Ryutarou Ohbuchi, Takahiro Minamitani, Tsuyoshi Takei, Shape-similarity search of 3D models by using enhanced shape functions, *International Journal of Computer Applications in Technology* (*IJCAT*), pp.70-85, Vol.23, No. 2/3/4, 2005

Shape-Similarity Search of 3D Models by using Enhanced Shape Functions

Ryutarou Ohbuchi, Takahiro Minamitani, Tsuyoshi Takei
ohbuchi@yamanashi.ac.jp, minamitani@jex.co.jp, f8058@kki.yamanashi.ac.jp
Graduate School of Medical and Engineering Science, University of Yamanashi,
4-3-11 Takeda, Kofu-shi, Yamanashi-ken, 400-8511, Japan

Abstract

We propose a pair of shape features for searching surface-based 3D shape models based on their shape similarity. Either of the features is computed by first converting an input surface based model into an oriented point set model and then computing joint 2D histogram of distance and orientation of pairs of points. Advantages of the shape features are (1) they can be computed for non-solid or non-manifold models, (2) they are invariant to similarity transformation, and (3) they are tolerant of topological and geometrical errors and degeneracies. Experiments showed that, with only modest increase in computational cost, our shape features achieved significant performance improvement over Osada's D2, on which our features are based.

Keywords: content-based search and retrieval, geometric modeling, polygonal mesh.

1. Introduction

Proliferation of 3D models on the Internet and in in-house databases prompted development of the technology for effective content-based search and retrieval of three-dimensional (3D) models. A 3D model could be searched by its textual annotation by using a conventional text-based search engine. This approach wouldn't work in many of the application scenarios for the 3D shape model, however. The annotations added by human beings depend on culture, language, age, and other factors. It is also extremely difficult to describe by words a shape that is not in an well-known shape or semantic category. It is thus necessary to develop content-based search and retrieval systems for 3D models that are based on the features intrinsic to the 3D models, one of the most important of which is shape.

In the study of shape similarity search of 3D models, current focuses are on the development of robust, concise, yet expressive shape features, and on the development of similarity (or, dissimilarity) comparison methods that conform well to the human notion of shape similarity.

In developing a shape feature for 3D models, we first have to decide which class of 3D shape representation we are targeting. A 3D shape may be defined by using any of a number of shape representations, many of which are not mutually compatible. Some of the shape representations are mathematically well founded, allowing for computations of such well-defined properties as volume, surface curvature, or surface (or volume) topology. Other shape representations are less nicer. For example, a "polygon-soup" model is a topologically disconnected collection of independent polygons and/or polygonal meshes. Neither volume nor surface curvature can be computed for the model. As many of the VRML models and 3D models generated by using 3D

animation software are defined as polygon soup models, it is quite important to develop effective shape similarity comparison methods for this class of models.

Another important requirement for a 3D shape similarity comparison method is invariance of the method to a required class of geometrical transformations. Most of the time, an invariance to similarity transformation, that is, a combination of translation, rotation, and uniform scaling, is required for a 3D shape similarity comparison. A 3D model has a higher degrees-of-freedom (DOF) for their pose than a 2D shape model; a description of a similarity transformation requires 7 DOF. On the other hand, a 2D shape needs only 4 DOF to define a similarity transformation. A previous shape similarity comparison method either:

- employed an orientation insensitive shape feature
- performed pose normalization prior to applying a pose orientation sensitive shape feature.

Osada et al [1] proposed what they call *shape distributions*. Osada's shape distributions, a set of shape features, have the advantage of being invariant, without pose normalization, to similarity transformations. Furthermore, they are designed to be applicable to a not-so-well-defined mesh-based model, i.e., a polygon soup defining a non-solid object consisting of non-manifold surfaces, multiple connected components, and such degenerate surfaces as zero-area polygons. All of their shape features first converts the surface based 3D models into unoriented point set models. Then, various statistics are computed from the point set model. Among the proposed shape distributions, the *D2* showed the best retrieval performance despite its low computational cost. The D2 is also easy to implement, for it is a 1D histogram of distance between pairs of points in the point set model. However, the retrieval performance of the D2 is not sufficient, failing to distinguish shapes that are quite different.

In this paper, we propose a pair of shape features for comparing polygon soup models. Our shape features are based on the Osada's D2 shape function [1]. Both our shape features try to statistically capture surface orientation as well as surface distance of the model. Our method first convert a given surface model into oriented point set model, i.e., a set of points having 3D position as well as orientation normal vector. Then, the methods compute, as a feature, a joint 2D histogram of the distances and mutual orientations of the pairs of oriented points. Experimental evaluation showed that our shape features have significantly better retrieval performance than Osada's D2 shape function while having only slightly increased computational cost.

The paper is organized as follows. In the next section, we will review the previous work on 3D shape similarity search and retrieval, focusing on those methods that target polygon soup models. Our shape-matching algorithms are described in Section 3, and the method and results for the experimental evaluation of our algorithm are presented in Section 4. We conclude the paper in Section 5.

2. Previous Work

A method for shape similarity comparison of 3D models can be classified by the shape representation it is targeting. Some of the shape comparison algorithms assume well-defined shape representation, that are, 3D solid represented by using voxels, boundary representation, or constructive solid geometry [2, 3, 4]. Others assume topologically well-defined 2-manifold surfaces [5, 6]. However these methods can't be used to compare polygon soup models. In this section, we review shape similarity comparison methods for not-so-well-defined shape

representations, especially those for "polygon-soup" models.

Another possible classification is by the method used to achieve invariance of the shape comparison method to a class of geometrical transformations. To achieve geometrical transformation invariance, some methods employ pose normalization. To fully invert a similarity transformation, 7 DOF must be fixed; 3 for position, 3 for rotation, and 1 for scaling. Some methods normalize all the 7 DOF, while the other normalize a subset of the 7 DOF.

A pioneering work in 3D shape similarity search by Paquet et al [7] computed, after a pose normalization, a set of geometrical features. Their method also employed attributes, such as color of the model, for their shape similarity search. Another pioneering work by Suzuki et al [8] computed, after pose normalization, distribution of vertices in the uniformly subdivided axis-aligned grid. Suzuki et al tried to relate impression words and 3D shape by means of multi-dimensional scaling. Zaharia [9] employed a 3D Hough transformation computed from distribution of vertices of a model as its shape feature, while Elad et al [10] computed various moments from the points generated randomly on the surface. Elad's tried to match the human notion of shape similarity with that of mechanical distance by employing a learning classifier support vector machine. The method by Ohbuchi et al [11] normalizes pose, then computes moment of inertia, average and variance of distance from the three principal axes to the model surface. These methods employed principal component analysis of the covariance matrix of the point distribution of the model for their pose normalization. The points used to generate the covariance matrix may be the original vertices or they may be generated uniformly on the surfaces for the purpose of pose normalization.

Ankerst [12] proposed one of the first 3D shape similarity matching algorithms that targeted 3D molecular databases. One of their shape descriptor parameterizes the 3D space using concentric spherical shells, making the feature invariant to rotation. The spherical harmonics descriptor by Funkhouser, et al. [13] also employs the concentric-shell parameterization of the 3D space, after normalizing the position and scale. Funkhouser et al used the spherical harmonics descriptor to capture distribution of polygon for each shell in the frequency domain. The spherical harmonics shape feature is combined with the normalization of translation and uniform scaling to achieve invariance to all the 7 DOF of similarity transformation. The methods by Chen et al [14] and Ohbuchi et al [15] compares 3D shape by using a set of 2D images taken from multiple orientations. These two methods first normalize position and scale DOF to place a normalized model at the coordinate origin. Then, 2 out of 3 rotational DOF are approximated by a few dozen discrete viewing locations. The remaining 1 DOF of rotation is removed by using a rotation invariant shape feature for 2D images.

These methods that require pose normalization could run into trouble if pose normalization fails. For example, identifying and inverting translation typically employs computing a barycenter of the model and translating the barycenter to the origin of the coordinate system. Such normalization of position may not work well if the shape contains a geometrical "outliers". Similar problem could arise when normalizing rotation and scale.

A set of shape features proposed by Osada et al [1] is inherently invariant to similarity transformation so that they don't require any normalization. Among the features, the D2 shape "distribution", one of the simplest to compute, performed the best in terms of computational cost and retrieval performance. Beside its geometrical transformation invariance, their shape features are quite robust against noise in, or even total lack of, topology of polygons and meshes. Their

shape features are also robust against geometrical noise and degeneracies such as zero-area polygons.

The method proposed in this paper is an extension of Osada's method. We will describe the D2 shape feature and our shape features in the next section.

2. Proposed Algorithm

Figure 1 shows the structure of our proof-of-concept 3D model database system. We adopted the *query-by-3D-shape-example* approach, in which a user presents the system with an example 3D shape and asks for k most similar shape models in the database. The shape feature extraction algorithm accepts a 3D shape defined as a collection of polygons and polygonal meshes. It may contain non-manifolds or geometrically degenerate polygons. If the surface of the input model is known to be orientable, we employ *mutual Angle-Distance histogram* (*AD*) shape feature. If we can't assume the surfaces of the models to be properly and consistently oriented, we employ *mutual Absolute-Angle Distance histogram* (*AAD*) shape feature. Both are the extensions of the D2 [1] shape feature. Unlike the D2, which is a 1D histogram, both our AD and AAD are 2D histogram. For the dissimilarity computation among shape features, we compared a few different methods based on L1 norm, L2 norm and an elastic matching algorithm. The database itself is organized as a one-dimensional array, and no attempt has yet been made to speed up the database access by indexing and other methods.

In the following, we first explain the D2 shape function (Osada, et al. [Osada02]) on which our proposed shape features are based. Then, our proposed shape features, the AD and the AAD will be described.

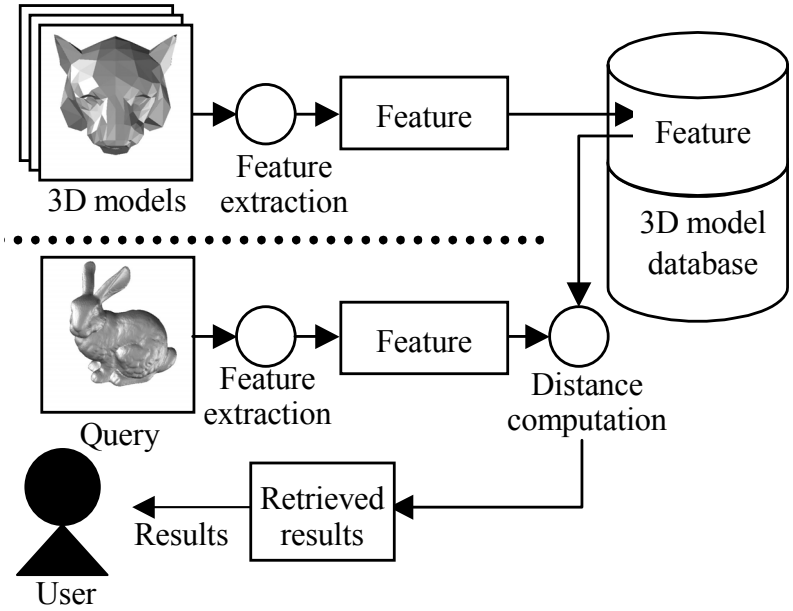


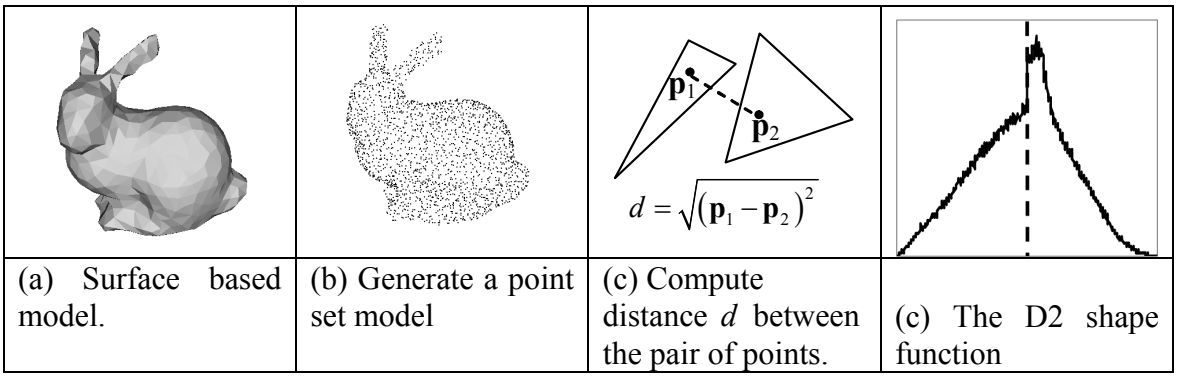
Figure 1. Structure of a shape similarity search system for 3D models.

2.1. Osada's D2

Osada et al proposed several different shape features in [1]. Advantages of these shape features

are that (1) they can be computed for a 3D polygon soup model that contain geometrical and topological noise, errors, and degeneracies, and that (2) the shape features is invariant, without pose normalization, to similarity transformation of the 3D model. Among the proposed shape descriptors, the D2 performed the best in terms of combined computational cost and retrieval performance.

To compute the D2 shape feature for a surface-based 3D model, the model is first converted into an unoriented point set representation by stochastically sampling the geometry of the model by generating points at random location on every surface of the model (See Figure 2). Distance is then computed for every possible pair of points generated, that is, $N_p\left(N_p-1\right)\big/2$ pairs for the N_p points generated. The D2 shape function is a 1D histogram generated by counting the population of the point-pair distances that falls within a certain distance interval. The D2 shape feature is insensitive to the variation in (or, total lack of) connectivity of polygons. As it is based on the unoriented point set representation, it is insensitive to the orientation of the surfaces in the original model.



(a) Surface based model.	(b) Generate a point set model	(c) Compute distance d between the pair of points.	(c) The D2 shape function

Figure 2. Computing the D2 shape function from the surface based 3D shape model.

We used Osada's method [1] to generate a point at a random location **P** on the surface of a triangle.

$$\mathbf{P}=\left(1-\sqrt{r_1}\right)\mathbf{t_1}+\sqrt{r_1}\left(1-r_2\right)\mathbf{t}_2+\sqrt{r_1}\left(r_2\mathbf{t}_3\right). \tag{1}$$

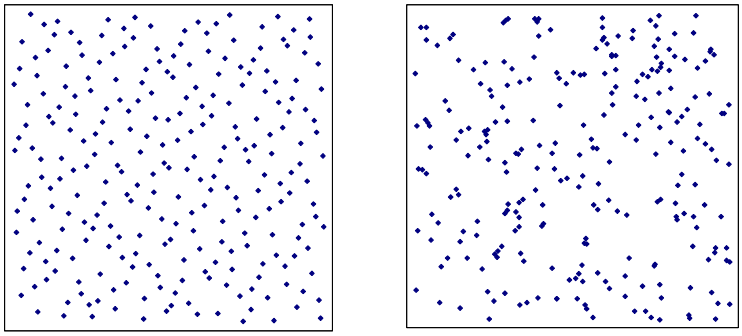
In the formula, $\mathbf{t}_1$, $\mathbf{t}_2$, and $\mathbf{t}_3$ are vertices of the triangle, and r_1 and r_2 are pseudo-random number sequences (PRNS) having the range $[0,1]$. If the model contained non-triangular polygons, they are triangulated prior to the point generation. The number n_i of points per ith triangle is determined in proportion to the area of the ith triangle by the following formula;

$$\begin{aligned} h_i &= N_p\, S_i \Big/ \sum_{i=1}^{M} S_i\,, \\ n_i &= \lceil h_i \rceil + \left(h_i - \lceil h_i \rceil\right) r_3. \end{aligned} \tag{2}$$

Here, S_i is the area of the ith triangle, M is the number of triangles for the model, and $r_3 \in [0,1]$ is a PRNS. The integral number n_i is probabilistically generated from the expected value h_i.

Our implementation of the D2 is somewhat different from the original D2, and we call our

version the *modified D2* (*mD2*). Instead of a PRNS used by Osada, we used a *Quasi-Random Number Sequence* (QRNS) by Sobol [16] for the r_1, r_2 above. The spatially uniformity of distribution of points on polygons are better if a QRNS is used, compared to the case in which a PRNS is used (See Figure 3.) That is, feature vectors generated by using a QRNS [16] tend to be more consistent, that is, low-variance, than those generated by using a PRNS. In the preliminary experiment, we experimentally compared the retrieval performance of the D2 shape feature computed by using the Sobol's QRNS [16] with that of the feature generated by using the PRNS `drand48()` function available in the standard C library. The QRNS version performed better than the PRNS version given a number of point N_p, especially if the N_p is small. We also performed the same experiment for the AD and the AAD shape features, and the QRNS versions of the AD and the AAD performed better than their PRNS counterparts. We thus chose the Sobol's QRNS for generating points for the mD2 shape feature as well as for the AD and the AAD shape features in the experiment described in Section 4.



(a) Sobol's QRNS. (b) drand48() PRNS.

Figure 3. Plots of 2D points that used the Sobol's quasi-random number sequence (a) compared with the plot that used the standard pseudo-random number generator function `drand48()` (b). The Sobol's QRNS samples the rectangle more uniformly than the PRNS `drand48()`.

For a proper comparison among the models having different size, the distance axis of the histogram needs to be normalized. We normalize the histogram by using the maximum, minimum, and the average of the point pair distance. Of the total I_d intervals of the histogram, $I_d / 2$ equally spaced intervals (bins) are allocated for distance values that range from the minimum to the average. The remaining $I_d / 2$ equally spaced intervals are allocated to the values from the average to the maximum. Consequently, the size of intervals in the upper half (i.e., above average) of the histogram is in general different from that of the intervals in the lower half (i.e., below average) of the histogram.

The dissimilarity $D_{mD2}^{L1}(\mathbf{x},\mathbf{y})$ of a pair of models A and B, whose features are 1D vectors $\mathbf{x}$ and $\mathbf{y}$, respectively, can be computed as follows;

$$D_{mD2}^{L1}(\mathbf{x},\mathbf{y}) = \sum_{i=1}^{I_d} |x_i - y_i| . \qquad (3)$$

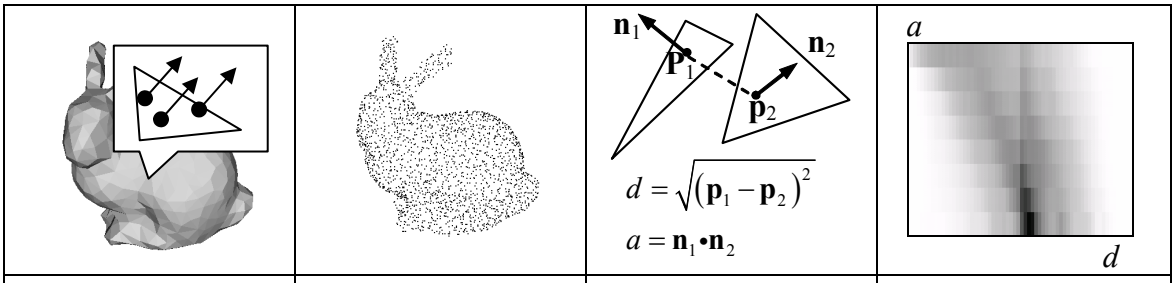
Here, I_d is the number of histogram intervals for the distance axis.

2.2. Shape Features AD and AAD

The AD and the AAD shape feature are 2D histograms of distances and angles formed by pairs of oriented points that are generated on the surfaces of the given 3D shape model. In computing the AD or the AAD shape feature, an oriented point set representation of the original (surface-based) 3D shape model is computed first. The orientations of the point are inherited from the surface normal vectors of the polygons on which the points are generated. The angle between a pair of points is actually represented as an inner product of the orientation vectors. The difference between the AD and the AAD is that the AAD ignores the sign of the inner product. Consequently, the AAD is more robust against models having inconsistent surface orientations than the AD.

2.2.2. AD

The AD shape feature measures, for each pair of points $\mathbf{p}_1$ and $\mathbf{p}_2$, the 3D Euclidian distance $d = \sqrt{(\mathbf{p}_1 - \mathbf{p}_2)^2}$ between the points and the inner product $a = \mathbf{n}_1 \bullet \mathbf{n}_2$ of the orientation vectors $\mathbf{n}_1$ and $\mathbf{n}_2$ of the points (Figure 4.) The points are generated in the manner identical to that of mD2, using the equation (1) and the Sobol's QRNS. Unlike the mD2, each point is oriented. Orientation of a point is inherited from the surface normal vector of a polygon on which the point is generated. Given the distance and the inner product for every pairs of the points, the AD is a joint 2D histogram of the distance d and the inner product a.



(a) Surface based model.	(b) Convert the model into an oriented point set.	(c) Compute distance d and the inner product a.	(d) The AD shape feature.

Figure 4. Computing the AD and AAD shape function from the surface based 3D shape model.

Similar to the D2 and mD2 shape features, a proper comparison of models having different sizes requires normalization of the distance axis of the AD histogram. We experimented with the four different normalization methods; (1) by maximum, (2) by average, (3) by median, and (4) by mode, of the distance values.

Normalization by Maximum: The maximum and the minimum of the distance values are found. Then, the interval the minimum distance and the maximum distances is subdivided into I_d equal width bins.

Normalization by Average: An average distance is computed. Then, given the total number of

distance bins I_d, a half ($I_d / 2$ equal sized bins) is allocated to the distance values above the average the other half (another $I_d / 2$ equal sized bins) is allocated to the distance values below the average.

Normalization by Median or Mode: These two normalization methods are similar to the "normalization by average" above, except that, instead of the average, either the median or the mode of the distance is used.

Inner product of a pair of orientation vectors always fall within the range $[-1,1]$, regardless of the size of the model. The structures of the bins for the angular axis are the same for all the normalization methods. The angular axis divides the interval $[-1,1]$ into I_a equal sized bins. The result is a 2D histogram having $I_a \times I_d$ elements for all the normalization methods.

Examples of the AD shape features computed by using the four normalization methods above are shown in Figure 5a-5d. In these figures, the darker the image, the higher the population is. These features are computed for the bunny model of Figure 2 by generating 2,000 points on the surface of the model for the 2D histogram having $I_d \times I_a = 64 \times 8$ elements.

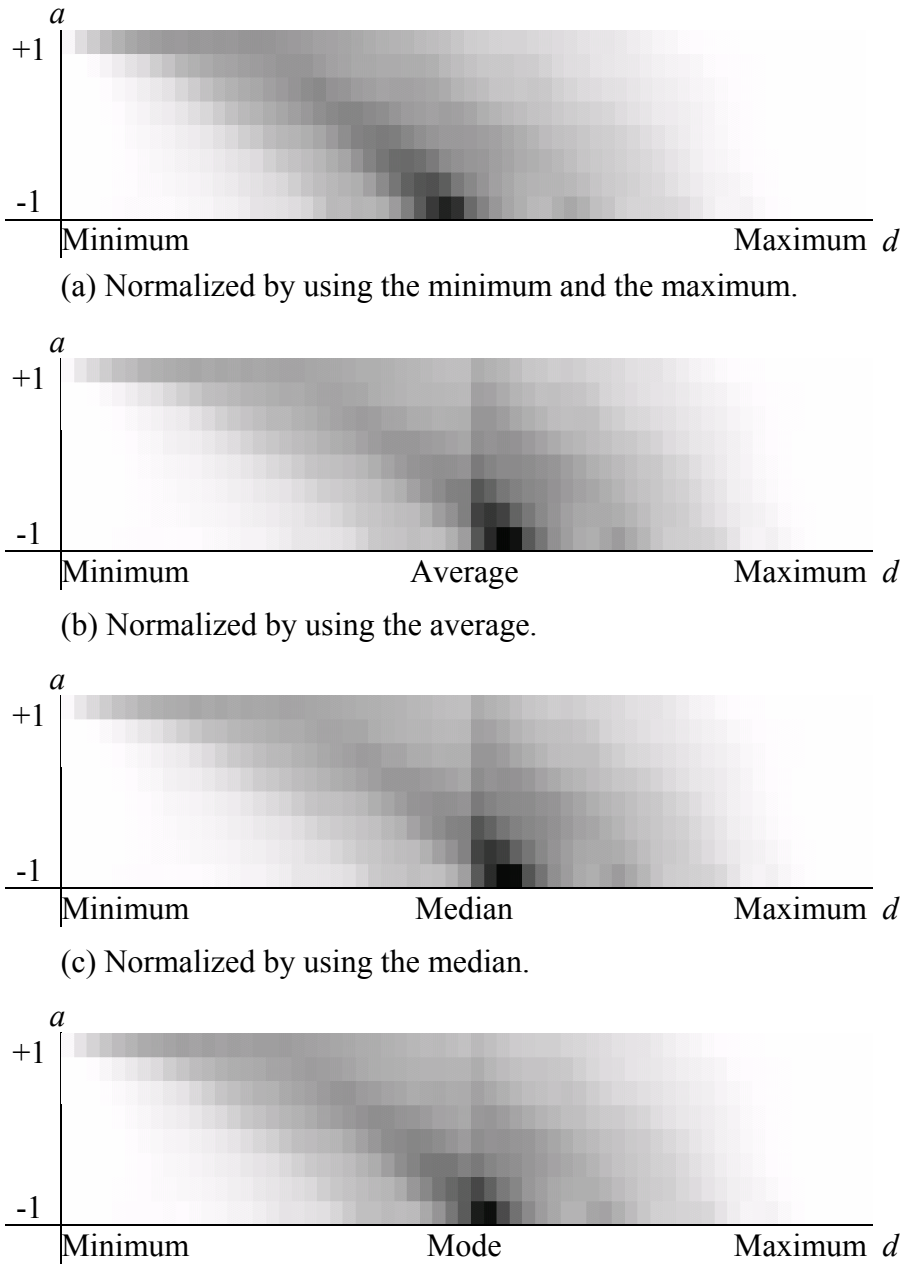


Figure 5. Examples of the AD shape feature, a 2D histogram, generated by using four distance normalization methods.

2.2.3. AAD

The AD shape feature described above is sensitive to the sign of the orientation vector of the point set model. If the models to be compared have a consistent surface orientation, e.g., a consistent traversal order of the vertices among polygons, the AD shape feature performs well. If, however, the database contains models having surfaces that are inconsistently oriented, the

performance of the AD shape feature suffers. Models generated by using different shape modeling tools might have different rules in determining surface orientations. Some of the models do not have coherent surface orientation at all.

The *mutual Absolute Angle and Distance* (*AAD*) *histogram* is computed similarly to the AD, except that the AAD ignores the sign of the inner product. This makes the AAD a more robust shape feature than the AD for the models having unoriented or inconsistently oriented surface orientations. The anglular axis of the 2D histogram of the AD takes the value in the range [0,1].

Figure 6 shows an example of the AAD shape feature computed for the same bunny model of Figure 2 using the maximum value normalization, $N_p = 2,000$ points and the histogram size of $I_d \times I_a = 64 \times 8$.

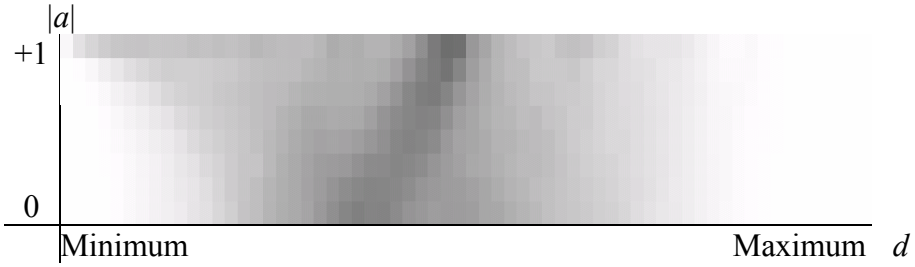


Figure 6. The AAD normalized by using the maximum.

2.3. Dissimilarity computation for the AD and AAD

We have implemented and compared three distance computation methods, the L1 norm (Manhattan distance), the L2 norm (Euclidian distance), and the elastic matching distance for the AD and the AAD shape features.

2.3.1. Dissimilarity measures using L1-norm and L2-norm

Assume that $\mathbf{X} = (x_{i,j})$ $(1 \le i \le I_d, 1 \le j \le I_a)$ and $\mathbf{Y} = (y_{i,j})$ $(1 \le i \le I_d, 1 \le j \le I_a)$ are the feature vectors for the models A and B, respectively. Note that a feature "vector" for the AD and the AAD shape features are in fact a 2D matrix of the dimension $I_d \times I_a$, in which I_d is the number of distance intervals and I_a is the number of angular (or, inner product) intervals. The L1 norm-based dissimilarity measure $D_{L1}(\mathbf{X}, \mathbf{Y})$ and the L2 norm-based distance $D_{L2}(\mathbf{X}, \mathbf{Y})$ for the AD and AAD shape features are defined as follows;

$$D_{L1}(\mathbf{X}, \mathbf{Y}) = \sum_{i=1}^{I_d} \sum_{j=1}^{I_a} \left| (x_{i,j} - y_{i,j}) \right|. \tag{4}$$

$$D_{L2}(\mathbf{X}, \mathbf{Y}) = \sum_{i=1}^{I_d} \sqrt{\sum_{j=1}^{I_a} (x_{i,j} - y_{i,j})^2} \ . \tag{5}$$

The distance axis is treated differently from the angular axis. The L1 or L2 distance among a pair of column vectors, each of which consisting of values from angular bins at the distance bin i,

is computed first. Then, a simple sum of these distance values over all the I_d intervals is computed.

2.3.2. Dissimilarity measure using elastic matching

We also computed elastic matching distance along the distance axis to compute the distance $D_E(\mathbf{X},\mathbf{Y})$ between the shape features $\mathbf{X}$ and $\mathbf{Y}$. It locally stretches and shrinks the *distance* axis of the histogram in order to a find minimal distance match. The elastic matching algorithm employs dynamic programming technique for an efficient implementation. In the past, elastic matching in the temporal axis had been used extensively in speech recognition in order to absorb variation in the speed of utterance.

If the matching is too elastic, a pair of shapes that are different could produce a small distance value. We compared the performance of linear and quadratic penalty functions, and chose the better performing quadratic penalty function, as shown in equation (8). The elastic distance $D_E(\mathbf{X},\mathbf{Y})$ among a pair of features $\mathbf{X}$ and $\mathbf{Y}$ is computed as follows.

$$D_E(\mathbf{X},\mathbf{Y}) = g(\mathbf{X}_n,\mathbf{Y}_n), \tag{6}$$

$$g(\mathbf{X}_n,\mathbf{Y}_n) = \min\begin{bmatrix} g(\mathbf{X}_n,\mathbf{Y}_{n-1}) + \Delta g(\mathbf{X}_n,\mathbf{Y}_n) \\ g(\mathbf{X}_{n-1},\mathbf{Y}_{n-1}) + 2\Delta g(\mathbf{X}_n,\mathbf{Y}_n) \\ g(\mathbf{X}_{n-1},\mathbf{Y}_n) + \Delta g(\mathbf{X}_n,\mathbf{Y}_n) \end{bmatrix}, \tag{7}$$

$$\Delta g(\mathbf{X}_i,\mathbf{Y}_j) = |i-j| \sqrt{\sum_{k=1}^{I_a} (x_{i,k} - y_{j,k})^2}\ . \tag{8}$$

3. Experiments and results

To evaluate the proposed shape features, we implemented the proof-of-concept 3D shape similarity database system using C++ on a Linux operating system.

3.1. Evaluation method

For the experiment, we used three different databases, each with its own "correct answer" categories.

- **Database A:** The database A consisted of 215 VRML models provided by Patrick Min and Prof. Funkhouser at the Princeton University. The model database is categorized *a priori* into 42, which are listed in Table 1.
- **Database B:** The database B consisted of 1,213 VRML models we have collected, modified, or created. We combined the database A above with the models from Johan Tangelder et al at the University of Utrecht [17]. We also collected more than 300 copyright free models from the Internet. To generate similar but different models, we modified some of the models (e.g., a bunny model) by using our shape morphing algorithm, a mesh simplification algorithm, or by adding geometrical noise. We also created a dozen or so new models, e.g., a "bunny house" model that contains the bunny model in a cube. Based on a consensus among

a few graduate students, we classified the models into 35 categories listed in Table 2. The 35 categories included the "other" category containing as many as 352 difficult to classify models. The database B contains a disproportionately large number of airplane models from the Utrecht database [17].

- **Database C:** The database C is the *Princeton Shape Benchmark* (PSB), a publicly available 3D model database [18]. The PSB version 1.0 consists of 1,814 models, which is divided into two groups; the training database (907 models) and the test database (907 models). We used the test database only for the experiment described in this paper. The 92 categories in the database C are listed in Table 3.

The retrieval experiment is as follows. We pick a query model q from a category C_q, and ask the system to find, in a database, models similar to q. If a retrieved model $r \in C_q$, it is a "success" retrieval. If $r \notin C_q$, then it is a failure. (In the case of the database B, we drew query models from categories other than the "other" category. Consequently, the larger the size of the "other" class, the lower various performance figures become.) Note also that many of the categories contained small number (2, 3, 4, or 5) of models.

As the objective measures of retrieval performance, we used the *First Tier* (*FT*), *Second Tier* (*ST*), and *Nearest Neighbor* (*NN*), as well as the *recall-precision* plot.

- **First Tier (FT):** Assume that the query belongs to the class C_q containing $k = |C_q|$ models. The FT figure is the percentage of the models from the class C_q in the top $(k-1)$ matches. As the query model is excluded, $(k-1)$ models from the class C_q in the top $(k-1)$ results produces the figure 100%.
- **Second Tier (ST):** The ST figure is the percentage of the models from the class C_q in the top $2(k-1)$ matches.
- **Nearest Neighbor (NN):** The percentage of the cases in which the top match is drawn from the query's class C_q.

Recall and *precision* are well known in the literature of content-based search and retrieval. Precision is the number of retrieved models that are in the class C_q divided by the number of all the retrieved models. Recall is the number of retrieved models that are in the class C_q divided by the number of models in the class C_q. In general, recall and precision are in trade-off relationship. If one goes up, the other usually comes down. As the objective of this database is similarity based search, if the similarity matching criteria is rather strict, the precision value goes up, while the recall value goes down. On the other hand, if the matching criterion is too loose, most of what has been retrieved is useless.

The FT, ST, NN figures as well as the recall-precision plots shown later are the averages produced by querying every model in the database once. As an exception, in the case of the database B, models in the "other" category are not queried.

Table 1. The 42 categories created for the test database A, which contains 215 models.

Class name	$\lvert C_q \rvert$	Class name	$\lvert C_q \rvert$	Class name	$\lvert C_q \rvert$
animal	5	head	4	plant	3
ball	6	helicopter	6	rifle	4
bed	2	human	17	shark	2
belt	3	lamp	9	skateboard	3
bicycle	2	leg	2	sofa	4
blimp	3	lightning	3	spaceship	6
boat	4	mechanoids	2	sub	3
bookshelf	2	misc	3	table	4
box	6	missile	7	tank	5
building	2	mug	5	tiefighter	2
car	7	openbook	4	tools	5
chair	9	pen	5	torus	2
claw	3	phone	4	tower	3
glove	2	plane	37	vase	4

Table 2. The 35 categories created for the test database B, which contains 1,213 models.

Class name	$\lvert C_q \rvert$	Class name	$\lvert C_q \rvert$	Class name	$\lvert C_q \rvert$
airship	5	delta-jet	74	plane2	49
animal4legs	22	dolphin	11	plane3	90
ball	33	head	23	plane4	54
biplane	24	helicopter	17	plane5	55
board-circular	6	holes	9	plane6	55
board-thick	16	humanoid	26	plane7	53
board	16	lamp	12	sofa	9
boat	10	missile	11	submarine	8
bunny	23	mug	4	sword	9
car	19	multi-fuselage	54	tank	7
chair	6	office-chair	8	other	352
cube	8	plane1	35		

Table 3. The 92 categories used in the database C, which is the "test" dataset containing 907 models of the Princeton Shape Benchmark [PSB] database.

Class name	$\|C_q\|$	Class name	$\|C_q\|$	Class name	$\|C_q\|$
Biplane	14	Book	4	Vase	11
Commercial	11	Barn	5	Mailbox	7
Fighter_jet	50	Church	4	Electrical_guitar	13
Glider	19	Gazebo	5	Newtonian_toy	4
Stealth_bomber	5	One_story_home	14	Bush	9
Hot_air_balloon	9	Skyscraper	5	Flowers	4
Helicopter	18	One_peak_tent	4	Potted_plant	26
Enterprise_like	11	Two_story_home	10	Barren	11
Flying_saucer	13	Chess_set	9	Conical	10
Satellite	7	City	10	Satellite_dish	4
Tie_fighter	5	Desktop	11	Large_sail_boat	6
Ant	5	Computer_monitor	13	Ship	11
Butterfly	7	Door	18	Submarine	9
Human	50	Eyeglasses	7	Billboard	4
Human_arms_out	20	Fireplace	6	Sink	4
Walking	8	Cabinet	9	Slot_machine	4
Flying_bird	14	School_desk	4	Staircase	7
Standing_bird	7	Bench	11	Hammer	4
Dog	7	Dining_chair	11	Shovel	6
Horse	6	Desk_chair	15	Umbrella	6
Rabbit	4	Shelves	13	Race_car	14
Snake	4	Rectangular	25	Sedan	10
Fish	17	Single_leg	6	Covered_wagon	5
Sea_turtle	6	Geographic_map	12	Motorcycle	6
Axe	4	Handgun	10	Monster_tuck	5
Knife	7	Hat	6	Semi	7
Sword	16	Hourglass	6	Jeep	5
Face	16	Ladder	4	Train_car	5
Hand	17	Streetlight	8	Wheel	4
Head	16	Glass_with_stem	9	Gear	9
Skull	6	Pail	4		

3.2. Selecting Parameters for the Proposed Shape Features

The AD and the AAD contains three parameters that affect their performance as well as computational cost; the number of points generated on each model N_p and the number of distance intervals I_d and angular intervals I_a . In this experiment, we varied these three parameters to find a best performing combination of parameters without too much in computation.

For the AD, we tested a set of 27 parameter combinations; $N_p = \{512, 1024, 2048\}$, $I_d = \{32, 64, 128\}$, $I_a = \{4, 8, 16\}$. For the AAD, we evaluated another set of 27 parameter combinations; $N_p = \{512, 1024, 2048\}$, $I_d = \{32, 64, 128\}$, $I_a = \{4, 8, 16\}$. In both of the cases, histograms are normalized by using the average, and the distance among features are computed by using the L2 norm. We compared the results using the retrieval performance in terms of FT, ST, NN, and the computational cost. If performances of two parameter combinations are equal, we chose the one having lower computational cost.

In terms of the number of points, $N_p = 1024$ performed better than $N_p = 512$. But the performances of $N_p = 1024$ and $N_p = 2048$ are indistinguishable. We chose $N_p = 1024$ for its lower computational costs.

The numbers of distance intervals I_d and I_a affected retrieval performance. If they are too small, the feature becomes insensitive to shape differences. If they are too large, the shape feature may be overly sensitive to minute shape differences, decreasing the overall similarity search and retrieval performance. And, the larger the number of intervals, the higher the storage cost and the distance computation cost. For example, when we increased the I_a from 8 to 16, the retrieval performance of the AD and the AAD shape feature decreased. What happened was that the features have become too sensitive. For example, two polygonal approximations of sphere, one having 20 facets and the other having 80 facets are determined to be different, although they are in the same "ball" category.

Overall, the combination of parameters we found to be the best are shown in Table 4.

Table 4. Parameters selected for the shape feature vectors.

Features	N_p	I_d	I_a
AD	1024	64	8
AAD	1024	64	4

3.3. Performance Comparison among the proposed methods

We compared the performance of various variations of our proposed methods using the database A. We performed three sets of comparisons;

(1) Comparison among the two shape features AD and AAD,
(2) Comparison among the four histogram normalization methods (by maximum, by average, by median, and by mode), and
(3) Comparison among the three distance computation methods, the L1-norm, the L2-norm, and the elastic matching distance.

Table 5 shows the results of the comparison (1) above. In this experiment, we used the average normalization method for the histogram normalization and the L2 norm for the distance computation. The figure shows that the AAD has the higher NN value, while its FT value is slightly lower, than the AD. Due to its smaller feature vector size (64×4 instead of 64×8), the AAD is somewhat faster than the AD at the database search step.

Table 6 shows the results of the comparison (2) above that compared among the histogram normalization methods. For this experiment, the AAD shape feature and the L2 norm are used. The result showed that the average based normalization performed the best. In terms of computational cost, maximum normalization was the least expensive, followed closely by the average normalization method.

Table 7 shows the results the comparison among the three distance computation methods (comparison (3) above.) In this experiment, we used the AAD shape feature and the average based normalization method. The figures in the table show that the L2 norm performed the best. The elastic matching did improve the performance for certain classes. But overall, the simple L2 norm performed better than the elastic matching.

Table 5. Performance comparison among the proposed shape features. (Database A).

Features	FT	ST	NN	Retrieval time
AD	39%	51%	56%	0.84s
AAD	38%	51%	60%	0.70s

Table 6. Performance comparison among the histogram normalization methods. (Database A.)

Normalization methods	FT	ST	NN	Feature computation time
Max-Min	36%	49%	58%	0.52s
Average	38%	51%	60%	0.54s
Median	36%	48%	58%	0.60s
Mode	33%	47%	54%	0.60s

Table 7. Performance comparison among the distance computation methods. (Database A.)

Distance computation methods	FT	ST	NN	Retrieval time
L1 norm	38%	49%	58%	0.68s
L2 norm	38%	51%	60%	0.70s
Elastic matching	37%	50%	54%	0.77s

3.4. Comparison with the other methods

We compared the performance of the AAD and the mD2 shape features by using the database A and the database B. The parameters used for this experiment are as follows;

- **mD2:** The number of points per model $N_p = 1024$ and the number of distance intervals $I_d = 512$. The normalization is performed by using the average based method [1]. Dissimilarity is computed by using the L1 norm.
- **AAD:** The number of points per model $N_p = 1024$ and the number of distance intervals $I_d = 64$, and the number of angle intervals $I_a = 4$. The normalization is performed by the average-based method. Dissimilarity is computed by using the L2 norm-based method.

The parameters for the mD2 are chosen so that its performance is the highest in our preliminary experiment. The choice of the AAD shape feature and the choice of various parameters of the AAD shape feature are due to the experiments described in the Section 3.3. The size, counted in number of numerical values to be stored, of a feature for the mD2 is 512, while that for the AAD is $64 \times 4 = 256$.

The FT, ST and NN figures resulted are shown in Table 8, Table 9, and Table 10, respectively, for the database A, the database B, and the database C. The recall-precision plots are shown in Figure 7, Figure 8, Figure 9, respectively, for the database A, the database B, and the database C. Table 8 and Table 9 also show computational costs in two parts; (1) the feature computation time, and (2) the total retrieval time, which is the sum of the feature computation time and distance computation time.

Table 8 shows that, using the database A, the AAD methods outperformed the mD2 in all of the FT, ST, and NN figures by the margin of 6% to 7%. Recall that these numbers are computed as averages over all the models and categories in the database. The AAD also outperformed the mD2 in the experiments that used the database B and the database C that are shown in Table 9 and Table 10. The performance advantage of the AAD is apparent in the recall-precision plots shown in Figure 7, Figure 8, and Figure 9 for the database A, B, and C, respectively. (In a recall-precision plot, a curve closer to the upper right corner means a better retrieval performance. Ideal retrieval performance would be the precision value of 1.0 for all the recall values.) The plots are smoother for the Figure 8 and Figure 9 than the Figure 7. This is because the database A has much smaller size than the database B and C.

In terms of computational cost, the feature extraction costs more for the AAD than the mD2. This modest increase in cost is well justified considering the performance advantage of the AAD. Furthermore, when it comes to the cost of distance computation, the AAD cost less to compare than the mD2 due to its smaller feature size.

Figure 10-13 show query examples using the mD2 (Figure 10a, 11a, 12a, and 13a) and the AAD (Figure 10b, 11b, 12b, and 13b) using the database B. In each figure, the upper left entry is the query, and the 5 by 4 matrix to the right shows the top 20 matches. Of the top 20 retrievals, the upper left corner shows the best match, which in every case is the query model itself. In these examples, the AAD appears to perform better than the mD2. In Figure 10, for example, the AAD appears to retrieve more “chair-like” models than the mD2. In the example of Figure 11 and Figure 12, compared to the mD2, the AAD retrieved models that appear to have “smooth and continuous” surfaces orientations..

Note that, in these figures, some of the images of the retrieved models contained black surfaces. Examples of black surfaces can be found in airplane models in Figure 11a. These are "backward-facing" polygons having flipped surface normal vectors. In these airplane models, one of the wings is made of backward-facing polygons presumably because of the "mirror and copy" operation used during the modeling process.

Table 8. Comparison among the mD2 and the AAD shape features using the database A (215 models).

Features	Performance			Computational cost	
	FT	ST	NN	Feature computation	Retrieval total
mD2	33%	44%	47%	0.41s	0.70s
AAD	38%	51%	60%	0.54s	0.70s

Table 9. Comparison among the mD2 and the AAD shape features using the database B (1215 models).

Features	Performance			Computational cost	
	FT	ST	NN	Feature computation	Retrieval total
mD2	20%	31%	37%	0.40s	2.00s
AAD	24%	35%	43%	0.52s	1.37s

Table 10. Comparison among the mD2 and the AAD shape features using the database C (907 models) (Computational costs are not available.)

Features	Performance		
	FT	ST	NN
mD2	19%	27%	36%
AAD	25%	35%	47%

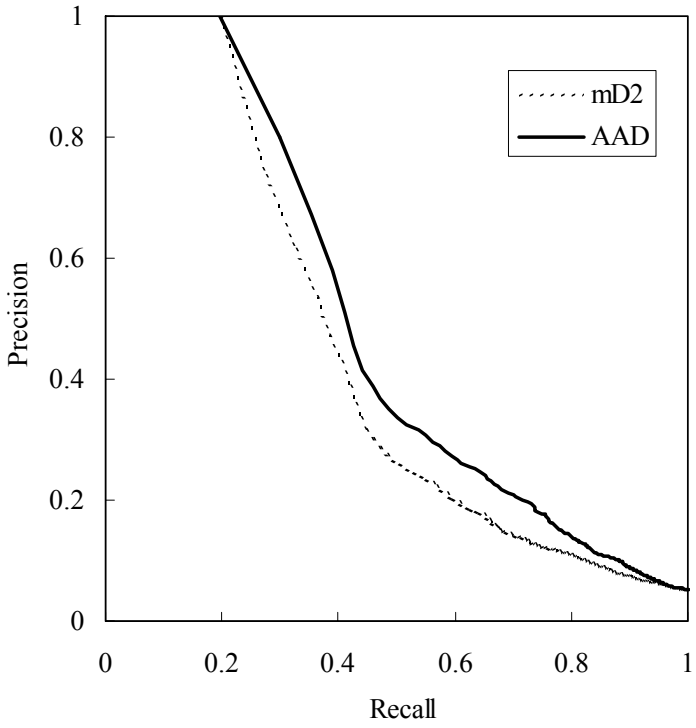


Figure 7. The recall-precision plot of the mD2 and the AAD shape features using the database A (215 models).

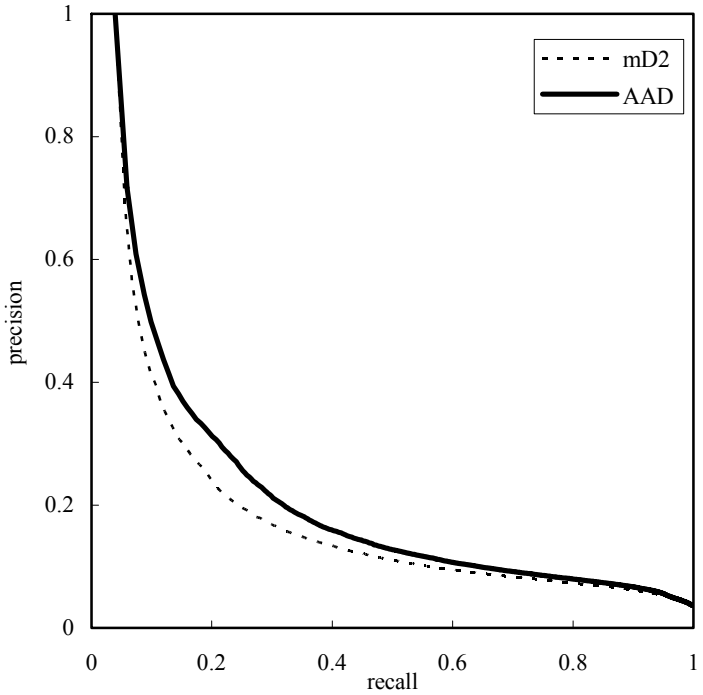


Figure 8. The recall-precision plot of the mD2 and the AAD shape features using the database B (1213 models).

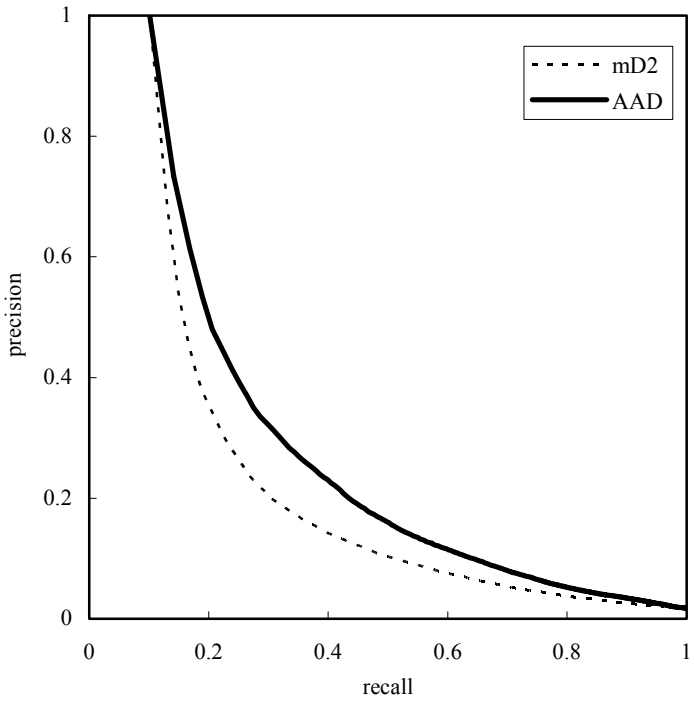
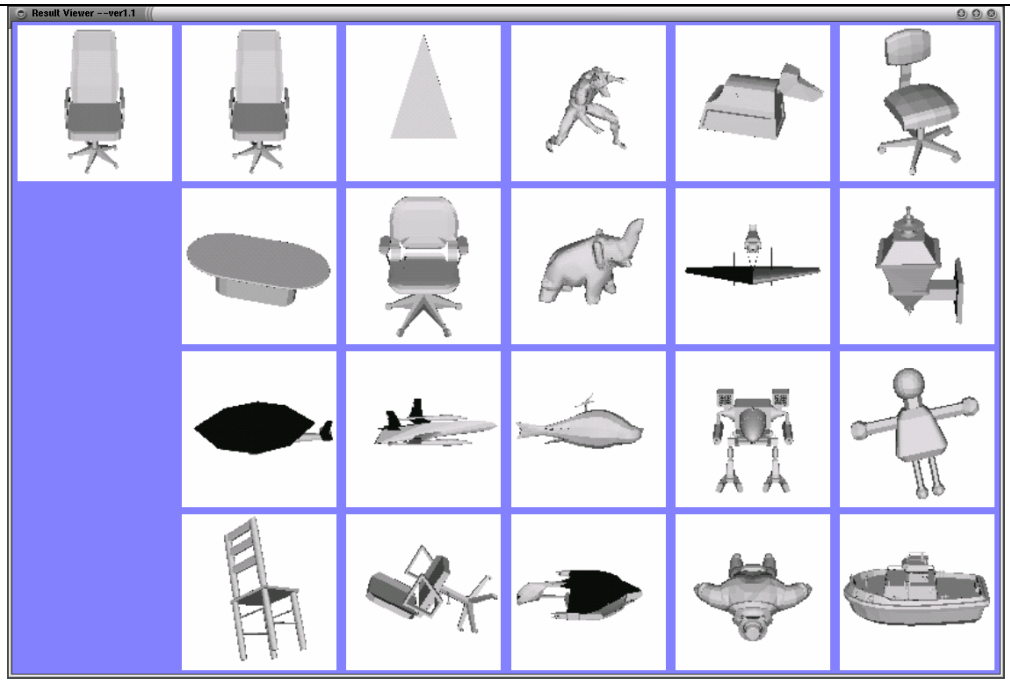


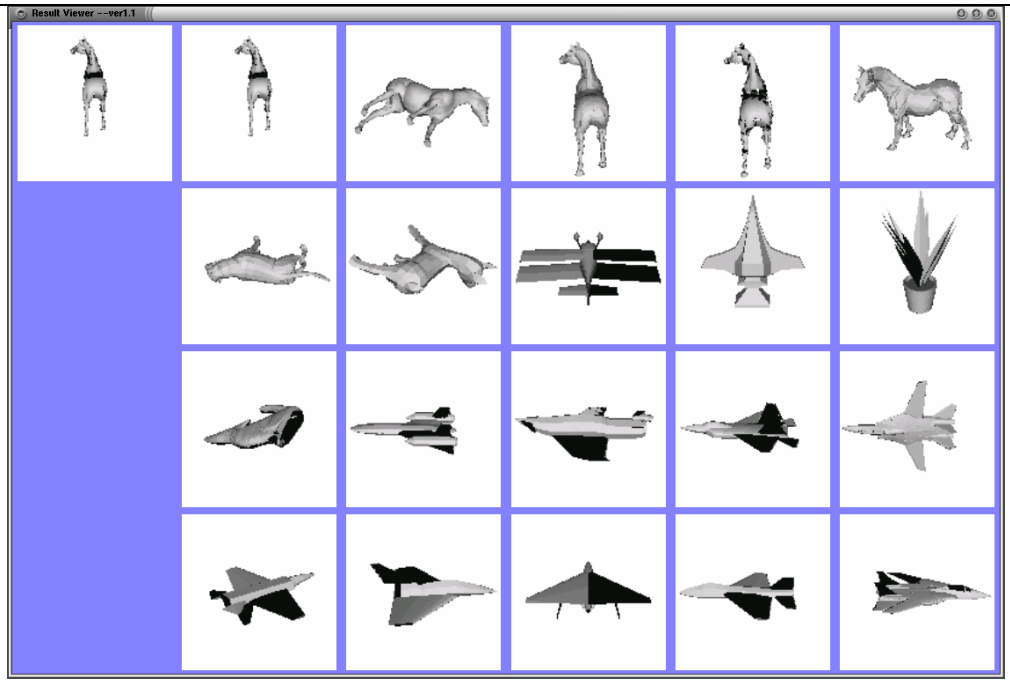
Figure 9. The recall-precision plot of the mD2 and the AAD shape features using the database C (907 models).



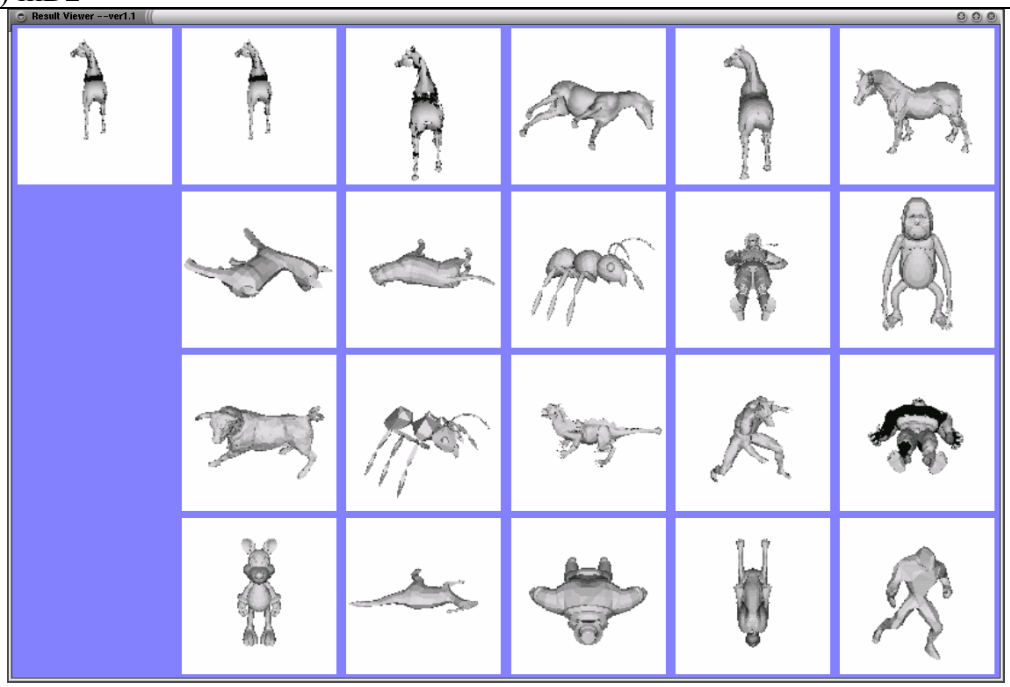
(a) mD2

(b) AAD

Figure 10. The retrieval example using the mD2 and the AAD shape features for the database B.

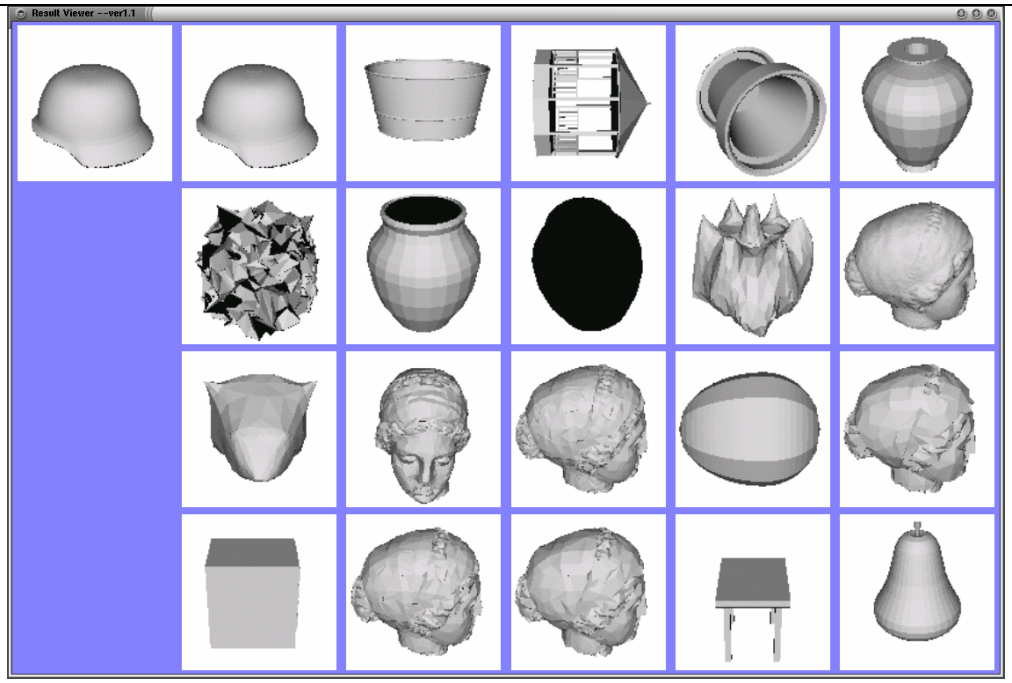


(a) mD2

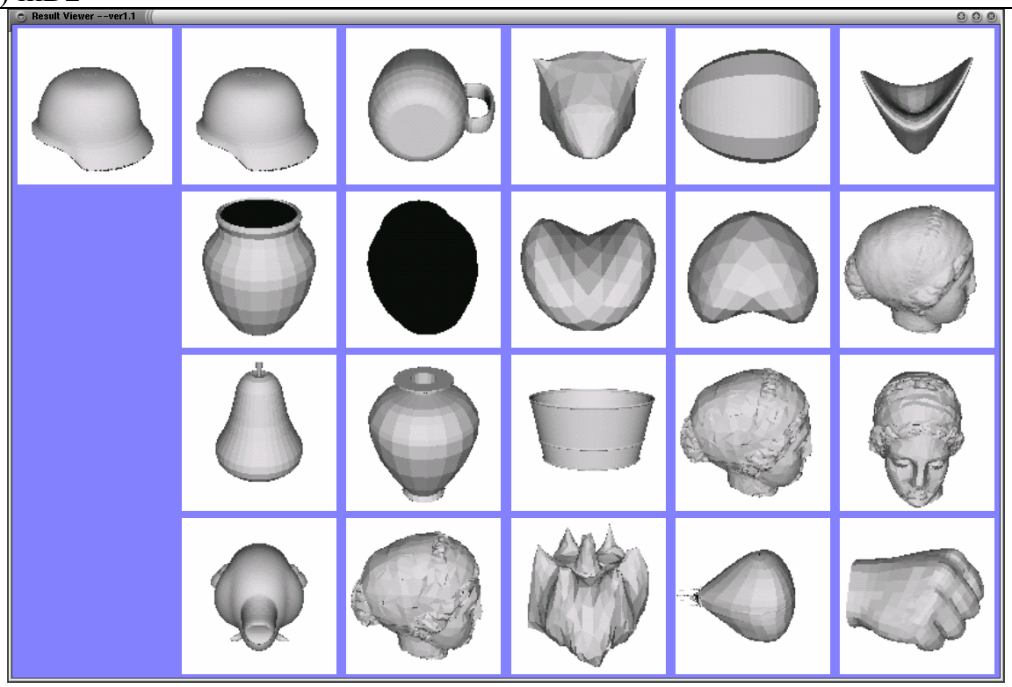


(b) AAD

Figure 11. The retrieval example using the mD2 and the AAD shape features for the database B.

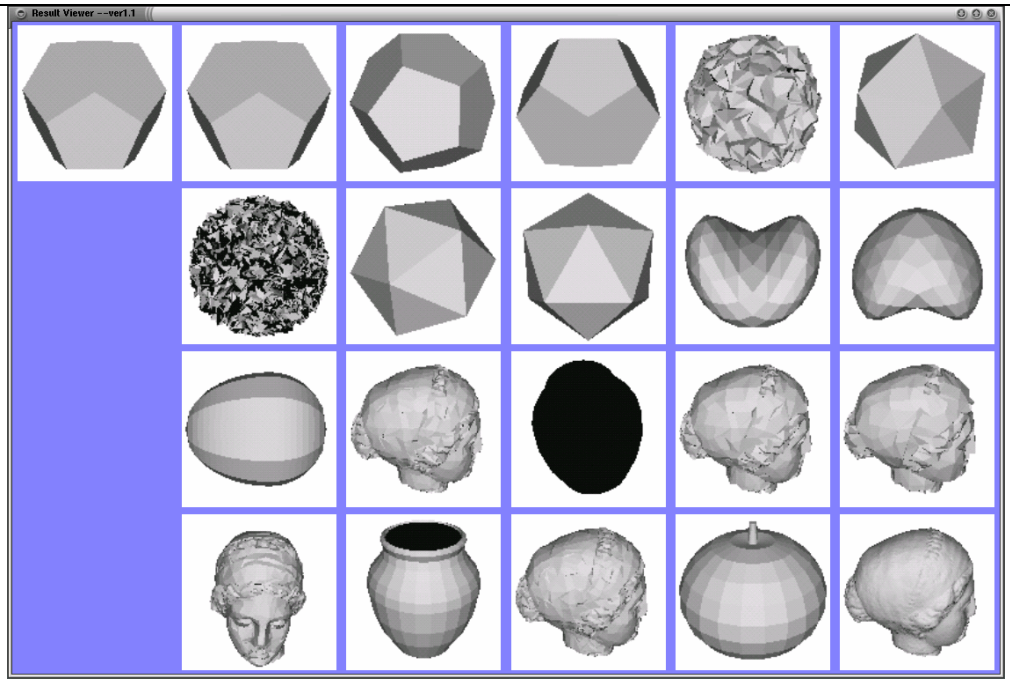


(a) mD2

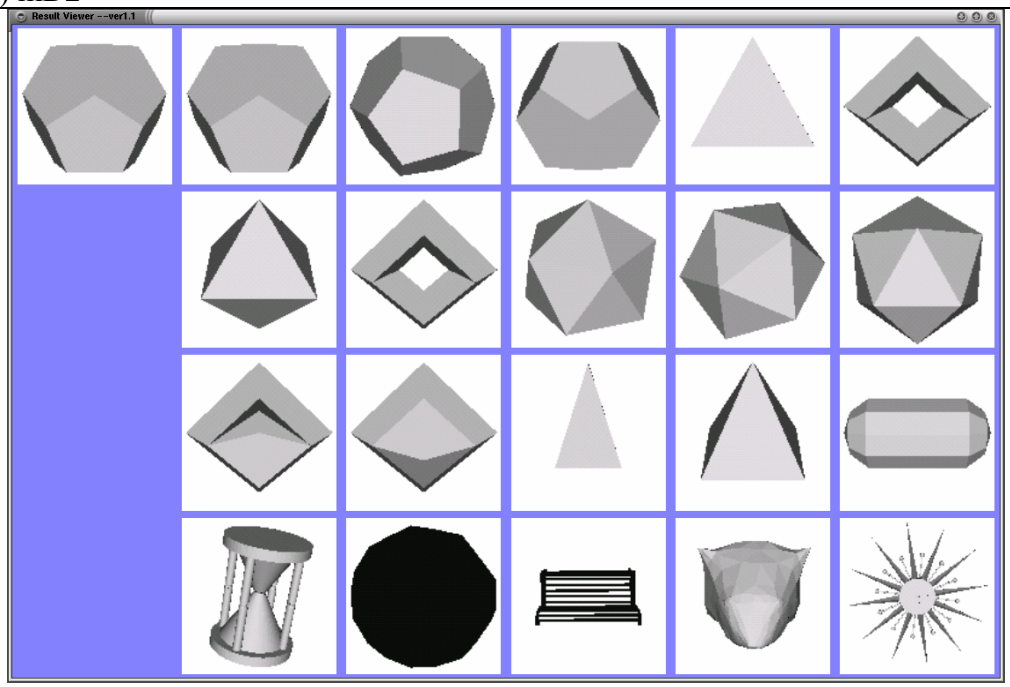


(b) AAD

Figure 12. The retrieval example using the mD2 and the AAD shape features for the database B.



(a) mD2



(b) AAD

Figure 13. The retrieval example using the mD2 and the AAD shape features for the database B.

4. Summary and conclusion

In this paper, we proposed and evaluated a pair of shape features for shape similarity search of 3D models. The shape features, called the AD and the AAD, are robust against topological and geometrical irregularities and degeneracies, which make them applicable to VRML and other so called "polygon soup" models. They are also invariant to similarity transformation, a quality valuable in comparing 3D shape models. While the AD and the AAD have computational cost somewhat higher (about 1.5 times) than the D2, they significantly outperformed D2 in our retrieval experiments. Although a direct comparison has not been made, the AD and the AAD might have the performance lower than that of the more elaborate methods, such as [13] and [14]. However, the computational costs of AD and AAD are significantly lower than these methods. The AD and AAD could thus be useful for a quick pre-screening of 3D shapes.

As a future work, we would like to improve our shape feature, for example by adding some form of multi-resolution approach to matching 3D shapes. We also would like to explore a hybrid shape feature that combines, possibly adaptively, shape features having different characteristics.

Acknowledgements

We thank Prof. Thomas Funkhouser and Patrick Min for providing us with the model database. We thank Prof. Shigeo Takahashi for insightful discussion and providing us with the software toolkit gmtools on which a part of our experimental system is based. This research has been funded by the Ministry of Education, Culture, Sports, Sciences, and Technology of Japan (No. 12680432), as well as by the grants from Okawa Foundation for Information and Telecommunications, and Artificial Intelligence Research Promotion Foundation.

References

[1] R. Osada, T. Funkhouser, Bernard Chazelle, and David Dobkin Shape Distributions, *ACM TOG*, 21(4), pp. 807-832, (October 2002).
[2] 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.
[3] D. McWherter, M. Peabody, W. Regli, A. Shokoufandeh, Transformation Invariant Shape Similarity Comparison of Solid Models, Proc. *ASME DETC '2001*, September 2002, Pittsburgh, Pennsylvania.
[4] S. Mukai, S. Furukawa, M. Kuroda, An Algorithm for Deciding Similarities of 3-D Objects, Proc. *ACM Symposium on Solid Modeling and Applications 2002*, Saarbrücken, Germany, June 2002.
[5] 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.
[6] T. Zaharia, F. Préteux, Three-dimensional shape-based retrieval within the MPEG-7 framework, Proceedings *SPIE Conference 4304 on Nonlinear Image Processing and Pattern Analysis XII*, San Jose, CA, January 2001, pp. 133-145.
[7] 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.
[8] 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.
[9] T. Zaharia, F. Préteux, Shape-based retrieval of 3D mesh models, *Proc. IEEE ICME 2002*, Lausanne, Switzerland, August, 2002.
[10] M. Elad, A. Tal, S. Ar., Content based retrieval of VRML objects: an iterative and interactive approach**,** Proc. *sixth Eurographics workshop on Multimedia 2001,* pp. 107-118, 2001.
[11] 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.
[12] 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.
[13] T. Funkhouser, P. Min, M. Kazhdan, J. Chen, A. Halderman, D. Dobkin, D. Jacobs, A search engine for 3D models, *ACM TOG*, 22(1), pp. 83-105, (January, 2003).
[14] Ding-Yun Chen, Xiao-Pei Tian, Yu-Te Shen, Ming Ouhyoung, On Visual Similarity Based 3D Model Retrieval, *Computer Graphics Forum*, Vol. 22, No. 3, pp. 224-232, (2003) (also as the proc. of EUROGRAPHICS 2003.)
[15] Ryutarou Ohbuchi, Masatoshi Nakazawa, Tsuyoshi Takei, Retrieving 3D Shapes Based On Their Appearance, Proc. *5th ACM SIGMM Workshop on Multimedia Information Retrieval (MIR 2003)*, pp. 39-46, Berkeley, Californica, USA, November 2003.
[16] W. H. Press et al., *Numerical Recipes in C-The Art of Scientific Programming*, 2nd Ed., Cambridge University Press, Cambridge, UK, 1992.

[17] Tangelder, et al.
http://www.cs.uu.nl/centers/give/imaging/3Drecog/3Dmatching.html
[18] Princeton Shape Benchmark, http://shape.cs.princeton.edu/search.html