Precision}=\frac{N}{A} \tag{6}$$

For example, the Precision becomes ideal value of 1.0 if the database returned *all* the models in the database, but such retrieval does not make sense. There is a trade-off relationship between precision and recall. In a recall-precision plot, a curve closer to the upper-right corner represents better performance.

FT, ST, and NN percentages are defined as follows. Assume that the query belongs to the class Q containing k models. The FT percentage is the percentage of the models from the class Q that appeared in the top (k-1) matches. As the query model is excluded from the computation, FT = 100% if (k-1) models from the class Q appeared in the top (k-1) matches. The ST percentage is similar to FT, except that it is the percentage of the models from the class Q in the top 2(k-1) matches. The NN percentage is the percentage of the cases in which the top matches are drawn from the class Q.

4.2 Retrieval Performance

We compared the performance of the MODFD method with our implementations of the D2 [16, 17], the *AAD* [14], and the AMR-AAD descriptors. We call our version of the D2 "*mD2*" as its implementation details are somewhat different from those of Osada's. For example, to generate points on the surfaces for computing the D2, our implementation used *Sobol*'s quasi-random sequence, instead of a pseudo-random sequence used by Osada, et al. The AAD was designed as an enhancement over the D2, taking into account not only the distance between the point pair but the angle formed by the surface normal vectors at the pair of points. According to our experiment, AMR-AAD performed the best, followed by the AAD, and then by the mD2 [14, 15].

Table 2 shows performance figures in terms of the FT, ST, and NN. Figure 8 shows the recall-precision plots, for the mD2, AAD, AMR-AAD, and the MODFD methods. To produce results presented in Table 2 and Figure 8, we queried all the 861 models in the "known" categories (that are, those not in the "Other" category) once, and checked the retrieved models against the predefined categories. The results shown in Table 2 and in Figure 8 are averaged over all the 861 queried models. The 352 models in the "Other" category are not queried. If a query using a model from one of the "known" categories produced models from the "Other" category, those models are considered as failures.

Table 2 shows that the MODFD has about 9% higher FT and ST figures than the mD2. Compared against the AAD, the MODFD performed 4% and 5% better, respectively, in FT and ST figures. The NN figure of the MODFD is more than 15% higher than that of the mD2, suggesting the higher sensitivity of the MODFD to shape details. The MODFD seems to outperform the runner-up AMR-AAD, but with a small margin. In fact, the precision of the AMR-AAD is better than that of the MODFD in the area where the recall value is higher, e.g., more than 0.6 or so.

Table 2. Overall retrieval performance comparing the MODFD, AAD, and mD2 shape descriptors.

Methods	Performance		
	FT	ST	NN
MODFD	29%	40%	54%
AMR-AAD	28%	40%	52%
AAD	24%	35%	43%
mD2	20%	31%	37%

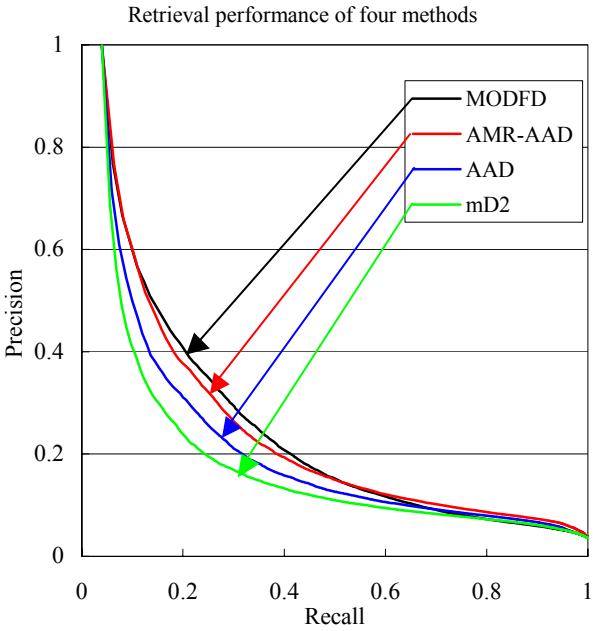

Figure 8. Overall recall-precision plot comparing the retrieval performance of the MODFD, AMR-AAD, AAD, and mD2 shape descriptors.

Figure 9. Querying an office chair model.

The performance of the MODFD method is slightly better than (but approximately equal to) that of the AMR-AAD method [15]. As both MODFD and AMR-AAD methods have many parameters to tune, their relative performance is not clear yet. It should be noted that the performance of the methods depends on the model category. For example, the MODFD performed better than the AMR-AAD in querying office chair models. But in querying four-legged animals, the AMR-AAD clearly won over the MODFD.

Figure 9 shows an example of retrievals using both MODFD and mD2 shape descriptors.

4.3 Computational Costs

Computational costs for the database query using the MODFD and the mD2 shape descriptors are compared in Table 5. The timings in Table 5 are the averages over all the 861 models in the "known" categories from the 1213 model database. We run the experiment on a pc with Athlon 1900+ CPU and 1.5GB RAM. Note that, for the OpenGL rendering, we did not use any hardware acceleration; we used a software renderer *MesaGL* (version 3.5). (http://www.mesa3D.org).

Table 5 shows that, as expected, the MODFD is slower than the mD2. Much time is spent on dissimilarity computation. On average, the system responds within about 5 s after presenting a query 3D model. In the worst case, the system took over 30s to produce results. The variation in time is due to the cost of rendering models having different polygonal complexity. The dissimilarity comparison time for a fixed-size database is constant. Table 6 shows the costs of MODFD (42 views) computations for models having different sizes. As noted above, we currently employ software rendering for the depth image generation. The MODFD computation time could be reduced drastically if we employed hardware acceleration for the OpenGL rendering API.

Potentially more serious than the cost of depth image generation is the cost of dissimilarity computation, especially if the database becomes large. There are several possible alternatives to make the dissimilarity computation more efficient. One is to reduce the number of DFDs, which is currently 42, by removing DFDs of a model that are very close and leaving only the "representative" DFDs for the model. This can be done by comparing all the DFDs of a model among themselves and removing those that are very close. Another possible alternative is to cull number of DFD-to-DFD comparison, for example, by removing comparison among DFDs of images having disparate non-background area size.

Table.5. Computational cost (Averaged over all the queries using 861 models in the "known" categories.).

	MODFD computation	Dissimilarity comparison
MODFD	3.3 s	11.1 s
mD2	0.4 s	1.6 s

Table 6. Model size and the MODFD feature descriptor computation time (using software rendering).

Model	Number of faces	Computation time
Horse	186,880	32.71 s
Rabbit	69,674	17.45 s
Old car	8,662	5.01 s
Icosahedron	20	2.72 s
Rectangle	6	2.65 s

5. Conclusion and Future Work

In this paper, we presented an appearance-based shape-similarity comparison algorithm for 3D shapes defined as polygon soup, to be used for shape-based retrieval of the 3D shapes.

The strength of the proposed method is its robustness in terms of the shape representations it accepts. The method can be used against topologically ill-defined mesh-based models, e.g., polygon soup models. This is because the method is appearance based; practically any 3D model that can be rendered into a 2D depth image can be compared. In fact, the model need not be polygon-based at all; a voxel-based model can be compared with polygon-based model if the former can be rendered into a 2D depth image. High DOF of the model posture is handled by combining (1) pose normalization for the 4 DOF of translation and scaling, (2) discrete enumeration of 2 rotational DOF, and (3) a rotation-invariant image shape feature for the last 1 rotational DOF. The method compares depth-images of the 3D objects by using the 2D image feature called *generic Fourier-descriptor* (GFD) developed by Zhang, et al. [27]. Computing distance among all the combinations of GFDs for a pair of models yields the distance (dissimilarity) value between the models.

The evaluation experiment showed that the method performed quite well despite its apparently simple approach. In the retrieval experiments, it performed significantly better than both our implementation of Osada's D2 shape function, the AAD shape descriptor [14], and about good as our AMR-AAD shape descriptor [15]. The MODFD method took, on average, about 5 seconds to retrieve 20 top matches from the 1213 model database after presenting the query 3D model. This is about 5 times more expensive than the AAD and the D2 methods, and is roughly as expensive as the AMR-AAD method to compute.

Among the top in the list of future work is the reduction of computational cost, possibly by using the methods suggested in Section 4.4. Hardware acceleration of depth-image rendering is a straightforward way to speed-up the algorithm. Speedup of dissimilarity computation is more important for a larger database. Possible methods to accelerate dissimilarity computation include, for example, a multiresolution approach to DFD representation and comparison, and a reduction in the number of DFDs per model based on similarity. We also need to perform more rigorous performance evaluation, e.g., by improving database and by adding better performance metric.

As a 3D model database system, one of our current weaknesses is the query interface. In addition to using a 3D model (given a priori) as the query, we would like to add 2D sketch, text annotation, and others to provide a of query methods, as is done by Funkhouser et al [5]. As our proposed method is based on depth images, a kind of 2D image, we hope to include a 2D sketch-based query interface soon.

6. ACKNOWLEDGMENTS

We thank Prof. Shigeo Takahashi and anonymous reviewers for their valuable comments. This research has been funded, in part, by the grants from the *Ministry of Education, Culture, Sports, Sciences, and Technology of Japan* (No. 12680432), *Okawa Foundation for Information and Telecommunications*, and the *Artificial Intelligence Research Promotion Foundation*.

REFERENCES

[1] M. Ankerst, G. Kastenmuller, H-P. Kriegel, T. Seidl, 3D Shape Histogram for Similarity Search and Classification in Spatial Databases, Proc. *Int'l Symp. Spatial Databases (SSD '99)*, Hong Kong, China, July 1999.

[2] J. Corney, H. Rea, D. Clark, John Pritchard, M. Breaks, R. MacLeod, Coarse Filter for Shape Matching, *IEEE CG&A*, pp. 65-73, May/June, 2002.

[3] Herbert Edelsbrunner, Ernst P. Mücke, Three-dimensional Alpha Shapes, *ACM TOG,* **13**(1), pp. 43-72, (1994)

[4] M. Elad, A. Tal, S. Ar., Content Based Retrieval of VRML Objects - An Iterative and Interactive Approach, Proc. *6th Eurographics workshop on Multimedia*, Manchester, UK., September 2001.

[5] T. Funkhouser, P. Min, M. Kazhdan, J. Chen, A. Halderman, D. Dobkin, D. Jacobs, A search engine for 3D models, *ACM TOGS*, 22(1), pp. 83-105, (January, 2003).

[6] M. Hilaga, Y. Shinagawa, T. Kohmura, and T. Kunii. Topology Matching for Fully Automatic Similarity Estimation of 3D Shapes. *Proc. SIGGRAPH 2001*, pp. 203-212, Los Angeles, USA. 2001.

[7] Takeo Igarashi, Hidehiko Tanaka, Satoshi Matusoka, Teddy: A Sketching Interface for 3D Freeform Design, Proc. *SIGGRAPH '99*, pp. 409-416, 1999.

[8] D. Keim, Efficient Geometry-based Similarity Search of 3D Spatial Databases, Proc. *ACM SIGMOD Int. Conf. On Management of Data*, pp. 419-430, Philadelphia, PA, 1999.

[9] D. McWherter, M. Peabody, W. Regli, A. Shokoufandeh, Transformation Invariant Shape Similarity Comparison of Solid Models, Proc. *ASME DETC '2001*, September 2002, Pittsburgh, Pennsylvania.

[10] Patrick Min, John A. Halderman, Michael Kazhdan, Thomas A. Funkhouser, Early Experiences with a 3D Model Search Engine, proc. Web3D 2003, pp. 101-112, Saint Malo, France, March 2003.

[11] S. Mukai, S. Furukawa, M. Kuroda, An Algorithm for Deciding Similarities of 3-D Objects, Proc. *ACM Symposium on Solid Modelling and Applications 2002*, Saarbrücken, Germany, June 2002.

[12] M. Novotni, R. Klein. A Geometric Approach to 3D Object Comparison. Proc. *Int'l Conf. on Shape Modeling and Applications 2001*, pp. 167-175, Genova, Italy, May, 2001.

[13] R. Ohbuchi, T. Otagiri, M. Ibato, T. Takei, Shape-Similarity Search of Three-Dimensional Models Using Parameterized Statistics, proc. *Pacific Graphics 2002*, pp. 265-274, October 2002, Beijing, China.

[14] R. Ohbuchi, T. Minamitani, T. Takei, Shape-Similarity Search of 3D Models by using Enhanced Shape Functions, accepted, proc. *Theory and Practice of Computer Graphics* 2003 (TP.CG.03), Birmingham, U.K., June 2003.

[15] R. Ohbuchi, T. Takei, Shape-Similarity Comparison of 3D Models Using Alpha Shapes, to appear, proc. *Pacific Graphics* 2003, October 2003, Canmore, Canada.

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

[17] R. Osada, T. Funkhouser, Bernard Chazelle, and David Dobkin Shape Distributions, ACM TOGS, 21(4), pp. 807-832, (October 2002).

[18] E. Paquet and M. Rioux, Nefertiti: a Query by Content Software for Three-Dimensional Databases Management, Proc. *Int'l Conf. on Recent Advances in 3-D Digital Imaging and Modeling*, pp. 345-352, Ottawa, Canada, May 12-15, 1997.

[19] E. Paquet, A. Murching, T. Naveen, A. Tabatabai, M. Roux. Description of shape information for 2-D and 3-D objects, *Signal Processing: Image Communication*, 16:103-122, 2000.

[20] W. Regli, V. Cicirello, Managing Digital Libraries for Computer-Aided Design, *Computer Aided Design*, pp. 110-132, Vol. 32, No. 2, 2000.

[21] M. T. Suzuki, T. Kato, H. Tsukune. 3D Object Retrieval based on subject measures, Proc. *9th Int'l Conf. and Workshop on Database and Expert Systems Applications (DEXA98)*, pp. 850-856, IEEE-PR08353, Vienna, Austria, Aug. 1998.

[22] M. T. Suzuki, T. Kato, N. Otsu. A similarity retrieval of 3D polygonal models using rotation invariant shape descriptors. *IEEE Int. Conf. on Systems, Man, and Cybernetics (SMC2000)*, Nashville, Tennessee, pp. 2946-2952, 2000.

[23] R. C. Veltkamp. Shape Matching: Similarity Measures and Algorithms, invited talk, Proc. *Int'l Conf. on Shape Modelling and Applications 2001*, pp. 188-197, Genova, Italy, May, 2001.

[24] D. V. Vranić, D. Saupe, and J. Richter. Tools for 3D-object retrieval: Karhunen-Loeve Transform and spherical harmonics. Proc. *IEEE 2001 Workshop on Multimedia Signal Processing*, Cannes, France, pp. 293-298, October 2001.

[25] T. Zaharia, F. Préteux, Three-dimensional shape-based retrieval within the MPEG-7 framework, Proc. *SPIE Conference 4304 on Nonlinear Image Processing and Pattern Analysis XII*, San Jose, CA, January 2001, pp. 133-145.

[26] T. Zaharia, F. Préteux, Shape-based retrieval of 3D mesh models, Proc. *IEEE ICME 2002*, Lausanne, Switzerland, August 2002.

[27] D. S. Zhang, G. Lu, Shape-based image retrieval using generic Fourier descriptor, *Signal Processing: Image Communication*, **17**(10), pp. 825-848, (November, 2002).