Polyhedral Model Retrieval Using Weighted Point Sets

Johan W.H. Tangelder, Remco C. Veltkamp
Institute of Information and Computing Sciences, Utrecht University
P.O. Box 80.089, 3508 TB Utrecht, The Netherlands
hanst@cs.uu.nl, Remco.Veltkamp@cs.uu.nl

Abstract

Due to the recent improvements in laser scanning technology, 3D visualization and modelling, there is an increasing need for tools supporting the automatic search for 3D objects in archives. In this paper we describe a new geometric approach to 3D shape comparison and retrieval for arbitrary objects described by 3D polyhedral models that may contain gaps. In contrast with existing approaches, our approach takes the overall relative spatial location into account by representing the 3D shape as a weighted point set. To compare two objects geometrically we first apply principal components analysis to bring the objects in a standard pose, and enclose each object by a 3D grid. Then we generate for each object a signature representing a weighted point set, that contains for each non-empty grid cell a salient point. We compare three methods to select in each grid cell a salient point and a weight: (1) choose the vertex in the cell with the highest Gaussian curvature, and choose as weight a measure for that curvature, (2) choose the area-weighted mean of the vertices in the cell, and choose as weight a measure denoting the normal variation of the facets in the cell and (3) choose the centre of mass of all vertices in the cell, and choose as weight one. Finally, we compute the similarity between two shapes by comparing their signatures using a new shape similarity measure based on weight transportation that is a variation on the Earth Mover's Distance. Unlike the Earth Mover's Distance, the new shape similarity measure satisfies the triangle inequality. This property makes it suitable for use in indexing schemes, which frequently depend on the triangle inequality, such as the one we introduce, based on so-called vantage objects. The strength of our approach is proven through experimental results using a database consisting of 133 models such as mugs, cars and boats, and a database consisting of 512 models, mostly air planes, classified into conventional air planes, delta-jets, multi-fuselages, biplanes, helicopters and other models.

1 Introduction

As a result of the recent improvements in laser scanning technology, the acquisition of 3D models by 3D digitizing now is commonplace. Applications of emerging relevance are augmented reality using digitized 3D models, 3D shape retrieval, and the creation of digital archives for all kind of purposes, e.g. recording cultural heritage and reverse engineering. The World Wide Web enables access to these digital archives and desktop computers now have the power to process and display huge 3D data sets. Hence, there is an increasing need for tools supporting the automatic search for 3D objects in archives.

The World Wide Web provides access to thousands of 3D objects mostly in virtual reality modelling language (VRML) format. Most 3D file formats like VRML represent 3D models as meshes. To represent a 3D shape properly, the mesh has to be closed, such that the mesh is a polyhedron. Since the main application of the VRML models is visualization, in practice VRML models are often not watertight, i.e. the polyhedra may contain small gaps. Also, they may contain wrongly-oriented polygons.

In this paper, we describe a new geometric approach to 3D shape comparison and retrieval for arbitrary objects described by 3D polyhedral models that may contain gaps. The key idea is to represent the signature of an object as a weighted point set that represents the salient points of the object and to compare these weighted point sets using a new shape similarity measure based on weight transportation that is a variation on the Earth Mover's Distance. Unlike the Earth Mover's Distance, the new shape similarity measure satisfies the triangle inequality. This property makes it suitable for use in indexing schemes, which frequently depend on the triangle inequality.

Also, we have implemented a 3D shape retrieval engine [35] that demonstrates the capabilities of this new approach. The strength of our approach is proven through experimental results using a database consisting of 133 models such as mugs, cars and boats, and a database downloaded from the World Wide Web consisting of 512 models, mostly air planes classified into conventional air planes, delta-jets, multi-fuselages, biplanes, helicopters and other models.

The outline of the paper is as follows. The next section contains a summary of related work. Section 3 describes how we extract the weighted point sets and how we compute the distance between two such sets. Experimental results are presented in Section 4 and discussed in Section 5. Finally, we present our conclusions and indicate future research topics in Section 6.

2 Related Work

An important issue in the context of 3D archives is how to search for 3D objects in a similar way as we already search for text, images, audio and video. Up to now there are only few references to the specific problem of content based retrieval of 3D models. However, an extensive amount of literature can be found in the related fields of computer vision, object recognition and geometric modelling. For a broad introduction to this literature, please consult the survey paper by Campbell and Flynn [7]. The vast majority of work in shape matching has focused on characterizing similarity between objects in 2D images. For an overview of 2D shape matching methods we refer the reader to the paper by Veltkamp [31]. Unfortunately, most 2D methods do not extend directly to 3D model matching. In particular, extending methods of comparing boundaries in two dimensions to higher dimensions is non-trivial, both in theory and in practice. In the following, we describe 3D shape matching research based on comparing shapes in 3D directly.

Feature based similarity: Cicirello and Regli [8] present an approach to compare the similarity of solid models of machined artifacts based on the similarity of their machining features and to interrogate databases of these models. Since machining features contain manufacturing process knowledge their research is especially relevant for the CAD/CAM community, but unfortunately it is not applicable for models of natural shapes like humans and animals.

2D view based similarity: A number of approaches compare 3D models by the similarity of their 2D views. Löffler [19] describes a content-based retrieval method that matches a user provided 2D sketch with views from the 3D model in the database. Cyr and Kimia [12] present a method to obtain representative views using a shape similarity based aspect graph that clusters views into aspects. Funkhouser et al. [15, 34] implemented an experimental 3D search engine supporting retrieval by shape using a sketch interface providing one, two or three 2D outlines of the shape to be retrieved.

Histogram based similarity: The following approaches to 3D shape matching compare histograms or distributions encoding shape properties. Shum et al. [29] use a spherical coordinate system to map the surface curvature of 3D objects to the unit sphere. By searching over a spherical rotation space a distance between two curvature distributions is computed and used as a measure for the similarity of two objects. Unfortunately, the method is limited to objects which contain no holes, i.e. have genus zero. Ankerst et al. [1] use shape histograms defined on shells and sectors around a model's centroid and compare shapes using a quadratic form distance measure to compare the histograms. Elad et al. [14] use a moments-based classifier and a weighted Euclidean distance measure. Their method supports iterative and interactive database searching in which the user can improve the weights of the distance measure by marking relevant search results. Paquet et al. [23] apply cords-based, moments-based and wavelets-based descriptors for 3D shape matching. A descriptor based on the 3D Discrete Fourier Transform is introduced by Vranić and Saupe [33]. Kazhdan [18] describes a reflective symmetry descriptor associating a measure of reflective symmetry to every plane through the model's centroid. Osada et al. [21, 22] introduce and compare shape distributions measuring properties based on distance, angle, area and volume measurements between random surface points. They evaluate the similarity between the objects using a metric that measures distances between distributions. In their experiments the shape distribution measuring distances between random surface points is most effective.

Topology based similarity: Hilaga et. al. [17] describe a topological matching method relevant especially for articulated objects. Their method uses Reeb graphs based on geodesic distances to encode the topology of 3D objects.

Volume based similarity: Novotni and Klein [20] describe a geometric similarity approach to 3D shape matching based on calculating a volumetric error between one object and a sequence of offset hulls of the other object. A drawback of their method is that their similarity measure is no metric, because it is not symmetric and does not obey the triangle inequality.

Deformation based similarity: A number of methods compare a pair of 2D shapes by measuring the amount of deformation required to register the shapes exactly. For example, Cohen et al. [10] use a representation based on curvature in order to encourage matching curvature extrema between counters. Also, Basri et al. [3] propose a method to measure the degree of similarity between two image contours taking into account deformations in object shape. In contrast with [10], they use a similarity function that is a metric. The 2D methods described above depend on the natural arc length parameterization of their contours. Another problem is that the dimensionality of 3D data is higher, which makes registration, finding feature correspondences, and fitting model parameters more expensive. As a result, methods that apply deformation for shape recovery [30] or shape evolution [13] are very difficult to apply for 3D shape matching.

Many of the methods mentioned above do not take the overall relative spatial location into account, but throw away some of this information, in order to deal with data of lower

complexity, e.g. 2D views, or 1D histograms. What is new in our method, is that we use the overall relative spatial position by representing the 3D shape as a weighted point set, without taking the connectivity relations into account however. The weighted point sets are compared using a new transportation distance that is a variant of the Earth Mover's Distance [28]. New in our approach is that in contrast with the Earth Mover's Distance this transportation distance satisfies the triangle inequality. Since this transportation distance obeys the triangle inequality, our method can be used in indexing schemes that employ this property, of which we will introduce one.

3 Overview of approach

To compare two objects independently of orientation, position and scaling we first apply principal components analysis to bring the objects in a standard pose. Also in the preprocessing step, we enclose each object by a 3D grid and generate for each object a signature representing a weighted point set, that contains for each non-empty grid cell a salient point. We compare three methods to select in each grid cell a salient point and a weight: (1) choose the vertex in the cell with the highest Gaussian curvature, and choose as weight a measure for that curvature, (2) choose the area-weighted mean of the vertices in the cell, and choose as weight a measure denoting the normal variation of the facets in the cell and (3) choose the centre of mass of all vertices in the cell, and choose as weight one. Finally, we compute the similarity between two shapes by comparing their signatures using a shape similarity measure that is a new variation on the Earth Mover's Distance.

We assume that a 3D shape is represented by a polyhedral mesh. We do not require that the polyhedral mesh is closed. Therefore, our method can also handle polyhedral models that may contain gaps.

3.1 Preprocessing

3D models have arbitrary scale, orientation and position in the 3D space. In order to compute a correct measure of similarity it is necessary to place the 3D models into a canonical coordinate system. This placing into the canonical coordinate system should be the same if we translate, rotate or scale the model. Furthermore, if a model is given in multiple levels-of-detail, canonical representations of different levels should be approximately the same. First, the centre of mass of the surfaces of each polyhedral model is translated to the origin. Note that we cannot translate the centre of mass of the solid enclosed by the model to the origin, because for a polyhedral model containing one or more gaps this solid is not defined. We use the Principal Component Analysis (PCA) method to compute for each polyhedral model the principal axes e_1, e_2 and e_3 and their eigenvalues λ_1, λ_2 and λ_3, and make the necessary conditions to get right-handed coordinate systems. These principal axes define an orthogonal coordinate system (e_1, e_2, e_3), with $\lambda_1 \geq \lambda_2 \geq \lambda_3$. Next, the polyhedral model is rotated around the origin such that the coordinate system (e_1, e_2, e_3) coincides with the coordinate system (e_x, e_y, e_z).

Conventionally, the PCA [24] is applied only to a set of points (e.g., vertices or centroids of facets), thus, the differing sizes of facets cannot be taken into account. Therefore, we applied a "weighted" PCA [33] that relates the area of facets to weights associated to the vertices of the polyhedral model. The PCA algorithm for pose estimation is fairly simple and efficient. However, as noted by Novotni and Klein [20] the application of this procedure

without further processing does not result always in a correct estimation. This is due to the following problems:

- The PCA method gives the three principal axis, but it lacks any information about their direction resulting thus in a two way ambiguity for each axis. This means that we have a total of 4 configurations corresponding to the four possible right-handed coordinate systems that all represent the same principal axes. Hence, if we want to obtain a similarity value $d(A, B)$ comparing two objects A and B, we consider four rotated copies B_1, B_2, B_3 and B_4 of B and compute $d(A, B)$ as the minimum of $d(A, B_1)$, $d(A, B_2)$, $d(A, B_3)$, and $d(A, B_4)$.
- If the eigenvalues are similar, principal axis may switch, without affecting the eigenvalues. Since solving this problem is difficult, we accept that in a number of cases we may obtain bad results.

After the PCA, as a last preprocessing step we divide the unit cube that encloses the object into a grid consisting of 25*25*25 cells of equal size.

3.2 Extracting salient points

Next, we generate for each object a signature S representing a weighted point set, that contains for each non-empty grid cell a salient point. Below we compare three methods to select in each grid cell a salient point. All three methods use only the vertices and the facets adjacent to the vertices to select a salient point. Therefore, they can handle models that contain gaps. Models containing polygons that are wrongly oriented, are only handled correctly by the third method.

3.2.1 Gaussian curvature method

For a smooth surface the Gaussian curvature c at a point is the product of the minimal and maximal principal curvature at that point.

For polyhedral meshes the Gaussian curvature $c(v)$ at a vertex v can be computed by the following rule from Calladine [6]:

$$c(v) = \frac{d(v)}{a(v)}.$$

Here, $d(v)$ denotes the angular defect at v, which is defined for interior vertices as 2π minus the sum of the interior angles of the facets meeting at v. For vertices v at the boundary of a gap the angular defect is defined as π minus the sum of the interior angles of the facets meeting at v. The scalar $a(v)$ denotes the area associated with vertex v, where each facet contributes to $a(v)$ the area of the facet divided by the number of its vertices.

The flatter the surface, the smaller c will be. If all the facets are coplanar, for interior vertices c will be zero. If for a boundary vertex the edges of the boundary do not meet at an angle, $d(v)$ is also zero.

The Gaussian curvature method computes for each non-empty grid cell the vertex v with the highest absolute value of the Gaussian curvature. We take the absolute value of the Gaussian curvature, because our similarity measure cannot handle point sets with negative weights. Hence, at an individual point we cannot distinguish between elliptic and hyperbolic

shapes. But, because in general the weighted point sets of elliptic and hyperbolic shapes will differ, these shapes will be found dissimilar using our similarity measure. Since the absolute value $|c|$ of the Gaussian curvature c may have values between zero and infinity, we normalize to the range $[0, 1]$, and define a measure M, such that for all $M(x) = 1 - 1/(1 + x)$. This normalization avoids that vertices with very high curvatures disturb the computation of our similarity measure and makes it less sensitive to noise. For each non-empty grid cell the weighted point $(v, M(|c|))$ is added to the signature S.

3.2.2 Normal variation method

Another approach to obtain a measure related to the curvature is the normal variation method. In this approach we estimate the curvature in a grid cell by the normal variation in the grid cell. We choose the area-weighted mean of the vertices $p(c)$ in the grid cell as a salient point and we choose as weight a measure for the normal variation.

We compute the area-weighted mean $p(c)$ of the vertices in the grid cell by

$$p(c) = \left(\sum_{j=1}^{M} w_j v_j \right) / \sum_{j=1}^{M} w_j,$$

where M is the number of vertices in the cell, v_j is the j^{th} vertex in the cell. The weight w_j denotes the area associated with vertex v_j, where each facet contributes to w_j the area of the facet divided by the number of its vertices.

To compute the normal variation we use the following from Brodsky and Watson [5]. For each grid cell we compute the mean normal $\vec{mn}$ that is the area-weighted mean of all the normals of facets adjacent to a vertex in the grid cell as given by the equation

$$\vec{mn} = \sum_{i=1}^{N} a_i \vec{n_i},$$

where N is the number of facets in the cell, $\vec{n_i}$ is the normal of facet i, and a_i is the area of facet i.

The flatter the surface, the larger the magnitude of $\vec{mn}$ will be. If all the facets are coplanar, the magnitude of $\vec{mn}$ will equal the area of surface of the surface in the cell. Hence, $cp = \|\vec{mn}\| / \sum_{i=1}^{N} a_i$ equals one. Otherwise cp will be smaller than one. Therefore, we choose $1 - cp$ as weight and add for each non-empty grid cell the weighted point $(p(c), 1 - cp)$ to the signature S.

In the normal variation method gaps will result in a number of missing facets, that are not taken into account in the procedure described above. If the area of these facets is small, then the error caused by the gaps will also be small.

3.2.3 Midpoint method

The two methods described above may fail if 3D models contain wrongly oriented polygons. This is the case for models that are represented by "polygonal soups", i.e. unorganized and degenerate sets of polygons. To handle such degenerate models, we also implemented a simple approach called midpoint method, that is similar to Rosignac's polygon simplification algorithm [27]. The midpoint method obtains a signature S by adding for each grid cell the centre of mass of all vertices in the cell with unit weight to the signature S.

3.3 Matching

Now that we have the signatures of our polyhedral models represented as sets of weighted points, we need a way to match two such sets. So we need a definition of a distance function, and an algorithm to compute the distance.

A distance measure is a function defined on pairs of patterns indicating the degree of their resemblance. Formally speaking, a distance measure d on a set S is a nonnegative valued function $d : S \times S \to \mathrm{R}^+ \cup \{0\}$. For many pattern matching applications, it is desirable that d has some of the following properties:

i. *Self-identity*: For all $x \in S, d(x, x) = 0$.

ii. *Positivity*: For all $x \neq y$ in $S, d(x, y) > 0$.

iii. *Symmetry*: For all $x, y \in S, d(x, y) = d(y, x)$.

iv. *Triangle inequality*:
For all $x, y, z \in S$, $d(x, z) \leq d(x, y) + d(y, z)$.

v. *Transformation invariant*: For a chosen transformation group G, for all $x, y \in S$, $g \in G$, $d(g(x), g(y)) = d(x, y)$. This also implies $d(g(A), B) = d(A, g^{-1}(B))$.

A function d having properties (i)–(iv) is called a metric. Other combinations are possible: a pseudo-metric is a function that has properties (i), (iii) and (iv), while a semi-metric is a function that obeys only (i), (ii) and (iii).

The triangle inequality is very useful for making searching more efficient [2]. This is based on the following observation. Consider a shape or weighted point set A_1 that closely matches a query A_q: $d(A_1, A_q)$ is small. Let A_r be some reference shape. If the triangle inequality holds, $d(A_r, A_q) \leq d(A_r, A_1) + d(A_1, A_q)$, then we know that $d(A_r, A_q) - d(A_r, A_1)$ is small as well. We can approximate the distance between a database shape A_1 and a query A_q by comparing their distances to a reference shape A_r. This observation can be applied to implement efficient indexing and searching of the shape database using the vantage method, as follows [32]. Calculate off-line the distance between all database objects and a reference shape called vantage object. The set of objects that have about the same distance to the vantage as a query object, contains also those objects that have about the same distance to the query object (if the triangle inequality holds). This can be extended to more vantage objects. In this way, on-line complex shape comparisons have to be done with only a few vantage objects, at the cost of false positives, but no false negatives. After computing the distances of all database objects to a fixed number of vantage objects, for querying only a few expensive shape comparisons are needed, the actual range query is done efficiently in higher dimensional Euclidean space.

There are two main approaches to compare weighted point sets. One approach is to interpret the point sets as fuzzy sets. However, a distance measure for fuzzy sets that is a metric, invariant under rigid motion and respects scaling of the underlying ground distance, does not exist [4]. In addition, a Hausdorff-like pseudo-metric fails to differentiate between fuzzy sets with arbitrarily different maximum membership values. The other approach is the Earth Mover's Distance (EMD). However, for sets of unequal total weights, it gives zero distance for arbitrarily different sets, and it does not obey the triangle inequality. Therefore, we describe below a new shape similarity measure based on weight transportation that is a

variation on the EMD and satisfies the triangle inequality. We refer the reader to [16] for an exhaustive description of the EMD and the new shape similarity measure.

3.3.1 Earth Mover's Distance

The distance that we will use for matching is a variation on the EMD [28]. The EMD between two weighted point sets measures the minimum amount of work needed to transport from a supplier set of weights to a demander set of weights. Stated in a different way, the EMD is the average ground distance that weights travels during an optimal flow [11]. Let $\mathcal{N}$ denote the space of weighted point sets, in which any two sets can have unequal total weights. In [16] it is demonstrated that the EMD has the following drawbacks when applied on $\mathcal{N}$:

1. It does not obey the positivity property. The EMD does not take into account the surplus of weight, if any, between two sets. As a result, there are cases where it does not distinguish between two non-identical sets. Even for arbitrarily different sets, the distance can be zero.

2. It does not obey the triangle inequality. As a result, the EMD prevents the triangle inequality from being used in speeding up database retrieval.

Consequently, the EMD on $\mathcal{N}$ is not a metric.

3.3.2 Proportional Transportation Distance

An interesting question, that naturally arises, is the following: is there a similarity measure based on weight transportation such that the surplus of weight between two point sets is taken into account and the triangle inequality still holds? In the sequel we present a new distance for weighted point sets in $\mathcal{N}$. Let $A, B \in \mathcal{N}$. When measuring the distance from A to B, rather than taking A as the supplier and B as the demander moving only as much weight as needed, trying to fill the 'holes' with 'earth', we move the total weight of A to the positions of the points in B. What we measure then, is the minimum amount of work needed to transform A to a new set A' that resembles B. In particular, we redistribute A's total weight from the position of its points, to the position of B's points leaving the old percentages of weights in B the same.

We call this distance the Proportional Transportation Distance (PTD); it is defined as follows. Let $A = \{a_1, a_2, \ldots, a_m\}$ be a weighted point set such that $a_i = (x_i, w_i)$, $i = 1, \ldots, m$ where $x_i \in I\!R^k$ with $w_i \in I\!R \cup \{0\}$ being its corresponding weight. Let also $W = \sum_{i=1}^{m} w_i$ be the total weight of set A. Let $B = \{b_1, b_2, \ldots, b_n\}$ be a weighted point set such that $b_i = (y_i, u_i)$, $i = 1, \ldots, n$ where $y_i \in I\!R^k$ with $u_i \in I\!R \cup \{0\}$ being its corresponding weight. Let also $U = \sum_{i=1}^{n} u_i$ be the total weight of set B. Let d be a ground distance between two single points, typically the Euclidean distance. Formally, the PTD can be expressed as a linear programming problem. We denote as f_{ij} the elementary flow of weight from x_i to y_j, over the elementary distance d_{ij}. The set of all feasible flows $\mathcal{F} = [f_{ij}]$ from A to B, is now defined by the following constraints:

1. $f_{ij} \geq 0, i = 1, ..., m, j = 1, ..., n$

2. $\sum_{j=1}^{n} f_{ij} = w_i, i = 1, ..., m$

3. $\sum_{i=1}^{m} f_{ij} = \frac{u_j W}{U}, j = 1, ..., n$

4. $\sum_{i=1}^{m} \sum_{j=1}^{n} f_{ij} = W$

The PTD(A, B) is given by the following objective function:

$$\text{PTD}(A, B) = \frac{\min_{F \in \mathcal{F}} \sum_{i=1}^{m} \sum_{j=1}^{n} f_{ij} d_{ij}}{W} \tag{1}$$

Constraints 2 and 4 forces all of A's weight to move to the positions of points in B. Constraint 3 ensures that this is done in a way that preserves the old percentages of weight in B. Next we examine PTD's properties.

3.3.3 Properties of the PTD

Let us take a closer look at PTD's definition. While measuring the PTD(A, B) for any sets A and B, if we substitute the variables f_{ij} with $f'_{ij}W$, $i = 1..m$, $j = 1..n$ in its linear programming (LP) formulation, call it LP_1 we get the following LP problem:

$$\min_{F \in \mathcal{F}} \sum_{i=1}^{m} \sum_{j=1}^{n} f'_{ij} d_{ij}$$

where $\mathcal{F}$ is defined by:

1. $f'_{ij} \geq 0$
2. $\sum_{j=1}^{n} f'_{ij} = w_i / W$
3. $\sum_{i=1}^{m} f'_{ij} = u_j / U$
4. $\sum_{i=1}^{m} \sum_{j=1}^{n} f'_{ij} = 1$

It is clear that this new formulation, call it LP_2, gives us the distance between the two sets of percentages of weights in A, B. Note that the total weights of the new sets are both equal to one. Since the substitution function $f_{ij} = f'_{ij}W, W \neq 0$ is bijective, LP_1 is equivalent to LP_2. This means that we are working on the space of equal total weight sets.

However, it's obvious that more than one LP_1 problems can be equivalent to the same LP_2 problem i.e. any two weighted point sets of the same cardinality and positionally coincident, can have the same percentages of weight at the same positions although their corresponding individual weights are different.

We can now state the properties of PTD.

1. It obviously has the *identity* property.
2. It obeys the *triangle inequality*.
3. It does not follow the *positivity* property since the distance between positionally coinciding sets with the same percentages of weights at the same positions is 0. However this is the only case in which the distance between two non-identical point sets is zero. The PTD will distinguish two sets B and B' where the one came from the other by adding even only one point.

It follows that the PTD is a pseudo-metric. Of course, by identifying sets with zero distance we can produce a metric on the resulting partition of the set $\mathcal{N}$ of generally unequal total weight sets.

The PTD can be computed efficiently by solving the corresponding linear programming problem, using for example a streamlined version of the simplex algorithm for the transportation problem. In practice simplex performs well, but in theory it can perform an exponential number of steps before giving a solution. Theoretically better (polynomial time) algorithms for general linear programming, like an interior point algorithm could be used; however it is likely to perform better than the simplex method only for very large problem sizes. Since the transportation problem is a special case of the minimum cost flow problem in networks, a polynomial time algorithm that solves the latter can be used as well.

4 Experimental results

We have tested our geometric approach to 3D shape comparison on two different databases, one consisting of 133 models such as mugs, cars and boats, and one consisting of 512 models including 376 models of air planes. Our primary objective is to show that our simple approach can actually be used in 3D shape comparison. This together with its properties, namely the triangle inequality, would make it a good candidate for shape retrieval applications as explained in the previous section.

In order to calculate the PTD between two weighted point sets, each weighted point set is first normalized by dividing each individual point weight by the total weight of the points in the set, as the LP_2 formulation suggests. The distance computation is based on the EMD publicly available code [9].

The computation of the PTD between two sets is possible in a reasonable amount of time. In our experiments in the average case computing the PTD takes about 5 seconds on a typical Pentium III, 1000 MHz with 256MB of memory. In our experiments the worst case is computing the PTD between two sets of around 300 points each, which takes about 40 seconds.

A linear search through a database of 512 models would take, on the average, over 42 minutes, which would make this similarity measure unsuitable for retrieval purposes. Employing the triangle inequality with the vantage method however, using for example 8 vantage objects, a query would take on the average 40 seconds.

4.1 Robustness to level of detail

We test the robustness of our similarity measure to level of detail by testing it with different polyhedral approximations of two 3D shapes using 4 approximative models of the Utah teapot and 4 approximative models of an ellipsoid. The polyhedral approximations of the Utah teapot have been obtained by exporting the B-spline representation to VRML using different levels of detail using the Rhinoceros software [26]. The polyhedral approximations of the ellipsoid have been obtained by first approximating the sphere for $k = 2, 3, 4, 5$ with k-frequency icosahedra [25] containing $20k^2$ triangular facets. To approximate an ellipsoid the x, y and z coordinates of the k-frequency icosahedra have been scaled with a factor 4, 2, and 1 respectively.

From table 1 we see that there is some difference in the similarity measure for the first level of detail and the other levels of detail. For increasing numbers of vertices the similarity is higher, as denoted by the smaller distance value. This can be explained as follows. For

	Gauss	**Norm**	**Midp**
ptd(e1,t1)	0.145	0.183	0.143
ptd(e2,t2)	0.137	0.163	0.122
ptd(e3,t3)	0.121	0.162	0.115
ptd(e4,t4)	0.115	0.148	0.103
ptd(e1,e2)	0.070	0.085	0.072
ptd(e2,e3)	0.057	0.074	0.055
ptd(e3,e4)	0.043	0.067	0.043
ptd(t1,t2)	0.131	0.153	0.130
ptd(t2,t3)	0.043	0.040	0.041
ptd(t3,t4)	0.031	0.025	0.021

Table 1: Overview of robustness to level of detail results. e_1, e_2, e_3, and e_4 denote approximations of an ellipsoid containing 42, 92, 162 and 252 vertices, respectively. t_1, t_2, t_3, and t_4 denote approximations of the Utah teapot containing 262, 3743, 9795 and 17233 facets, respectively. *ptd* denotes the Proportional Transportation Distance for signatures generated with the Gaussian curvature (**Gauss**), the normal variation (**Norm**) and the midpoint method (**Midp**).

Method	Nearest Neighbour	First Tier	Second Tier
Gauss	82%	53%	70%
Norm	75%	50%	68%
Midp	78%	53%	71%
D2	66%	49%	66%

Table 2: Comparison of the Gaussian curvature (**Gauss**), the normal variation (**Norm**) and the midpoint method (**Midp**) to the $D2$ shape distribution based method [21, 22] using the Princeton database.

a low number of vertices in the model, also the number of points in the model's signature, which is equal to the number of grid cells containing at least a vertex, is low. Hence, the weighted point set does not represent the shape very well. For instance $ptd(t1, t2)$ is almost the same as $ptd(e1, t1)$. In our case the signatures of e_1, e_2, e_3, and e_4 contain 42, 78, 110 and 152 points respectively, and the signatures of t_1, t_2, t_3, and t_4 contain 47, 157, 208, and 225 points, respectively. We conclude that for the higher level of details (> 100 points) our signatures and similarity measure together are reasonably robust against change in level of detail.

4.2 Shape retrieval results

Also, we compare the ability of the Gaussian curvature, normal variation and midpoint shape matching method to find shapes similar to a query image. We tested our results using a database from Princeton, that has also been used by Osada et al. [21, 22], and a test database consisting of 512 models.

The database from Princeton consists of 133 models retrieved from the World Wide Web and grouped qualitatively (by function more than by shape) into 25 classes: 5 animals, 4

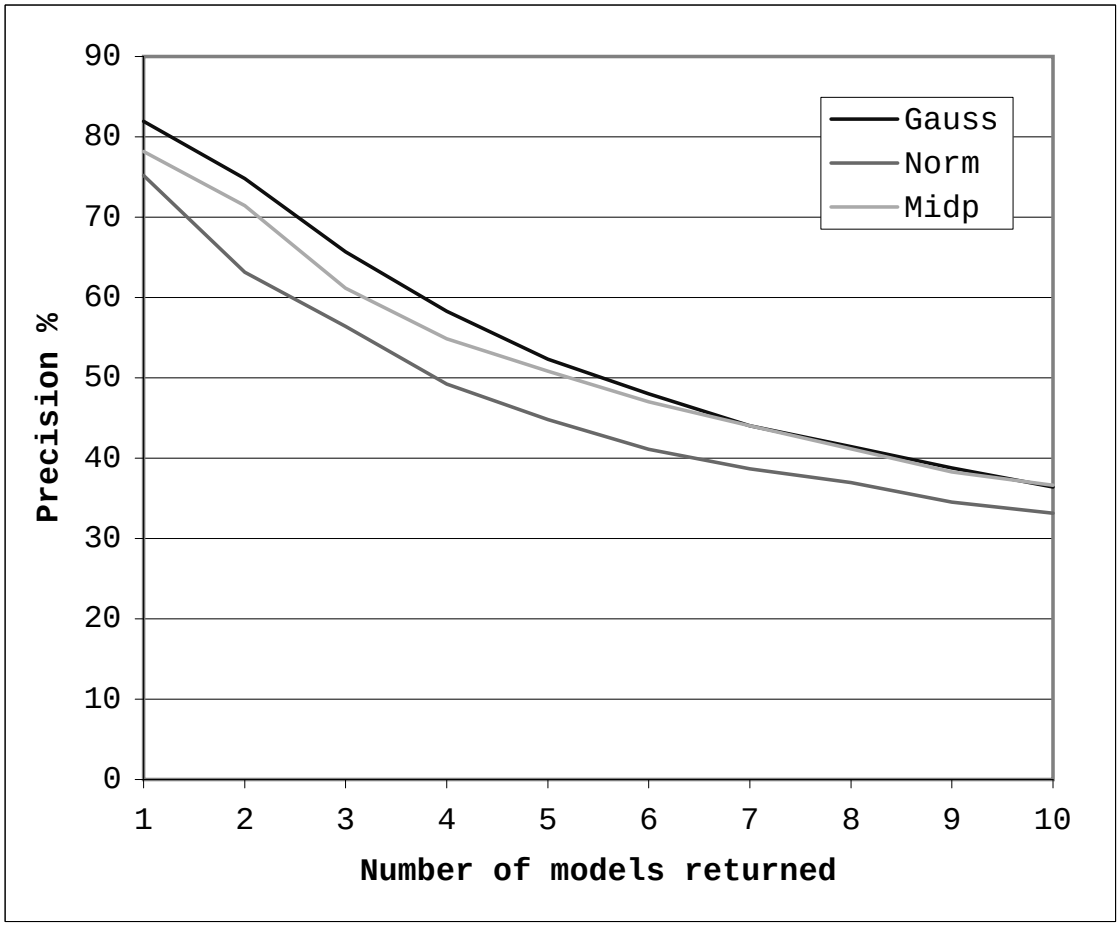

Figure 1: Precision versus number of models returned for the Princeton database.

balls, 2 belts, 3 blimps, 3 boats, 6 cars, 8 chairs, 3 claws, 4 helicopters, 11 humans, 3 lamps, 3 lightnings, 6 missiles, 4 mugs, 4 open books, 4 pens, 4 phones, 27 planes, 4 rifles, 3 skate boards, 4 sofas, 6 spaceships, 3 subs, 4 tables, and 5 tanks. Some classes (such as ball, mug, open book, pen and sub) contained 3D models with shapes greatly resembling each other, while others (such as animal, boat, car and plane) contained models with a wide variety of shapes.

To investigate the ability of our shape matching methods to discriminate between classes of objects, for each method, we computed for each object in the database the distances to all other objects in the database. In our tests we used each object as query object. Table 2 presents the results of these tests. We compare the results against the best of the shape distribution based methods described in [21, 22]. This is the so-called $D2$ method, that represents a shape by the distribution of Euclidean distances between pairs of randomly selected points on the surface of a 3D model. The first column indicates the shape matching method. The second column lists the percentage of retrieved objects in which the nearest neighbour was from the query's class. Let k denote the number of objects in the query's class. The third column ("First Tier") lists the percentage of retrieved objects from the query's class within the first $k-1$ hits excluding the query. The fourth column ("Second Tier") lists the percentage of retrieved objects from the query's class within the first $2(k-1)$ hits excluding

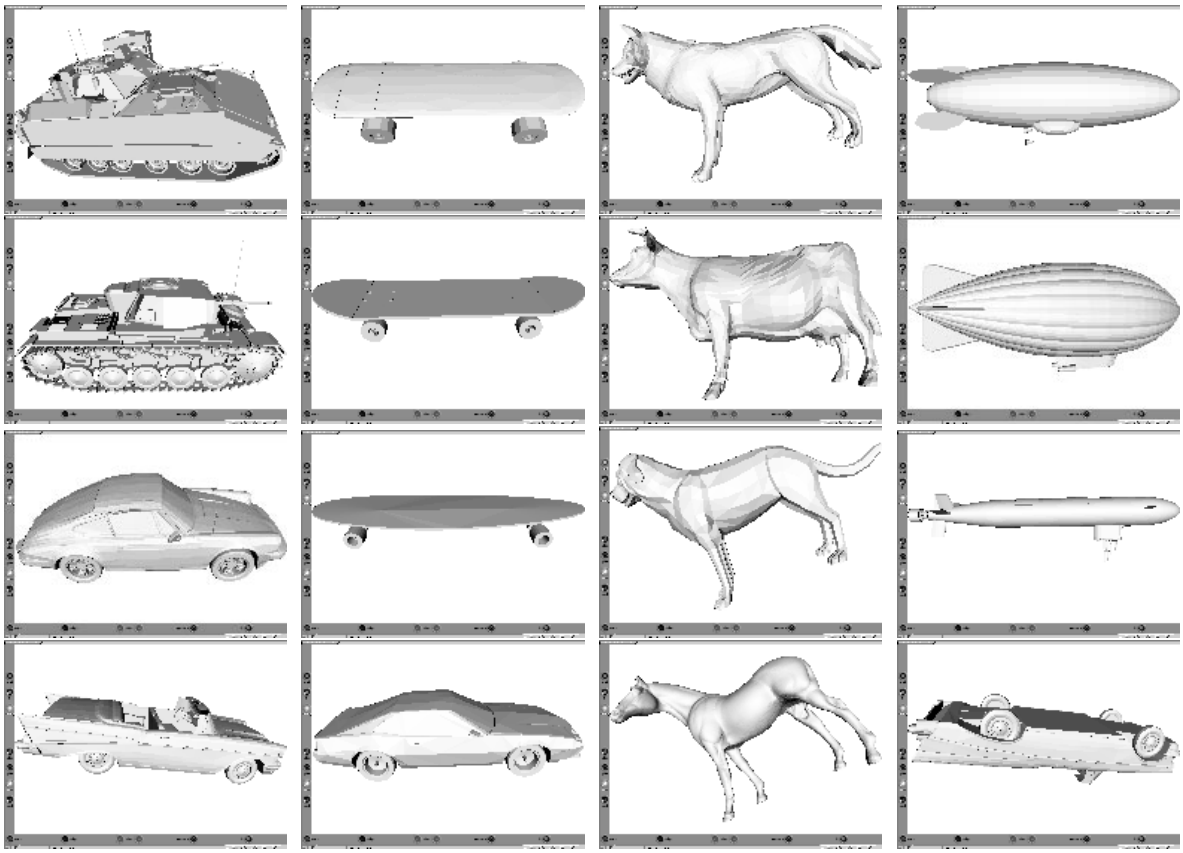

Figure 2: Query results using the Gaussian curvature method for the Princeton database. Each column illustrates a query. The top row shows the query objects, the second row the nearest neighbour, the third and fourth row show the second and third nearest neighbour, respectively.

the query.

For the Princeton database, figure 1 shows for each weighting scheme the precision, i.e. the proportion of returned models that are in the same class as the query object, as a function of the number of models returned.

Figure 2 illustrates for the Princeton database shape retrieval using the Gaussian curvature method. Each column shows a query result. We also examined query results on this database using our experimental shape retrieval engine that can also be found at our web site [35]. The search engine allows the user to choose a method and generate random queries or specific queries identified by a number identifying a model. The reader can verify the query results from figure 2 selecting for the Princeton database the Gaussian curvature method and query the models in the first row of figure 2 from left to right using the numbers 130, 110, 5 and 12.

To avoid the drawbacks of a classification by functionality, as in the Princeton database, we made a test database with a shape-based classification. First we collected 684 VRML models, mostly airplanes, from the World Wide Web. From this collection we classified 512 models into six categories: 242 conventional air planes, 60 delta-jets, 45 multi-fuselages, 19 biplanes, 10 helicopters and 136 other models. This classification was purely on the basis of shape, not on the type of object. We did not classify the remaining 172 models, because it was not clear

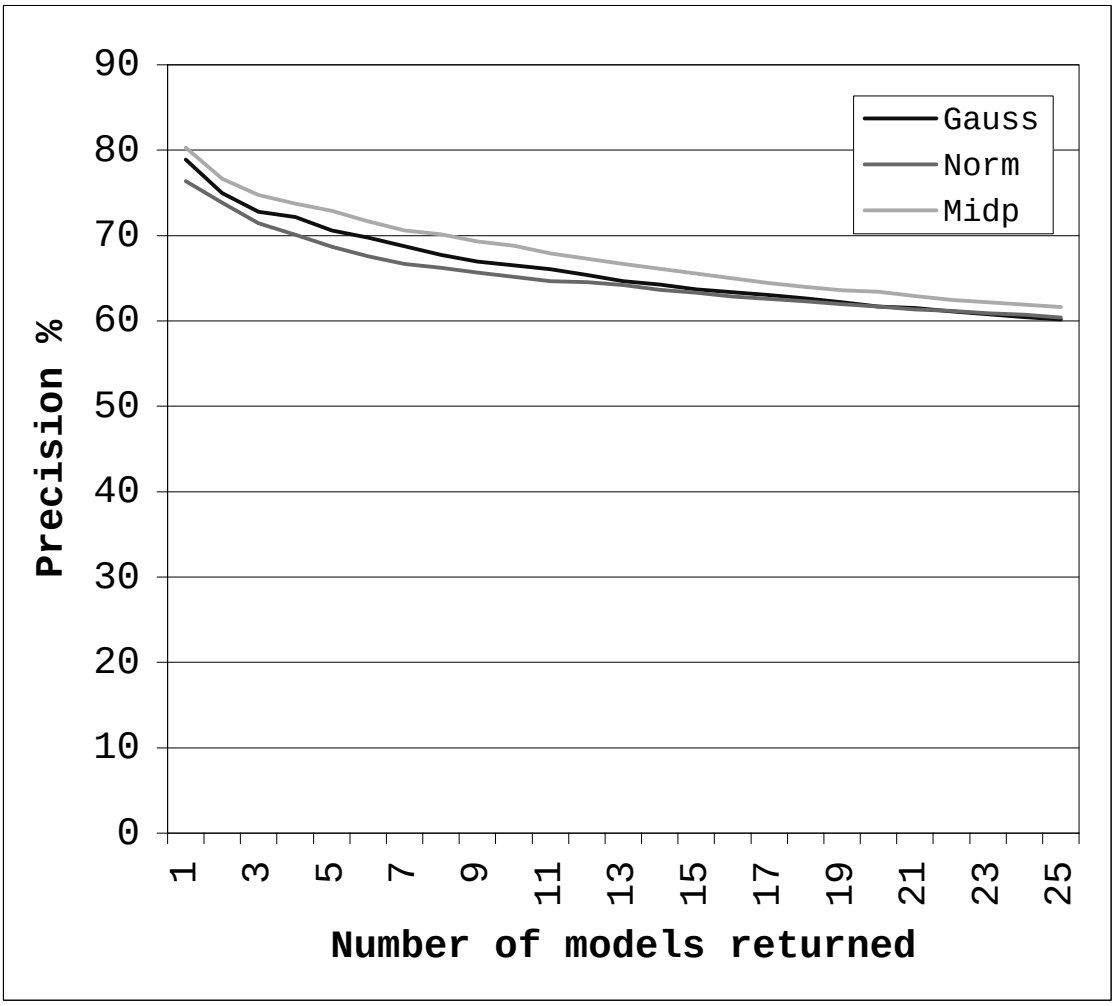

Figure 3: Precision versus number of models returned for the Utrecht database.

to which class these models should belong, looking at their shape. For verification, on our web site the reader can query the complete database containing all 684 models, download the complete database, and a database containing the 512 classified models only.

For our test database (the Utrecht database), figure 3 shows the precision as a function of the number of returned models n, i.e. the proportion of returned models that are in the same class as the query object.

Figure 4 illustrates for the Utrecht database shape retrieval using the midpoint method. The reader can verify the query results from figure 4 selecting for the Utrecht database the midpoint method and query the models in the top row of figure 4 from left to right using the numbers 532, 389, and 129. The last column of figure 4 shows the signatures of the models in the third column.

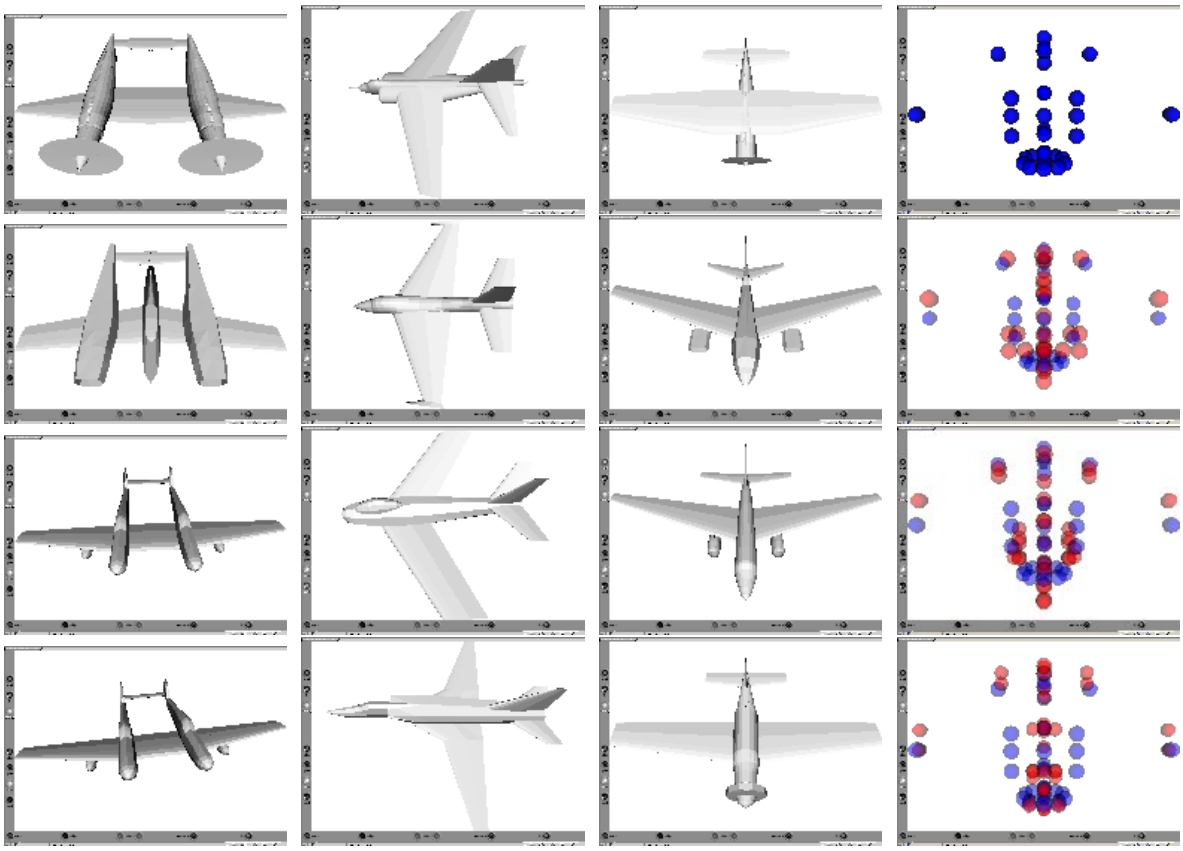

Figure 4: Query results using the midpoint method for the Utrecht database. Each of the first three columns illustrates a query in the same way as in Figure 2. The last column shows the signatures of the model in the third column. The blue spheres denote weighted points of the query object and the red spheres denote weighted points of the retrieved object.

5 Discussion

Figures 1 and 3 demonstrate that our methods can be fairly effective in retrieving similar 3D models. In the first case the Gaussian curvature method was the best and in the second case the midpoint method. In both cases the normal variation method is less effective than the Gaussian and midpoint method.

From the results in table 2 we observe that our methods perform better than the $D2$ shape distribution based method described in [21, 22].

The difference in precision between figures 1 and 3 is caused by the difference of the class sizes and by the difference in classification of both databases. Figure 3 corresponds to the Utrecht database. Figure 1 corresponds to the Princeton database, were the classification is based on the function of models rather than their shape. This causes a number of ambiguous classifications in terms of shape. This is illustrated in Figure 2. The first query object is a tank, only the nearest neighbour is also a tank. The other three tanks are not found among the 3-nearest neighbours. This is not surprising as these cars and tanks have similar shape. However, because tanks and cars are in different classes, this result will cause a lowering of

the precision in figure 1. Also, the smaller class sizes in the Princeton database will cause a lowering of the precision in figure 1.

The second query object is a skate board. The other two skate boards in the Princeton database are found as nearest neighbour and second nearest neighbour. The third query object is a wolf of the class animals. Also, the three nearest neighbours are animals. The class animals consists of 5 models. The fifth animal, a dinosaur, is found on rank 5 instead of rank 4. These results are satisfactory.

Our method is designed only for single polyhedral objects, it is not robust against outliers in the VRML scene. So if the scene contains more than a single object, the results are distorted. This is illustrated in figure 5, showing a blimp. The bottom of the figure shows some text modelled as VRML, which severely influences the matching. Indeed, Figure 2 shows a query with a blimp. The blimp class contains two other blimps, but only one blimp is found as nearest neighbour. The other blimp of figure 5 is found on rank 92, due to the outliers.

Figure 4 shows three queries on the Utrecht database obtained with the midpoint method. The first query object is classified as a multi-fuselage. The query returns three models, that are also classified as multi-fuselages. The second and third query items are classified as conventional air planes and the returned models are also classified as conventional air planes. If we take a closer look at the third query, we expect that the air plane ranked third would be first, because of the orientation of the wings. This result is explained as follows. If we look at the signatures of the models in the last column, the signatures contain only weighted points at the tips of the wings and near the fuselage. This is so, because the VRML models contain only vertices at these places. Hence, the shapes of the wings contribute only a few weighted points to the signature. Overall, the query results on the Utrecht database are satisfactory.

The experiments show that our approach can actually be used in 3D shape comparison, for example as an effective filter, after which more detailed comparisons can be made. Our experimental results demonstrate that our geometric approach to shape matching is fairly effective in finding objects similar to a query object. Since the similarity between two shapes is computed using the Proportional Transportation Distance, which satisfies the triangle inequality, our method is suitable for use in indexing very large collections of models.

6 Conclusions

In this paper we have presented the first ideas you would think of in trying to incorporate spatial distribution of shape information for comparisons, in much the same way as the shape distribution based methods described in [21, 22] implement the first ideas you would think of for making shape distributions without spatial relations. In this sense the results provide a lower bound of any following attempt to do retrieval of polyhedral models using spatial distribution. Compared to the best of the shape distribution based methods described in [21, 22] the experimental results presented in table 2 show that the performance of our approach is better. Of course, there are many improvements that can be made to our method. E.g., for grid cells that intersect facets of the VRML model but contain no vertices of it, also a salient point should be added to the signature of the model. Instead of using the vertices of the mesh directly, it may be better to resample the mesh to generate evenly spaced samples on the surfaces of the model, and use those to generate the signature. Also, it would be interesting to investigate the method for grids consisting of smaller cells. We expect that the results will be more precise, but for small cells we cannot generate a salient point for each

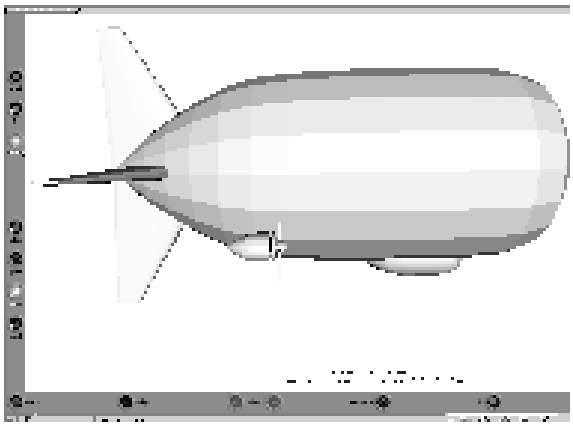

Figure 5: Blimp ranked far away from the blimp used as query in figure 2.

cell, because the running time for computing the PTD between signatures containing many weighted points would be very high. In the preprocessing stage, the pose estimation can be improved in cases for which the PCA method obtains similar eigenvalues. Another important issue for further research is the development of publicly available benchmark databases so that different shape matching methods can be compared.

7 Acknowledgements

We thank Geert-Jan Giezeman from our institute for implementing the 3D shape search engine, we thank Bram de Smit from the Integrated Concept Advancement Group, Delft University of Technology, for providing us with the Utah teapot data sets, and Patrick Min from the Princeton Shape Retrieval and Analysis Group for providing us the Princeton database.

References

[1] M. Ankerst, G. Kastenmuller, H.-P. Kriegel, and T. Seidl. 3D Shape Histograms for Similarity Search and Classification in Spatial Databases. *Symposium on Large Spatial Databases*, 207-226, 1999.

[2] J. Barros, J. French, W. Martin, P. Kelly, and M. Cannon. Using the triangle inequality to reduce the number of comparisons required for similarity-based retrieval. *Proc. of SPIE*, vol. 2670, 392-403, 1996.

[3] R. Basri, L. Costa, D. Geiger, and D. Jacobs. Determining the similarity of deformable shapes. *Vision Reasearch*, 38:2365-2385, 1998.

[4] P. Braß. On the non-existence of Hausdorff-like metrics for fuzzy sets. Freie Universität Berlin, Fachbereich Mathematik und Informatik, TR B 00-02, 2000.

[5] D. Brodsky, and B. Watson. Model Simplification Through Refinement. *Proc. Graphics Interface*, 221-228, 2000.

[6] C.R. Calladine. Gaussian Curvature and Shell Structures. *The Mathematics of Surfaces*, 179-196, Oxford University Press, 1985.

[7] R.J. Campbell, and P.J. Flynn. A Survey of Free-Form Object Representation and Recognition Techniques. *Computer Vision and Image Understanding*, (81):166–210, 2001.

[8] V. Cicirello, and W.C. Regli, Machining Feature-based Comparisons of Mechanical Parts. *International Conference on Shape Modeling and Applications (SMI 2001)*, 176-185, 2001.

[9] Code for the Earth Movers Distance (EMD), `http://robotics.stanford.edu/~rubner/emd/default.htm`.

[10] I. Cohen, N. Ayache, and P. Sulger, Tracking Points on Deformable Objects Using Curvature Information. *ECCV '92*, Lecture notes in Computer Science 588, 458-466, 1992.

[11] S. Cohen, and L. Guibas. The Earth Mover's Distance under Transformation Sets. *Proc. of the 7th IEEE Int. Conf. on Computer Vision*, 173-187, 1999.

[12] C.M. Cyr, and B. Kimia. 3D Object Recognition Using Shape Similiarity-Based Aspect Graph. *Int. Conf. on Computer Vision (ICCV01)*, I:254-261, 2001.

[13] D. DeCarlo, and D. Metaxas. Shape evolution with structural and topological changes using blending. *PAMI*, 20(11):1186-1205, 1998.

[14] M. Elad, A. Tal, and S. Ar. Content Based Retrieval of VRML Objects - An Iterative and Interactive Approach *Eurographics Multimedia Workshop*, 97-108, September 2001.

[15] T. Funkhouser, P. Min, M. Kazhdan, J. Chen, A. Halderman, D. Dobkin, and D. Jacobs. A Search Engine for 3D Models. *Submitted for publication*, 2002.

[16] P. Giannopoulus, and R.C. Veltkamp. A pseudo-metric for weighted point sets *To appear in European Conf. on Computer Vision 2002.*

[17] M. Hilaga, Y. Shinagawa, T. Kohmura, and T.L. Kunii. Topology Matching for Fully Automatic Similarity Estimation of 3D Shapes. *SIGGRAPH 2001*, 203-212, August 2001.

[18] M. Kazhdan, B. Chazelle, D. Dobkin, A. Finkelstein, and T. Funkhouser. A Reflective Symmetry Descriptor. *To appear in European Conf. on Computer Vision 2002*, `http://www.cs.princeton.edu/~funk/ecvv02.pdf`.

[19] J. Löffler, Content-based Retrieval of 3D models in Distributed Web Databases by Visual Shape Information, *Int. Conf. on Information Visualisation (IV2000)*, 2000.

[20] M. Novotni, and R. Klein, A Geometric Approach to 3D Object Comparison, *Int. Conf. on Shape Modeling and Applications (SMI 2001)*, 167-175, 2001.

[21] R. Osada, T. Funkhouser, B. Chazelle, and D. Dobkin, Matching 3D Models with Shape Distributions. *Int. Conf. on Shape Modeling and Applications (SMI 2001)*, 154-166, 2001.

[22] R. Osada, T. Funkhouser, B. Chazelle, and D. Dobkin, Shape Distributions *To appear in ACM Transactions on Graphics 2002.* `http://www.cs.princeton.edu/~funk/tog02.pdf`.

[23] E. Paquet, A. Murching, T. Naveen, A. Tabatabai, and M. Rioux, Description of Shape Information for 2-D and 3-D Objects, *Signal Processing: Image Communication*, (16):103–122, 2000.

[24] M. Petrou, and P. Bosdogianni, *Image Processing: The Fundamentals.* John Willey, 1999.

[25] A. Pugh. *Polyhedra: A Visual Approach.* University of California Press, Berkeley, California, 1993.

[26] Rhinoceros: NURBS modeling for Windows, `http://www.rhino3d.com/`.

[27] J. Rossignac, and P. Borrel, Multi-resolution 3D approximation for Rendering Complex Scenes. *Geometric Modeling in Computer Graphics*, 455-465, 1993, Springer Verlag.

[28] Y. Rubner, C. Tomasi, and L.J. Guibas, A Metric for Distributions with Applications to Image Databases *IEEE Int. Conf. on Computer Vision*, 59-66, 1998.

[29] H.-Y. Shum, and M. Hebert, and K. Ikeuchi, On 3D Shape Similarity, *Proc. IEEE Computer Vision and Pattern Recognition*, 526-531, 1996.

[30] D. Terzopoulus, and D. Metaxas. Dynamic models with local and global deformations: Deformable superquadrics. *PAMI*, 13(7):703-714, 1991.

[31] R.C. Veltkamp, Shape Matching: Similarity Measures and Algorithms, *Int. Conf. on Shape Modeling and Applications (SMI 2001)*, 188-197, 2001.

[32] J. Vleugels, and R.C. Veltkamp Efficient Image Retrieval through Vantage Objects, *Pattern Recognition*, 35(1):69-80, 2002.

[33] D.V. Vranić, and D. Saupe, 3D Shape Descriptor Based on 3D Fourier Transform, *Proc. of the EURASIP Conference on Digital Signal Processing for Multimedia Communications and Services (ECMCS 2001)*, Budapest, Hungary, September 2001.

[34] 3D Model Search Engine, `http://shape.cs.princeton.edu/search.html`.

[35] 3D Shape Retrieval Engine, `http://www.cs.uu.nl/centers/give/imaging/3Drecog/3Dmatching.html`.