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

[1]Ryutarou Ohbuchi, [1]Takahiro Minamitani, [1]Tsuyoshi Takei
ohbuchi@acm.org, f9084@kki.yamanashi.ac.jp, f8058@kki.yamanashi.ac.jp
[1] Computer Science Department, University of Yamanashi, 4-3-11 Takeda, Yamanashi-shi, Japan

Abstract

In this paper, we propose a pair of shape features for shape-similarity search of 3D polygonal-mesh models. The shape features are extension of the D2 shape functions *proposed by Osada et al. [Osada01, Osada02]. Our proposed shape features are tolerant of topological variations and geometrical degeneracies. Our shape feature is also invariant to similarity transformation. Experiments showed that, with only modest increase in computational cost, our shape feature achieved significant performance improvement over Osada's D2.*

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

1. Introduction

Proliferation of 3D models in the 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 [Paquet97, Suzuki98, Keim99, Elad00, Paquet00, Regli00, Suzuki00, McWherter01, Osada01, Novotni01, Hilaga01, Veltkamp01, Vranic01, Zahaira01, Corney02, Ibato02, Mukai02, Ohbuchi02, Osada02, Zahaira02, Funkhouser03, Tangelder03].

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

Compared to 2D shape similarity matching, 3D models have higher degree-of-freedom for their pose; orientation about three axes and position add up to 6 degrees-of-freedom. Previous methods either used orientation insensitive shape features [Regli00, Hilaga01, Osada01, Osada02, McWherter02, Funkhouser03], or performed pose normalization prior to using pose orientation sensitive shape features [Paquet97, Suzuki98, McWherter01, Novotni01, Zaharia01, Mukai01, Corney02, Ohbuchi02].

Diverse and often "ill-defined" shape representations are major cause of difficulty in finding an effective shape feature. Consider the VRML, one of the most popular shape representations. It is a collection of independent polygons, polygonal meshes, predefined parametric primitives such as cones and spheres, polylines, points, and other geometric objects. Such 3D models are often called "polygon soup". Volume of objects in VRML can't be computed for they are not solids. Surfaces in VRML are often not manifold, precluding application of many differential geometry operators, such as the computation of curvature. Some of the existing methods assumed well-defined manifolds or even solid models, often targeting 3D CAD models [Regli00, McWerter01, Hilaga01, Novotni01, Zaharia01, Corney02, Mukai02]. Others tried to cope with ill-defined shape representations such as VRML models [Paquet97, Suzuki98, Paquet00, Osada01, Osada02, Ohbuchi02, Zahaira02, Funkhouser03, Tangelder03].

Osada et al [Osada01, Osada02] proposed what they called shape functions. Osada's shape functions have the advantage of being invariant to similarity transformation. Furthermore, they are designed to be applicable to not-so-well-defined meshes, i.e., polygon soup that may contain non-solid objects, non-manifold surfaces, multiple connected components, and such degenerate surfaces as zero-area polygons. Of several shape functions, the D2 showed best retrieval performance for its low computational cost. However, the shape functions often fail to distinguish shapes that are quite different.

In this paper, we propose a pair of shape features for polygon soup models. Our shape features are based on Osada's D2 shape function [Osada01, Osada02], having identical robustness characteristics. However, our shape features are more sensitive to shape variations by measuring not only the distance but mutual orientation of the surfaces on which a pair of points are located. 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. We review the previous work on 3D shape similarity search in the next section. Our shape-matching algorithms are described in Section 3, and extensive experimental evaluation results of our algorithm are presented in Section 4. We conclude and discuss future work in Section 5.

2. The shape similarity comparison algorithm

In our prototype content-based 3D model database system (Figure 1), an entry in the database stores a 3D model along with a pre-calculated feature vectors for the model.

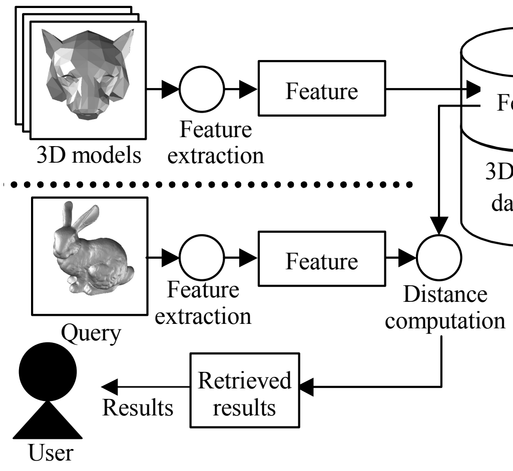


Figure 1. Overview of the shape similarity search system.

Query formation: We adopted the simplest of the query formation methods, the query-by-example-3D-shape approach. A user presents the system with an example 3D shape and asks for *k*-nearest shapes.

Shape representation: The database accepts a 3D shape defined as a collection of polygons and polygonal meshes. It may contain non-manifolds or geometrically degenerate polygons. One of our shape features also allows models having unoriented surfaces.

Feature vector: If the surface of the input model is orientable, we employ *mutual Angle-Distance histogram (AD)*. If we can't assume the surfaces of the models to be properly and consistently oriented, we employ *mutual Absolute-Angle Distance histogram (AAD)*. Both are the extensions of Osada's D2 [Osada02] shape function. Unlike Osada's D2, which is a 1D histogram, both our AD and AAD are 2D histogram.

Dissimilarity computation: We compared a few different methods to compute distance, or dissimilarity, between a pair of shape features. In addition to the simple L1 norm (Manhattan distance) and L2 norm (Euclidian distance) between the vectors, we tried elastic matching.

Indexing and retrieval: As the testbed for the shape features and dissimilarity computation methods, our proof-of-concept system employs no indexing. In the current implementation, the database itself is organized as a one-dimensional array, and no attempt has been made to speed up the database access by indexing and other methods.

2.1. Shape Features

As mentioned before, our shape features AD and AAD are based on Osada's D2 [Osada02]. Of several different what they called *shape functions*, the D2 performed the best in terms of combined computational cost and retrieval performance.

Our AD and AAD shape feature are 2D histograms, in which the additional dimension is an angle formed by the surface normal vectors of the surface the pair of points are located. The mutual surface orientation is actually computed as an inner product of the surface normal vectors. The difference between AD and AAD is that the AAD ignores the sign of the inner product. Consequently, AAD is more robust against models having inconsistent surface orientations.

2.1.2. Osada's D2

As mentioned before, our shape features AD and AAD are extensions of Osada's D2 function [Osada02]. The most favorable qualities of the D2 is its topological and geometrical robustness, and the lack of need for pose normalization.

To compute the D2 shape function for a 3D model, points are generated at random location on every surface of the model. Then, distance is computed for every possible pair, *i.e.*, $N_p(N_p-1)/2$ pairs for N_p points generated (Figure 2a). The 1D feature vector of D2 shape function is a histogram generated by counting the population of pairs that falls within a certain distance interval. It is robust against variation in tessellation, not sensitive to connectivity of surfaces nor surface orientation.

To generate points on triangles, we adopted the method by Osada et al. [Osada01], which uses the following formula to generate a point **P** at a random location on a triangle.

$$\mathbf{P} = \left(1-\sqrt{r_1}\right)\ \mathbf{t}_1 + \sqrt{r_1}\ \left(1-r_2\right)\ \mathbf{t}_2 + \sqrt{r_1}\ \left(r_2 \cdot \mathbf{t}_3\right). \quad (1)$$

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). If the model contained a non-triangular polygons, they are triangulated prior to the point generation.

In our implementation of the D2, which we call *modified D2* (*mD2*) to distinguish from Osada's, we used the *low-discrepancy sequence* or *quasi-random number sequence* (QRNS) by *Sobol* [Press92], instead of the PRNS. Compared to PRNS, the *Sobol's* QRNS produces feature vectors that are more consistent, *i.e.*, low-variance. That is, using the Sobol's sequence, generated points are more spatially uniformly distributed on the triangle. We compared the retrieval performance using the PRNS (`drand48()`) and the *Sobol's* QRNS. We chose the QRNS for generating point for the mD2, as well as the AD and AAD methods in the experiment described in Section 4.

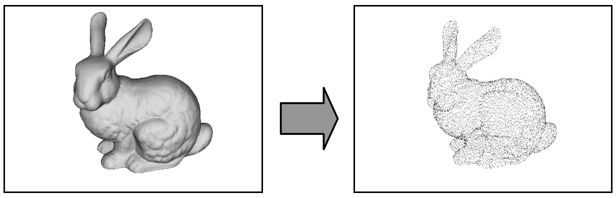


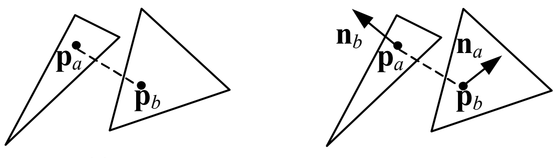
Figure 2. Points generated on the surface of a model.

For a proper comparison among the models having different size, the histograms need 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 are allocated for distance values that ranges from the minimum to the average. The remaining $I_d/2$ equally spaced intervals are allocated to the values from average to the maximum. Note that the intervals are in general different from the upper half (i.e., above average) to the lower half (i.e., below average).

2.1.2. AD

The mD2 computed and used only the distance among a pair of points. In our *Angle and Distance (AD)* shape feature, we measure both distance between a pair of points and angle formed by the surfaces the pair of points are located (Figure 3.) Then, the AD is a 2D histogram using the angle and distance as the two independent variables.

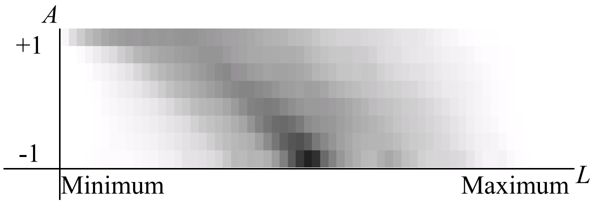
As in the case of the mD2 function, computation of the AD shape feature starts by generating N_p points on triangles of the model using the equation (1) and the Sobol's QRNS. As with the mD2, for every pair of points, we compute Euclidian distance. In addition, we compute inner product of the surface normal vectors of the triangles on which the pair of points are located.



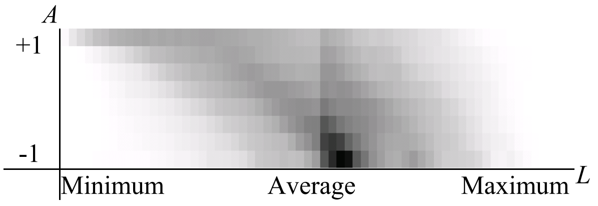
(a) D2 (b) AD and AAD

Figure 3. While the D2 (a) computes distance only of the pair of points, AD and AAD computes an inner product of surface normal vectors of the surfaces on which the points are located.

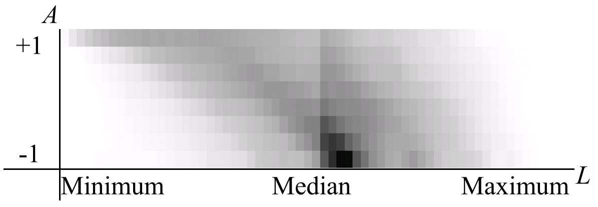
The 2D histogram of the AD shape feature must be normalized against the size of the model for proper comparison. We experimented with four different



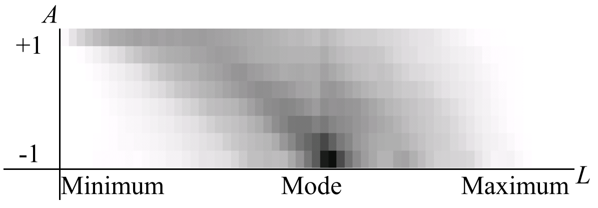
(a) The AD normalized using the maximum.



(b) The AD normalized using the average.



(c) The AD normalized using the median.



(d) The AD normalized using the mode.

Figure 4. The 2D histograms of the AD shape feature using various normalization methods.

normalization methods; (1) by maximum, (2) by average, (3) by median, and (4) by mode, of the distance values.

Normalization by Maximum: In this method, the distance value ranging from the minimum to the maximum is divided into I_d equally spaced intervals. The range of the inner product $[-1,1]$ is divided into I_a equally spaced intervals. The result is a 2D histogram having $I_a \times I_d$ elements. An example of the histogram generated from the bunny model of Figure 2 is shown in Figure 4a. In this example, 2000 points are generated, and the number of intervals are set to $I_d = 64$, and $I_a = 8$.

Normalization by Average: Normalization using average subdivides values below and above the average value into $I_d/2$ equally spaced intervals. The sizes of the interval may differ above and below the average. The range of the inner product $[-1,1]$ is divided into I_a equally spaced intervals. The result is a 2D histogram having $I_a \times I_d$ elements as above. An example of the histogram generated from the bunny model of Figure 2 using the normalization by average is shown in Figure 4b. In this example, 2000 points are generated, and the number of intervals are set to $I_d = 64$, and $I_a = 8$.

Normalization by Median or Mode: These two normalization methods are similar to the normalization methods using average above. Only difference is, instead of the average value, either median or mode of the distance values is used.

Examples of the histograms are shown in Figure 4c for the median and in Figure 4d for the mode. The histogram interval having higher population is shown in dark colors. The D2 function, a 1D histogram, can be obtained by collapsing the angular axis.

2.1.3. AAD

The AD shape feature described above is sensitive to the orientation (front/back) of the surface. If the models to be compared have properly oriented surfaces with a consistent representation (i.e., both using clockwise traversal of vertices), the AD shape feature performs quite well. If, however, the database contains models having surfaces that are inconsistently oriented, the performance of the AD shape feature suffers. In fact, for the models collected from the Internet, it is quite common for them to have either unoriented surfaces or surfaces oriented with different rules.

The *Absolute Angle and Distance (AAD)* shape feature takes absolute value of the inner product in order to increase robustness of the shape feature against models having such unoriented or inconsistently-oriented surfaces. Consequently, orientation axis of the 2D histogram of the AD takes the value in the range [0,1]. Other than that, AAD is computed in the identical manner as the AD.

Figure 5 shows an example of the AAD shape feature computed for the same bunny model of Figure 2 using the maximum value normalization, N_p=2000 points, $I_d = 64$, and $I_a = 8$.

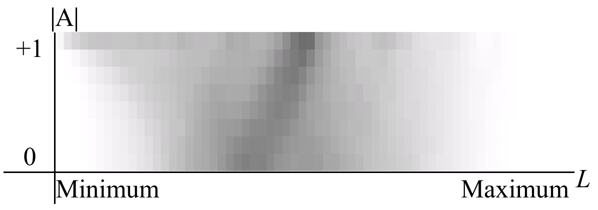


Figure 5. The AAD normalized by the maximum.

2.2. Dissimilarity computation

We have implemented three feature distance computation methods, L1 norm, L2 norm, and elastic matching.

2.2.1. L1 norm and L2 norm

The L1 norm (Manhattan distance) and the L2 norm (Euclidian distance) are two of the simplest distance computation methods. Let $X = (x_{i,j})$ and $Y = (y_{i,j})$ be the feature vectors for the model A and B, respectively. Assuming the number of intervals for the distance be I_d and the mutual angle (inner product) I_a, the L1 norm $D_{L1}(X,Y)$ and the L2 norm $D_{L2}(X,Y)$ are defined as follows;

$$D_{L1}(X,Y) = \sum_{i=1}^{I_d} \sum_{j=1}^{I_a} \left| (x_{i,j} - y_{i,j}) \right| \tag{3}$$

$$D_{L2}(X,Y) = \sum_{i=1}^{I_d} \sqrt{\sum_{j=1}^{I_a} (x_{i,j} - y_{i,j})^2} \tag{4}$$

2.2.2. Elastic matching distance

In the past, elastic matching had been used extensively in speech recognition. We performed elastic matching along the distance axis, using the dynamic programming technique for its implementation to compute the distance $D_E(\mathbf{X},\mathbf{Y})$. It locally stretches and shrinks the distance axis of the histogram in order to find minimal distance matches. If the matching is too elastic, however, a pair of shapes having very different histograms could have a low distance value. We implemented and experimentally compared the performance of the linear and the quadratic penalty functions, the latter of which is depicted in the equation (7). We used the better performing quadratic penalty function for the experiments described later in Section 4.

$$D_E(\mathbf{X},\mathbf{Y}) = g(\mathbf{X}_n,\mathbf{Y}_n) \tag{5}$$

$$g(\mathbf{X}_n,\mathbf{Y}_n) = \min\begin{bmatrix} g(\mathbf{X}_n,\mathbf{Y}_{n-1}) + \Delta g(\mathbf{X}_n,\mathbf{Y}_n) \\ g(\mathbf{X}_{n-1},\mathbf{Y}_{n-1}) + 2\Delta g(\mathbf{X}_n,\mathbf{Y}_n) \\ g(\mathbf{X}_{n-1},\mathbf{Y}_n) + \Delta g(\mathbf{X}_n,\mathbf{Y}_n) \end{bmatrix} \tag{6}$$

$$\Delta g(\mathbf{X}_i,\mathbf{Y}_j) = |i-j|\sqrt{\sum_{k=1}^{I_P}\left(x_{i,k} - y_{j,k}\right)^2} \tag{7}$$

3. Experiments and results

We implemented the experimental system using C++ on a Linux operating system.

3.1. Evaluation method

For the experiment, we used a database consisting of 215 VRML models provided by Patrick Min and Prof. Funkhouser at the Princeton University. The model database were categorized *a priori* into 42, one of which being "the other". Many of the categories contained only a few models.

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

First Tier (FT): Assume that the query belongs to the class C_q containing *k* 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.

The FT, ST, NN figures as well as the recall-precision plots shown later are the average produced by querying every one of the 215 models in the database once. Also note that performance figures depends on the model database; change in the model database or the categories used will result in different performance figures.

3.2. Parameter Selection for the Shape Features

The number of points generated on the surfaces N_p, the number of distance intervals I_d and angle intervals I_a affects the performance of the AD and AAD shape features. For example, if we increase the number of angular intervals to 16 for AD (or 8 for AAD), the feature became too sensitive to polygonal approximation; for example, a circle and its hexagonal approximation would have a relatively large distance. We varied these three parameters to find a parameter combination that are nearly optimal both in terms of performance and computational cost..

For the AD, we tested the total of 27 cases, in which N_p={512, 1024, 2048}, I_d={32, 64, 128}, I_a={4, 8, 16}. For the AAD, we tested total of 27 cases, in which N_p={512, 1024, 2048}, I_d={32, 64, 128}, I_a={2, 4, 8}. In both 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 FT, ST, NN, and the computation cost. The performances of two parameter combinations being equal, we chose the combination having lower computational cost. The combination of parameters we found to be the best are shown in Table 1.

Table 1. Selected parameters for the shape feature vectors.

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

3.3. Comparison among the proposed methods

We compared the performance of various variations of our proposed methods. We tested (1) two shape features AD and AAD, (2) four histogram normalization methods (by maximum, by average, by median, and by mode), and (3) three distance computation method (L1, L2, and elastic matching).

Table 2 shows the results of the experiment that compared retrieval performance and computation time of the shape features AD and AAD. We used the average normalization for histogram normalization and the L2 norm for feature 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. Figure 6 shows an example of queries

Table 2. Comparison among the proposed shape features.

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

Table 3. Performance comparison among the histogram normalization methods.

Normalization methods	FT	ST	NN	Feature comput'n 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 4. Performance comparison among the distance computation methods.

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

using the AD and the AAD shape features. The top-left box surrounded by a bold line is the query, and the other five are the retrieved results. The AAD produced more models with circular cross sections in the top five. The black surface of the model of a pot in Figure 6b is the result "flipped" surface normal.

Table 3 shows the results of comparison among the histogram normalization methods. For this experiment, the AAD shape feature and the L2 norm are used. This experiment shows that histogram normalization using average performed the best in all the performance figures. In terms of computational cost, maximum normalization was the least expensive, followed closely by the average normalization method.

Table 4 shows the comparison among the three distance computation methods. In this experiment, we used the AAD shape feature with the average normalization method. The figures in the table show that the L2 norm performed the best. The elastic matching improved the results for certain kind of models and queries. But overall, the simple L2 norm performed better than the elastic matching.

3.4. Comparison with the other methods

We compared the performance of the AD, AAD and the mD2 shape features. The parameters used for the experiment are as follows;

mD2: N_p=1024 points, I_d=512 intervals, L1 norm.

AD: N_p=1024, I_d=64, I_a=8, normalized using average, distance computed using L2 norm.

AAD: N_p=1024, I_d=64, I_a=4, normalized using average, distance computed using L2 norm.

The FT, ST, and NN figures and computational costs in time are shown in Table 5, and the recall-precision plot is shown in Figure 7.

Table 5 shows that the AD and AAD methods outperformed the mD2 in all of the FT, ST, and NN figures by the margin of 6% to 7%. The numbers are the average of all the models and categories in the database.

Table 5. Comparison among the mD2, AD, and AAD shape features.

Features	Performance			Computational cost	
	FT	ST	NN	Retrieval total	Feature computation
mD2	33%	44%	47%	0.70s	0.41s
AD	39%	51%	56%	0.84s	0.54s
AAD	38%	51%	60%	0.70s	0.54s

The recall-precision plots of Figure 7 shows that the AD and AAD clearly outperformed the mD2 method. Also, while it is difficult to discern, the AAD method slightly outperformed the AD method for this experiment.

Please note that, in a recall-precision plot, a curve closer to the upper right corner means a better retrieval performance. Also note that, if the experiment is ideal, the recall-precision plot converges to the two axes. In Figure 7, however, the minimum value for the recall is about 0.2 since the database contains categories having quite small cardinalities. For example, if a category contained only two members, the minimum recall value for the category is 0.5. Many of the 42 categories created using 215 models have only a few models as their members.

In terms of computational cost for extracting features, the AD and the AAD have somewhat higher cost than the D2. However, the cost differences among the three shape features are only modest. The added costs for the AD and AAD are well justified considering their performance advantage. In terms of distance computation cost, the AAD has the lowest cost due to its small feature vector size.

Figure 8 and Figure 9 show example queries using D2 (Figure 8a and Figure 9a) and the AD (Figure 8b and Figure 9b). Each figure lists the top 8 matches including the query. The query model is always the top match. Note that examples in Figure 8 and Figure 8 are obtained by using an expanded database containing 1213 models. The expanded database is created by combining the models from Patrick Min at the Princeton University, models from Johan Tangelder et al at the University of Utrecht [Tangelder03b], and our own set of models (about 300). Note also that the black faces (e.g., the model of vase in Figure 9a, rank 8) are due to "back-facing" polygons having flipped normal vectors.

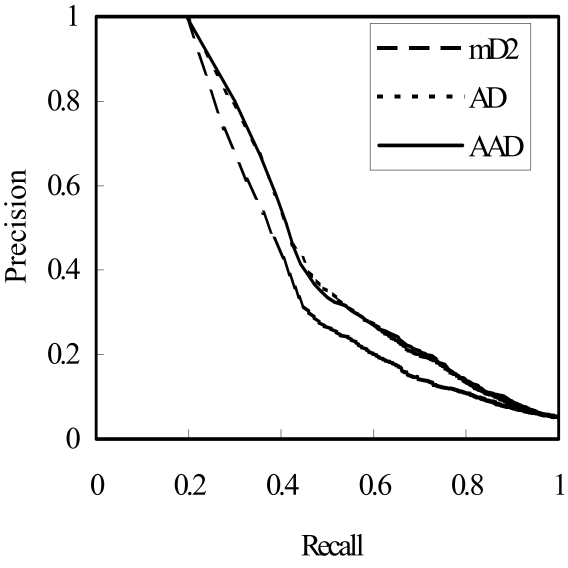


Figure 7. The precision-recall plot of the mD2, AD and AAD shape features.

In these examples, the AAD seem to perform better than the mD2. In Figure 8, for example, the AAD seems to retrieve more of the chair-like models than the mD2. In the example of Figure 9, compared to the mD2, the AAD retrieved models that appear to have "smooth and continuous" surfaces configurations.

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 AD and AAD, are robust against topological and geometrical irregularities and degeneracies, which make them applicable to VRML and other so called "polygon soup" models. While AD and AAD have computational cost somewhat higher (about 1.5 times) than the D2, they significantly outperformed D2 in our evaluation experiment. While the AD and AAD might have the performance lower than the more elaborate shape features, such as the spherical harmonics based feature used in [Funkhouser03]. However, the computational costs of AD and AD2 estimated to be significantly lower than that of the sophisticated shape feature used in [Funkhouser03].

As the future work, we'd like to improve our performance evaluation method, for example by developing more extensive database and categories. We also would like to improve our shape feature, for example by adding some form of multi-resolution approach to matching 3D shapes.

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

[Corney02] 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.

[Elad00] M. Elad, A. Tal, S. Ar. Directed Search in A 3D Objects Database Using SVM, *HP Laboratories Israel Technical Report*, HPL-2000-20 (R.1), August, 2000.

[Funkhouser03] 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).

[Hilaga01] 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.

[Ibato02] Masatoshi Ibato, Tomo Otagiri, Ryutarou Ohbuchi, Shape-similarity Search of Three-Dimensional Models Based on Subjective Measures, *IPSJ SIG report on Graphics & CAD*, 2002-CG-106, pp. 25-30, February, 2002.

[Keim99] 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.

[McWherter01] D. McWherter, M. Peabody, W. Regli, A. Shokoufandeh, Transformation Invariant Shape Similarity Comparison of Solid Models, Proc. *ASME DETC '2001*, September 2002, Pittsburgh, Pennsylvania.

[Mukai02] 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.

[Novotni01] 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.

[Osada01] 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.

[Osada02] R. Osada, T. Funkhouser, Bernard Chazelle, and David Dobkin Shape Distributions, *ACM TOGS*, **21**(4), pp. 807-832, (October 2002).

[Paquet97] 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.

[Paquet00] 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**, pp. 103-122, 2000.

[Press92] W. H. Press et al., *Numerical Recipes in C-The Art of Scientific Programming*, 2nd Ed., Cambridge University Press, Cambridge, UK, 1992.

[Regli00] W. Regli, V. Cicirello, Managing Digital Libraries for Computer-Aided Design, *Computer Aided Design*, pp. 110-132, **32**(2), 2000.

[Suzuki98] 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.

[Suzuki00] 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.

[Tangelder03a] J. W. H. Tangelder, R. C. Veltkamp, Polyhedral Model Retrieval Using Weighted Point Sets, *Int'l Journal of Image and Graphics*, **3**(1), pp. 209-229 (2003).

[Tangelder03b] http://www.cs.uu.nl/centers/give/imaging/

3Drecog/3Dmatching.html

[Veltkamp01] R. C. Veltkamp, Shape Matching: Similarity Measures and Algorithms, invited talk, Proc. *Int'l Conf. on Shape Modeling and Applications 2001*, pp. 188-197, Genova, Italy, May, 2001.

[Vranić01] D. V. Vranić, D. Saupe, and J. Richter, Tools for 3D-object retrieval: Karhunen-Loeve Transform and spherical harmonics. Proc. of the *IEEE 2001 Workshop on Multimedia Signal Processing*, Cannes, France, pp. 293-298, October 2001.

[Zaharia01] 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.

[Zaharia02] T. Zaharia, F. Préteux, Shape-based retrieval of 3D mesh models, Proc. *IEEE ICME 2002*, Lausanne, Switzerland, August, 2002.

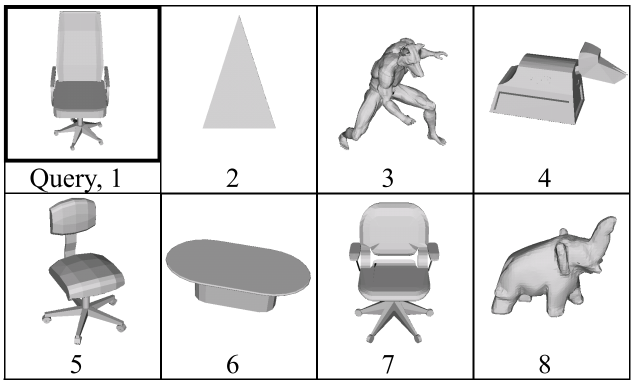


Figure 8a. Query result by using the mD2 shape feature.

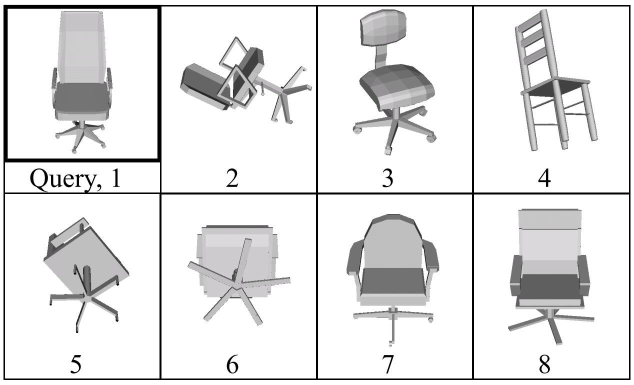


Figure 8b. Query result by using the AAD shape feature.

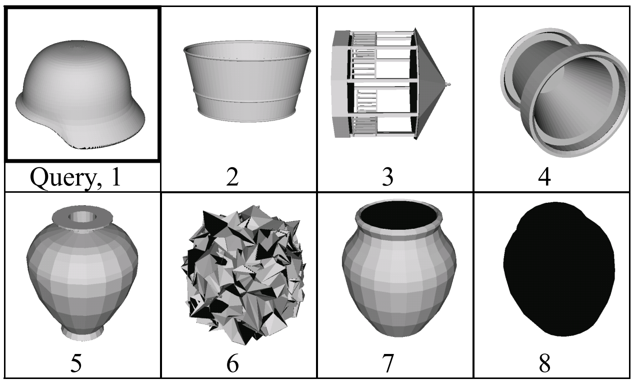


Figure 9a. Query result by using the mD2 shape feature.

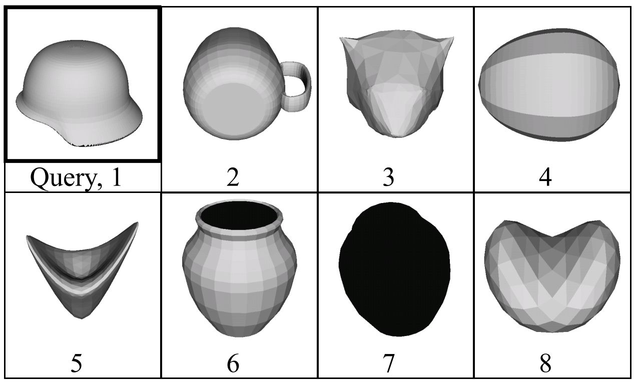


Figure 9b. Query result by using the AAD shape feature.