

## Effective Similarity Search on Voxelized CAD Objects

Hans-Peter Kriegel<sup>\*</sup>, Peer Kröger<sup>\*</sup>, Zahi Mashael<sup>\*</sup>, Martin Pfeifle<sup>\*</sup>, Marco Pötke<sup>†</sup>, Thomas Seidl<sup>‡</sup>

<sup>\*</sup>*Institute for Computer Science, University of Munich*

<sup>†</sup>*sd&m AG software design & management*

<sup>‡</sup>*RWTH Aachen University*

{kriegel | kroegerp | mashael | pfeifle | poetke | seidl}@dbs.informatik.uni-muenchen.de

### Abstract

*Similarity search in database systems is becoming an increasingly important task in modern application domains such as multimedia, molecular biology, medical imaging and many others. Especially for CAD applications, suitable similarity models and a clear representation of the results can help to reduce the cost of developing and producing new parts by maximizing the reuse of existing parts. In this paper, we adapt two known similarity models to voxelized 3-D CAD data and introduce a new model based on eigenvectors. The experimental evaluation of our three similarity models is based on two real-world test datasets. Furthermore, we introduce hierarchical clustering as a new and effective way to analyse and compare similarity models. We show that both our similarity model as well as our evaluation procedure are suitable for industrial use.*

### 1. Introduction

In the last ten years, an increasing number of database applications has emerged for which efficient and effective support for similarity search is substantial. The importance of similarity search grows in application areas such as multimedia, medical imaging, molecular biology, computer aided engineering, marketing and purchasing assistance, etc. [13, 1, 9, 10, 2, 6, 7, 16]. Particularly, the task of finding similar shapes in 2-D and 3-D becomes more and more important. Examples for new applications that require the retrieval of similar 3-D objects include databases for molecular biology and medical imaging.

The development, design, manufacturing and maintenance of modern engineering products is a very expensive and complex task. Effective similarity models are required for two- and three-dimensional CAD applications to cope with rapidly growing amounts of data. Shorter product cycles and a greater diversity of models are becoming decisive competitive factors in the hard-fought automobile and plane market. These demands can only be met if the engineers have an overview of already existing CAD parts. In this paper we introduce a new and very effective similarity model for 3-D CAD data, which helps to find and group similar

parts. This model is particularly suitable for voxelized data, which naturally occur in CAD applications. The experimental evaluation is based on two real-world test data sets of our industrial partners, a German car manufacturer and an American plane producer. Both data sets consist of high resolution voxelized data.

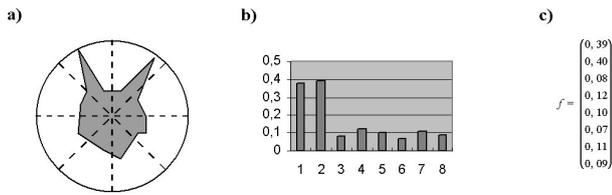
The remainder of the paper is organized as follows: In Section 2 we review already existing spatial similarity models and provide a classification of the techniques into feature-based models and directly geometric models. Section 3 provides the basis for similarity models based on voxelized objects. Furthermore, we adapt two known similarity models to voxelized 3-D data and introduce a new model based on eigenvectors. In Section 4, we introduce hierarchical clustering as a new and effective way to analyse similarity models. We empirically show the superiority of the eigenvector model compared to the other two models. Based on hierarchical clustering we sketch in Section 5 a possible user-friendly prototype which makes best use of the data by browsing through a class hierarchy. The paper concludes in Section 6 with a short summary and a few remarks on future work.

### 2. Related Work

In recent years, considerable work on similarity search in database systems has been published. Many of the previous approaches, however, deal with 1-D or 2-D data, such as time series, digital images or polygonal data, most of them do not support 3-D objects. In this section, we discuss some competing approaches to establish similarity measures. We provide a classification of the techniques into feature-based models and direct geometric models.

#### 2.1. Feature-Based Similarity

A widely used class of similarity models is based on the paradigm of feature vectors. The basic idea is as follows: Using a feature transform, the objects are mapped onto a feature vector in an appropriate multidimensional feature space. The similarity of two objects is then defined as the proximity of their feature vectors in the feature space: The



**Figure 1:** Section coding of 2-D regions: **a)** Original space and object. **b)** Corresponding histogram. **c)** Corresponding feature vector.

closer their feature vectors are located, the more similar two objects are considered.

Several reasons lead to the wide use of feature-based similarity models: First, the more complex the objects are, the more difficult it may be to find an appropriate similarity distance function. A second reason wherefore feature-based similarity models are quite popular is that they may be easily tuned to fit to specific applications. In general, this task is performed in close cooperation with domain experts who specify appropriate features and adapt them to the specific requirements. Since the existing techniques for query processing are independent from the particular definition of the features, efficient support may be provided without an in-depth insight into the application domain.

Examples where the paradigm of feature-based similarity has been successfully applied to the retrieval of similar spatial objects include structural features of 2-D contours [20], angular profiles of polygons [5], rectangular covers of regions [13], algebraic moment invariants [9], and 2-D section coding [6]. Non-geometric applications include similarity search on time series [1, 10], and on color histograms in image databases [22, 9], among several others.

Mehrotra and Gary suggest the use of boundary features for the retrieval of shapes [20]. Here, a 2-D shape is represented by an ordered set of surface points, and fixed-sized subsets of this representation are extracted as shape features. All of these features are mapped to points in multidimensional space which are stored using a Point Access Method (PAM). This method is essentially limited to two dimensions.

Jagadish proposes a technique for the retrieval of similar shapes in two dimensions [13]. He derives an appropriate object description from a rectilinear cover of an object, i.e. a cover consisting of axis-parallel rectangles. The rectangles belonging to a single object are sorted by size, and the largest ones serve as retrieval key for the shape of the object. Though this method can be generalized to three dimensions by using covers of hyperrectangles, it has not yet been evaluated for real world 3-D data.

Agrawal et al. present a method for similarity search in a sequence database of one-dimensional data [1]. The se-

quences are mapped onto points of a low-dimensional feature space using a Discrete Fourier Transform, and then a PAM is used for efficient retrieval. This technique was later generalized for subsequence matching [10], and searching in the presence of noise, scaling, and translation [2]. However, it remains restricted to one-dimensional sequence data.

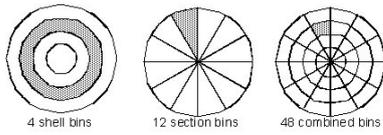
**Histograms as Feature Vectors.** Histograms represent a quite general class of feature vectors which have been successfully applied to several applications. For any arbitrary distribution of objects, a histogram represents a more or less fine grained aggregation of the information. The general idea is to completely partition the space of interest into disjoint regions which are called cells, and to map every object onto a single bin or to distribute an object among a set of bins of the corresponding histogram. Then a histogram can be transformed directly into a feature vector by mapping each bin of the histogram onto one dimension (attribute) of the feature vector. The histogram approach applies to geometric spaces as well as to non-geometric spaces.

A popular example for the use of histograms to define the similarity of complex objects is the color histogram approach which is a core component of the QBIC system [22, 9]. Among other techniques, color histograms are used to encode the percentage of colors in an image [11]. Our second example is taken from a spatial database application: The 2-D section coding approach [7] represents a particular histogram technique that is used in the S3 system [6] for the retrieval of similar mechanical parts. For each object, the circumscribing circle is decomposed into a fixed number of sectors around the center point. For each sector, the fraction of the area is determined that is overlapped by the object. Altogether, the resulting feature vector is a histogram over the 2-D, whose bins represent the corresponding 2-D sectors. Figure 1 illustrates the technique by an example with 8 sectors. This approach, however, is also limited to two dimensions since a linear ordering of the boundary is required. In the 3-D, there is no canonical linearization of the two-dimensional boundary of arbitrary solids.

In [14, 4] the retrieval of similar 3-D objects from a biomolecular database was investigated. The introduced models are based on 3-D shape histograms, where three different approaches were used for space partitioning: shell bins, section bins and combined bins (cf. Figure 2). Unfortunately, these models are not inherently suitable for voxelized data which are axis-parallel.

## 2.2. Geometry-Based Similarity

A class of models that is to be distinguished from the feature-based techniques are the similarity models that are defined by directly using the geometry. Two objects are considered similar if they minimize a distance criterion that is purely defined by the geometry of the objects. Exam-



**Figure 2:** Shells and sections as basic models for shape histograms. In each of the 2-D examples, a single bin is marked.

ples include the similarity retrieval of mechanical parts, the difference volume approach, and the approximation-based similarity model for 3-D surface segments:

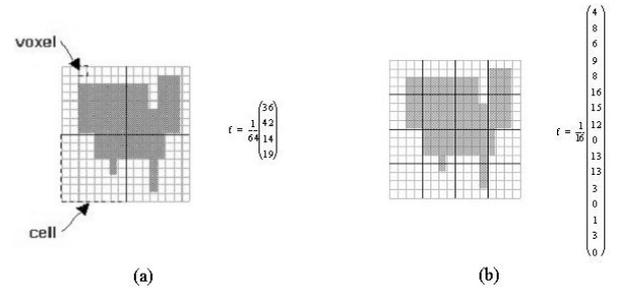
**Rotational Symmetric Mechanical Parts.** In [23], a method is presented to retrieve similar mechanical parts from a database. The similarity criterion is defined in terms of tolerance areas which are specified around the query object. All objects that fit into the tolerance area count for being similar. Although the parts are 3-D, only their 2-D contour is taken into account for the retrieval technique.

**Difference Volume Approach.** The difference volume or error volume of spatial objects is a promising approach which has been already successfully applied to medical images, for instance [12, 24]. Furthermore, extensions such as the combination with methods from mathematical morphology have been investigated on a tumor database [17]. However, they considered only 2-D images. A competing approach is based on a new geometric index structure as suggested in [16]. The basic idea of this solution is to use the concept of hierarchical approximations of the 3D objects to speed up the search process.

**Approximation-based Similarity of Surface Segments.** The retrieval of similar 3-D surface segments is a task that supports the docking search for proteins in biomolecular databases. Following the approximation-based model, the similarity of 3-D surface segments is measured by their mutual approximation error with respect to a given multi-parametric surface function which serves as the underlying approximation model. To state it simply, two segments are the more similar, the better they fit to the approximation of the partner segment [18].

### 3. Similarity Models

In this section, we describe our new similarity models which are based on two major techniques: First, the voxel approximations of the objects are transformed into shape histograms. Second we use these histograms as intuitive feature vectors and define similarity of two spatial objects as the vicinity of their corresponding feature vectors.



**Figure 3:** Space partitioning with (a) 4 cells and (b) 16 cells. The corresponding feature vectors are depicted on the right hand side, respectively.

### 3.1. Voxelized CAD Objects

Engineering products can be regarded as a collection of individual, three-dimensional parts. Each of these parts may consist of a complex and an intricate geometric shape with a very high precision. Accurate representations of CAD surfaces are typically implemented by parametric bicubic surfaces, including Hermite, Bézier, and B-spline patches. For many operations, such as graphical display or the efficient computation of surface intersections, these parametric representations are too complex [21]. As a solution, approximative polygon (e.g. triangle) meshes can be derived from the accurate surface representation. These triangle meshes allow for an efficient and interactive display of complex objects. In order to apply spatial indexing, often, a coarser, conservative approximation of the parts, by means of voxels, is applied.

A basic algorithm for the 3D scan-conversion of polygons into a voxel-based occupancy map has been proposed by Kaufmann [15]. Similarly to the well-known 2D scan-conversion technique, the runtime complexity to voxelize a 3D polygon is  $O(n)$ , where  $n$  is the number of generated voxels. If we apply this conversion to the given triangle mesh of a CAD object, a conservative approximation of the part surface is produced. In the following, we assume a uniform three-dimensional voxel grid covering the global product space. The grid resolution determines the finest possible granularity for the approximation of the objects. By means of space filling curves, each cell of the grid can be encoded by a single integer number, and thus an extended CAD object is represented by a set of integers.

### 3.2. Shape Histograms

Histograms are usually based on a complete partitioning of the data space into disjoint cells which correspond to the bins of the histograms.

We divide the 3-D data space into axis parallel, equi-sized partitions (cf. Figure 3, which provides an illustration in 2-D). This kind of space partitioning is especially suitable for voxelized data, as cells and voxels are of the same shape, i.e. cells can be regarded as coarse voxels.

Each of these partitions is assigned to one or several bins in a histogram, depending on the specific similarity model. Thereby the partitions are numbered from left to right, top to bottom. By scaling the number of partitions, the number of dimensions of the feature vector can be regulated (cf. Figure 3). Obviously, the more partitions we use, the more small differences between the objects become decisive.

By means of the resulting feature vectors, the similarity of two objects can be defined as follows.

**Definition 1 (Feature-Based Object Similarity)** *Let  $O$  be the domain of the objects and  $F : O \rightarrow \mathbb{R}^d$  be a mapping of the objects into the  $d$ -dimensional feature space. Furthermore, let  $\text{dist} : \mathbb{R}^d \times \mathbb{R}^d \rightarrow \mathbb{R}$  be a distance function between two  $d$ -dimensional feature vectors. Then  $\text{fdist} : O \times O \rightarrow \mathbb{R}$  is defined as follows:*

$$\text{fdist}(Obj_1, Obj_2) = \text{dist}(F(Obj_1), F(Obj_2)).$$

There exist a lot of distance functions which are suitable for similarity search. In the literature, often the  $L_p$ -norm is used, as for instance the Manhattan distance ( $p = 1$ ) or the Euclidean distance ( $p = 2$ ). Throughout our experiments (cf. Section 4) the common Euclidean distance was used.

### 3.3. Normalization

Similarity models for CAD data should recognize similar parts, independently of their spatial location. The four, respectively five, tires of a car are similar, although they are located differently. Furthermore, reflected parts, e.g. the right and left front door of a car, should be recognized as similar as far as design is concerned. If we look at the production, reflected parts are no longer similar but have to be treated differently. Likewise, the actual size of the parts may or may not exert influence on the similarity model. To sum up, a similarity model for CAD data should take translation and rotation invariances into account whereas reflection and scaling invariances have to be tuneable.

CAD objects are designed and constructed in a standardized position, normalized to the center of the coordinate system. We store each object normalized w.r.t. translation and scaling in the database. Furthermore, we store the scaling factors for each of the three dimensions, so that we can (de)activate scaling invariance depending on the users needs at runtime. In the case of CAD applications, not all possible rotations are considered, but only  $90^\circ$ -rotations. This yields 24 different possible positions for each object. For similarity search, where we are not confined to  $90^\circ$ -rotations, we can apply principal axis transformation in or-

der to achieve invariance with respect to rotation. Taking also reflection into account, we may obtain  $24 \cdot 2 = 48$  varying positions. We could achieve  $90^\circ$ -rotation and reflection invariance by storing 48 different feature vectors for each object in the database or by carrying out 48 different permutations at runtime. As we want to decide at runtime whether we want to consider reflection invariance or not, we chose the second variant. Throughout our experiments, we considered translation, reflection, scaling and  $90^\circ$ -rotation invariance.

Taking all these transformations into account, we get the following extended similarity definition.

### Definition 2 (Extended Feature-Based Object Similarity)

*Let  $O$  be the domain of the objects,  $F : O \rightarrow \mathbb{R}^d$  a mapping of the objects into the  $d$ -dimensional feature space, and  $\text{dist} : \mathbb{R}^d \times \mathbb{R}^d \rightarrow \mathbb{R}$  a distance function between two  $d$ -dimensional feature vectors. Furthermore, let  $C$  be a set of all user-dependent combinations of translation, scaling, rotation and reflection transformations. Then  $\text{fdist} : O \times O \rightarrow \mathbb{R}$  is defined as follows:*

$$\text{fdist}(Obj_1, Obj_2) = \min \{ \text{dist}(F(Obj_1), F(T(Obj_2))) \mid \forall T \in C, T : O \rightarrow O \}$$

### 3.4. Spatial Features

After partitioning the data space, we have to determine the spatial feature of the objects for each grid cell depending on the chosen model. In order to do that we first have to introduce some notations:

The data space is partitioned in each dimension into  $p$  grid cells. Thus, our histogram will consist of  $k \cdot p^3$  bins where  $k \in \mathbb{N}$  depends on the model specifying the kind and number of features extracted from each cell. For a given object  $o$ , let  $V^o = \{v \in V_i^o \mid 1 \leq i \leq p^3\}$  be the set of voxels that represents  $o$  where  $V_i^o$  are the voxels covered by  $o$  in cell  $i$ .  $\bar{V}^o \subseteq V^o$  denotes the set of voxels at the surface of the objects and  $\dot{V}^o \subseteq V^o$  denotes set of the voxels inside the object, such that  $\bar{V}^o \cup \dot{V}^o = V^o$  and  $\bar{V}^o \cap \dot{V}^o = \emptyset$  holds.

Let  $f_o$  be the computed feature vector of an Object  $o$ . The  $i$ -th value of the feature vector of object  $o$  is denoted by  $f_o^{(i)}$ .

Let  $r$  be the number of voxels of the dataspace in each dimension. In order to ensure a unique assignment of the voxels to a grid cell, we presume that  $\frac{r}{p} \in \mathbb{N}$ .

#### 3.4.1 The Volume Model

A simple and established approach to compare two objects is based on the number of the object voxels  $|V_i^o|$  in each cell  $i$  of the partitioning. In the following, this model is referred to as *Volume Model*. Each cell represents one dimension in the feature vector of the object. The  $i$ -th dimension

of the feature vector ( $1 \leq i \leq p^3$ ) of object  $o$  can be computed by the normalized number of voxels of  $o$  lying in cell  $i$ , formally:

$$f_o^{(i)} = \frac{|V_i^o|}{K} \quad \text{where} \quad K = \left(\frac{r}{p}\right)^3$$

Figure 3 illustrates the Volume Model for the 2-D case.

### 3.4.2 The Solid-Angle Model

The *Solid-Angle* method [8] measures the concavity and the convexity of geometric surfaces. Let  $K_{c,r}$  be a set of voxels that describes a 3-D voxelized sphere with central voxel  $c$  and radius  $r$ . For each surface-voxel  $\bar{v}$  of an object  $o$  the so called Solid-Angle value is computed as follows. The voxels of  $o$  which are inside  $K_{\bar{v},r}$  are counted and divided by the size of  $K_{\bar{v},r}$ , i.e. the number of voxels of  $K_{\bar{v},r}$ . The resulting measure is called the Solid-Angle value  $SA(\bar{v}, r)$  and can be computed as follows:

$$SA(\bar{v}, r) = \frac{|K_{\bar{v},r} \cap V^o|}{|K_{\bar{v},r}|}$$

A small Solid-Angle value  $SA(\bar{v})$  indicates that an object is convex at voxel  $\bar{v}$ . Otherwise, a high value of  $SA(\bar{v})$  denotes a concave shape of an object at voxel  $\bar{v}$ .

The Solid-Angle values of the cells are transferred into the according histogram bins as described in the following. We distinguish between three different types of cells:

1. Cell  $i$  contains surface-voxels of object  $o$ , i.e.  $\bar{V}_i^o \neq \emptyset$ . The mean of all SA-values of the surface-voxels is computed as the feature value of this cell:

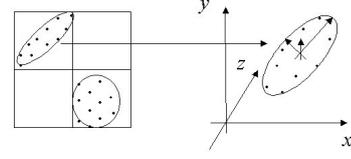
$$f_o^{(i)} = \frac{1}{m} \sum_{j=1}^m SA(\bar{v}_{i_j}, r)$$

where  $\bar{V}_i^o = \{\bar{v}_{i_1}, \dots, \bar{v}_{i_m}\}$ .

2. Cell  $i$  contains only inside-voxels of object  $o$ , i.e.  $\bar{V}_i^o = \emptyset$  and  $V_i^o \neq \emptyset$ . The feature value of this cell is set to 1 (i.e.  $f_o^{(i)} = 1$ ).
3. Cell  $i$  contains no voxels of object  $o$  (i.e.  $V_i^o = \emptyset$ ). The value of the according bin of the histogram is 0 (i.e.  $f_o^{(i)} = 0$ ).

### 3.4.3 The Eigen Value Model

In the following, we introduce a new approach to extract local features which is based on eigen values. The set of voxels of an object can be considered as a set of points in the 3-D data space following a particular scattering. The *Eigen Value Model* uses this scattering of the voxel sets in each



**Figure 4:** Computation of the ellipsoids based on their eigen-values

cell of the partitioning to distinguish the objects by computing the minimum bounding ellipsoid of the voxel set.

A minimum bounding ellipsoid in the 3-D space can be described by 3 vectors (cf. Figure 4). In order to compute these vectors, we consider each voxel  $v$  of the object  $o$  as Euclidian vector  $\vec{v}^o = \begin{pmatrix} x \\ y \\ z \end{pmatrix}$  in the 3-D data space and apply principal axis transformation.

To determine the principal axes of the vectors in cell  $i$ , we first compute their centroid  $\vec{C}_i^o$ :

$$\vec{C}_i^o = \begin{pmatrix} x_C \\ y_C \\ z_C \end{pmatrix} = \frac{1}{|V_i^o|} \begin{pmatrix} \sum_{j=1}^{|V_i^o|} x_j \\ \sum_{j=1}^{|V_i^o|} y_j \\ \sum_{j=1}^{|V_i^o|} z_j \end{pmatrix}$$

After that, for each vector  $\vec{v}^o$  in cell  $i$ , the following translation is carried out:  $\vec{v}^o := \vec{v}^o - \vec{C}_i^o$ .

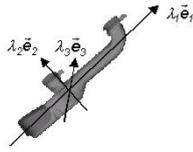
Based on these transformed vectors  $\vec{v}^o$ , the covariance matrix  $\text{COV}_i^o$  for each cell  $i$  can be computed as follows:

$$\text{COV}_i^o = \frac{1}{|V_i^o| - 1} \begin{pmatrix} \sum_{j=1}^{|V_i^o|} x_j^2 & \sum_{j=1}^{|V_i^o|} x_j y_j & \sum_{j=1}^{|V_i^o|} x_j z_j \\ \sum_{j=1}^{|V_i^o|} x_j y_j & \sum_{j=1}^{|V_i^o|} y_j^2 & \sum_{j=1}^{|V_i^o|} y_j z_j \\ \sum_{j=1}^{|V_i^o|} x_j z_j & \sum_{j=1}^{|V_i^o|} y_j z_j & \sum_{j=1}^{|V_i^o|} z_j^2 \end{pmatrix}$$

The eigen vectors  $\vec{e}_i^j$  ( $j = 1, 2, 3$ ) of the matrix  $\text{COV}_i^o$  correspond to the vectors spanning the minimum bounding ellipsoid of the voxel set  $V_i^o$ . The eigen values  $\lambda_i^j$  represent the scaling factors for the eigen vectors (cf. Figure 5). Both eigen values and eigen vectors are determined by the following equation:

$$\text{COV}_i^o \cdot \vec{e}_i^j = \lambda_i^j \vec{e}_i^j$$

The interesting values that are inserted in the bins of the his-



**Figure 5:** Principal axes of a sample object.

togram are the eigen values which describe the scattering along the principal axis of the voxel set. These three values can be computed using the characteristic polynomial:

$$\det(\text{Cov}_i^o - \lambda_i^j \text{Id}) = 0, \quad \text{for } j = 1, 2, 3$$

Using this equation we obtain 3 eigen values which are sorted in descending order in the vector  $\vec{\lambda}_i$ . The highest value represents the variance along the first principal axis, the second value represents the variance along the second principal axis, and the third value represents the variance along the third principal axis.

For each cell  $i$  of the partitioning we compute the vector  $\vec{\lambda}_i$  of the three eigen values as described right above and register it in the according bins of the histogram:

$$f_o^{(i)} = \vec{\lambda}_i = \begin{pmatrix} \lambda_i^1 \\ \lambda_i^2 \\ \lambda_i^3 \end{pmatrix}$$

Note that for  $p^3$  cells we obtain a feature vector with  $3 \cdot p^3$  dimensions.

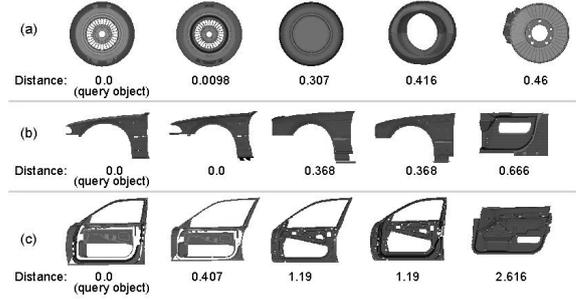
## 4. A New Approach to Evaluate Similarity Models

In this section, we present the results of an exhaustive evaluation based on nearest-neighbor queries and clustering. We also show how the use of the hierarchical clustering algorithm OPTICS [3] is applicable as a user-friendly visualisation tool to meet industrial requirements.

### 4.1. Data Sets

We evaluated the three proposed models on the basis of two real-world datasets. The first one – in the following referred to as *Car Dataset* – contains 177 CAD objects from a German car manufacturer. The Car Dataset contains several groups of intuitively similar objects, e.g. a set of tires, doors, fenders, engine blocks and kinematic envelopes of seats.

The second dataset contains 10,000 CAD objects from an American aircraft producer and in the following is called *Air Dataset*. This dataset contains many small objects (e.g. nuts, bolts, etc.) and a few large ones (e.g. wings).



**Figure 6:** Results of 5-nn queries for the three different models: (a) Volume Model, (b) Solid-Angle Model, (c) Eigen Value Model. The particular distances to the query object are depicted below each object.

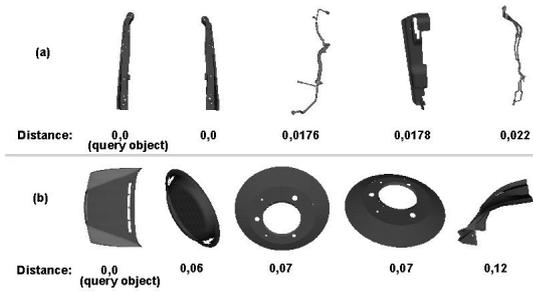
For both, the Car Dataset and the Air Dataset, the data space contains objects represented as voxel approximations using a raster resolution of  $r = 30$ . Furthermore, the data space is partitioned into  $p = 3$  cells in each dimension. Thus,  $|V^o|$  ranges from 1 to  $30^3 = 27,000$  for each object  $o$ . We retrieve  $3^3 = 27$ -dimensional feature vectors for the Volume Model and the Solid-Angle Model. The Eigen Vector Model yields feature vectors of dimensionality  $3 \cdot 3^3 = 81$ .

## 4.2. Evaluation of the Similarity Models

In general, similarity models can be evaluated by computing k-nearest neighbour queries (k-nn queries). A drawback of this evaluation approach is that the quality measure of the similarity model depends on the results of few similarity queries and, therefore, on the choice of the query objects. A model may perfectly reflect the intuitive similarity according to the chosen query objects and would be evaluated as “good” although it produces disastrous results for other query objects. As a consequence, the evaluation of similarity models with sample k-nn queries is subjective and error-prone.

A better way to evaluate and compare several similarity models is to apply a clustering algorithm. Clustering groups a set of objects into classes where objects within one class are similar and objects of different classes are dissimilar to each other. The result can be used to evaluate which model is best suited for which kind of objects.

We evaluated our models using both approaches. In the following, we first discuss the results of the evaluation based on k-nn queries (see Section 4.2.1). In Section 4.2.2 we then evaluate our models using the hierarchical clustering algorithm OPTICS [3].



**Figure 7:** Results of 5-nn queries for the Volume Model. The particular distances to the query object are depicted below each object.

#### 4.2.1 Nearest-Neighbour Queries

**Definition 3 (k-Nearest-Neighbor Query)** Let  $O$  be a set of objects. For a query object  $q \in O$  and a query parameter  $k$ , the  $k$ -nearest neighbor query returns the smallest set  $NN_k(q) \subseteq O$  that contains  $k$  objects from the database, and for which the following condition holds:

$$\forall o \in NN_k(q), \forall o' \in O - NN_k(q) : \text{fdist}(o, q) \leq \text{fdist}(o', q)$$

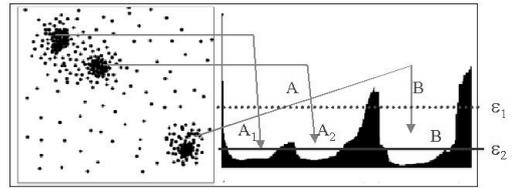
where  $\text{fdist}$  is defined as in Definition 2.

We evaluated the three models described in Section 3.4 using  $k$ -nn queries with  $k = 5$ . We performed the 5-nn queries on the Car Dataset and evaluated the resulting objects according to our intuitive notion of similarity.

The results of the 5-nn queries for each model are presented in Figure 6. We achieved satisfying results for each model depending on the query object, e.g. for a tire, the Volume Model performs very well, yielding objects that are intuitively very similar to the query object (cf. Figure 6(a)). Comparably good results are also produced by the Solid-Angle Model for a part of the wing (cf. Figure 6(b)) and by the Eigen Value Model for a door (cf. Figure 6(c)).

Although all three models deliver rather accurate results for the chosen query objects, we see in Figure 7 that these results are delusive.

Figure 7(a) shows a nearest neighbour query for an object which belongs to a cluster (cf. Figure 10), i.e. there exist several similar parts to this object. The Volume model does not recognize this, although it works perfectly well for other query objects (cf. Figure 6). Furthermore, there might be objects where a nearest neighbour query does not yield any intuitively similar parts (cf. Figure 7(b)). Obviously, we should not discard a similarity model if the chosen query object belongs to noise. This confirms the assumption that the method of evaluating similarity models using several  $k$ -nn queries is subjective and error-prone, due to its dependency on the choice of the query objects.



**Figure 8:** Reachability plot (right) computed by OPTICS for a sample 2d dataset (left).

In the next section, we introduce hierarchical clustering in order to overcome the above described difficulties.

#### 4.2.2 Clustering

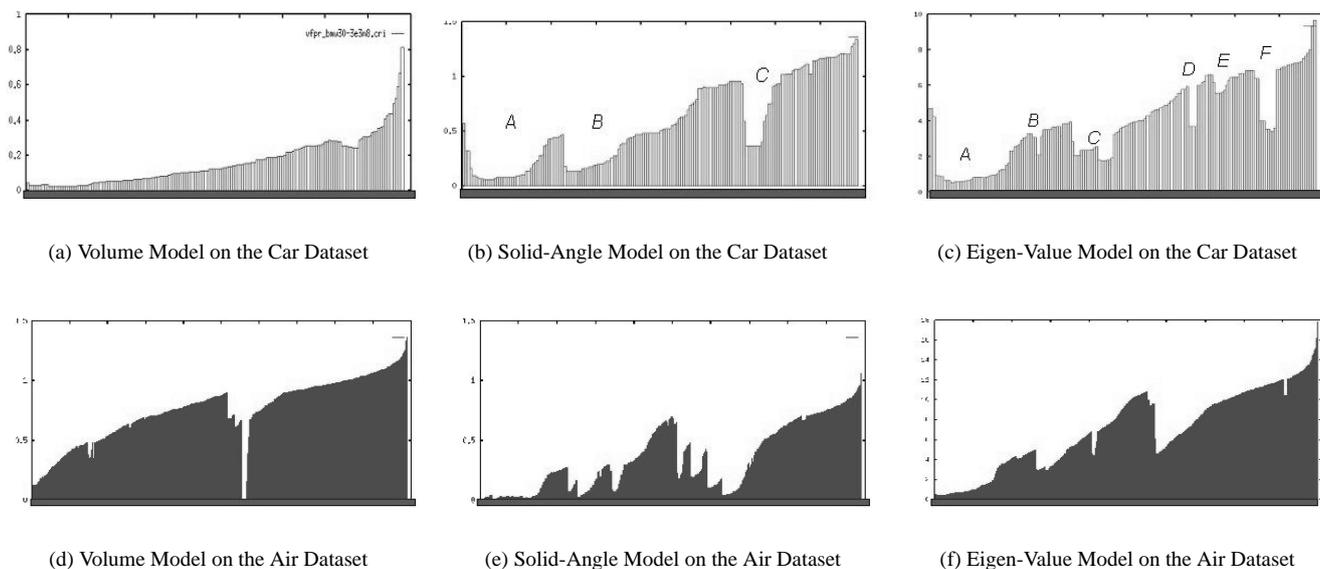
For the more objective evaluation via clustering we used the density-based, hierarchical algorithm OPTICS which is described in full details in [3].

We choose OPTICS due to the following reasons. First, OPTICS is – in contrast to most other algorithms – relatively insensitive to its two input parameters. The authors in [3] state that the input parameters just have to be big enough to retrieve good results. Second, OPTICS is a hierarchical clustering method which yields more information about the cluster structure than a method that computes a flat partitioning of the data (e.g.  $k$ -means[19]).

The output of OPTICS is a linear ordering of the database objects minimizing a binary relation called *reachability* which is in most cases equal to the minimum distance of each database object to one of its predecessors in the ordering. Instead of a dendrogram, which is the common representation of hierarchical clusterings, the resulting reachability-plot is much easier to analyse. The reachability values can be plotted for each object of the cluster-ordering computed by OPTICS. Valleys in this plot indicate clusters: objects having a small reachability value are more similar to their predecessor objects than objects having a higher reachability value.

The reachability plot generated by OPTICS can be cut at any level  $\epsilon$  parallel to the abscissa. It represents the density-based clusters according to the density threshold  $\epsilon$ : A consecutive subsequence of objects having a smaller reachability value than  $\epsilon$  belong to the same cluster. An example is presented in Figure 8: For a cut at the level  $\epsilon_1$  we retrieve two clusters denoted as  $A$  and  $B$ . Compared to this clustering, a cut at level  $\epsilon_2$  would yield three clusters. The cluster  $A$  is split into two smaller clusters denoted as  $A_1$  and  $A_2$  and cluster  $B$  has decreased its size. Usually, for evaluation purposes, a good value for  $\epsilon$  would yield as many clusters as possible.

In the following, we present the results on our two test datasets.



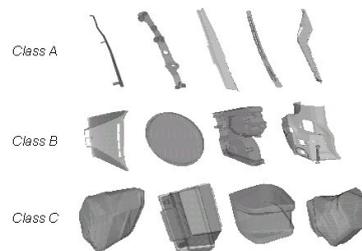
**Figure 9:** Reachability plots computed by OPTICS for the Car Dataset and the Air Dataset.

**Evaluation of the Volume-Model.** The reachability plots computed by OPTICS using the Volume Model for both the Car Dataset and the Air Dataset are depicted in Figure 9(a) and 9(d). In both cases no clear classification of the objects can be found. None of the groups described in Section 4.1 were distinguished by the clustering algorithm. Although we get satisfying results using k-nn queries (cf. Figure 6), the Volume Model is rather ineffective if applied to the whole data set. This indicates the suitability of clustering to evaluate the quality of similarity models.

Let us note that according to the plot the objects are marked with increasing reachability values along the ordering. An analysis of representatives shows that the objects are ordered according to their volume. This might be the explanation why the Volume Model is rather ineffective. Objects with related volume are modeled as similar. Most likely this similarity is too simple.

**Evaluation of the Solid-Angle Model.** The reachability plots computed by OPTICS using the Solid-Angle Model for both the Car Dataset and the Air Dataset are depicted in Figure 9(b) and 9(e).

On the Car Dataset the Solid-Angle Model provides three clusters denoted as *A*, *B*, and *C* in Figure 9(b). We analyzed the resulting clusters by picking samples out of the set of objects grouped in each cluster. The result of this evaluation is presented in Figure 10. As it can be seen, cluster *A* consists mainly of long and thin objects. This might be still inside the intuitive notion of similarity. The same observation can be made for the objects in cluster *C*. But the



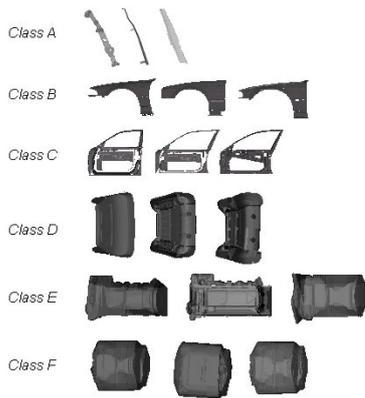
**Figure 10:** Objects in the clusters *A*, *B*, *C* in Figure 9(b) found by OPTICS.

objects that are grouped together in cluster *B* are no more intuitively similar.

Evaluating the Solid-Angle Model using the Air Dataset we made similar observations. The reachability plot computed by OPTICS (cf. Figure 9(e)) yields a clustering with a large number of hierarchical classes. But the analysis of the objects within each cluster displays that intuitively dissimilar objects are counted as similar according to the model. A further observation is the following: objects clustered in different groups are intuitively similar.

To sum up, the Solid-Angle Model does not generate all clusters for the Car Dataset, whereas for the Air Dataset it yields clusters with dissimilar parts. This suggests the conclusion that the Solid-Angle Model is also rather unsuitable as a similarity model for our real-world test datasets.

**Evaluation of the Eigen Value Model.** In contrast to the



**Figure 11:** Objects in the classes A, B, C, D, E, and F in Figure 9(c) found by OPTICS.

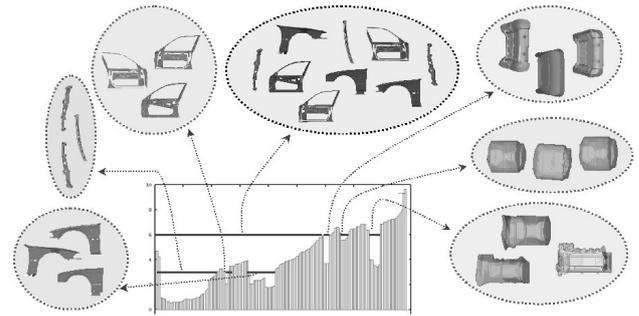
other two approaches, the Eigen Value Model yields valuable results. The plots computed by OPTICS for the Eigen-Value Model are presented in Figure 9(c) and 9(f).

On the Car Dataset (cf. Figure 9(c)) OPTICS finds six clusters which are analysed in Figure 11. Each class consists of intuitive similar objects. Class *A* represents a large number of small and thin objects (similar to the Solid-Angle Model – cf. Figure 10). Class *B* consists of fenders, class *C* represents doors, all objects in class *D* are seats, class *E* consists of engine blocks and class *F* represents kinematic envelopes of seats. This result suggests that based on the Eigen Value Model, OPTICS finds all intuitively classes in the Car Dataset.

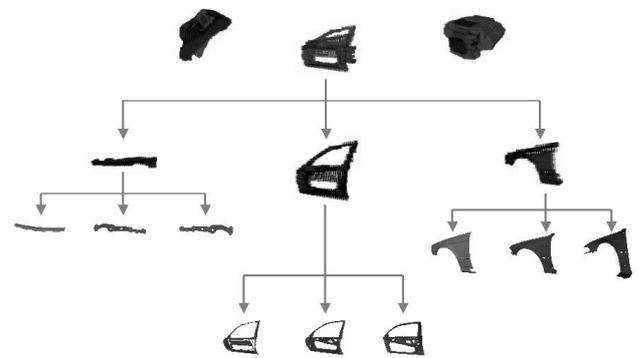
Analysing the Air Dataset with OPTICS based on the Eigen-Value Model yields affirmative results as well. The reachability plot (cf. Figure 9(f)) depicts a clear hierarchy of five clusters. The analysis of these classes confirms that the objects in each cluster are intuitively similar.

## 5. Industrial Application

In this section, we sketch an industrial prototype, called *BOSS* (Browsing OPTICS-Plots for Similarity Search). *BOSS* is based on the Eigen-Value Model and on the evaluation approach using the hierarchical clustering algorithm OPTICS. *BOSS* is an interactive data browsing tool which depicts the reachability plot computed by OPTICS in a user friendly way together with appropriate representatives of the clusters. This clear illustration supports the user in his time-consuming task to find similar parts. From the industrial user’s point of view, *BOSS* meets the following two requirements: (i) The hierarchical clustering structure of the dataset is revealed at a glance. The reachability plot is an intuitive visualisation of the clustering hierarchy which helps to assign each object to its corresponding cluster or to noise,



**Figure 12:** BOSS: Browsing through reachability plots with different density thresholds  $\epsilon$



**Figure 13:** BOSS: Hierarchically ordered representatives.

respectively. Furthermore, the hierarchical representation of the clusters by the reachability plot helps the user to get a quick overview over all clusters and their relation to each other. As each entry in the reachability plot is assigned to one object, we can easily illustrate some representatives of the clusters belonging to the current  $\epsilon$ -value (cf. Figure 12). (ii) The user is not only interested in the shape and the number of the clusters, but also in the specific parts building up a cluster. As for large clusters it is rather difficult to depict all objects, *BOSS* also displays representatives of each cluster. These representatives are simply constructed by superimposing all parts belonging to the regarded cluster. We can browse through the hierarchy of the representatives in the same way as through the OPTICS-Plots (cf. Figure 13).

*BOSS* helps to reduce the cost of developing and producing new parts by maximizing the reuse of existing parts because it allows the user to browse through the hierarchical structure of the clusters in a top-down way. Thus the engineers get an overview of already existing parts and are able to navigate their way through the diversity of existing variants of products, such as cars.

## 6. Conclusions

In this paper, we introduced three different similarity models for voxelized 3-D CAD data. We adapted two known 2-D models and introduced a new model based on eigen vectors. Based on two real-world test data sets we showed the superiority of our new *Eigen Value Model* to the other two models, the *VolumeModel* and the *Solid-AngleModel*. Furthermore, we introduced hierarchical clustering as a new and effective way to analyse and compare similarity models. We showed that hierarchical clustering is more suitable for the evaluation of similarity models than the commonly used  $k$ -nn queries. Based on both our new evaluation procedure and our effective similarity model, we sketched a prototype suitable for industrial use, called BOSS which helps the user to cope with rapidly growing amounts of data, and helps thereby to reduce the cost of developing and producing new parts.

In our future work, we plan to advance our prototype, so that the information contained in the representatives of the clusters is better perceivable.

## References

- [1] R. Agrawal, C. Faloutsos, and A. Swami. "Efficient Similarity Search in Sequence Databases". In *Proc. 4th. Int. Conf. on Foundations of Data Organization and Algorithms (FODO '93)*, Evanston, ILL, volume 730 of *Lecture Notes in Computer Science (LNCS)*, pages 69–84. Springer, 1993.
- [2] R. Agrawal, K.-I. Lin, H. Sawhney, and K. Shim. "Fast Similarity Search in the Presence of Noise, Scaling, and Translation in Time-Series Databases". In *Proc. 21th Int. Conf. on Very Large Databases (VLDB'95)*, pages 490–501, 1995.
- [3] M. Ankerst, M. Breunig, H.-P. Kriegel, and J. Sander. "OPTICS: Ordering Points to Identify the Clustering Structure". In *Proc. ACM SIGMOD Int. Conf. on Management of Data (SIGMOD'99)*, Philadelphia, PA, pages 49–60, 1999.
- [4] M. Ankerst, G. Kastenmüller, H.-P. Kriegel, and T. Seidl. "3D Shape Histograms for Similarity Search and Classification in Spatial Databases". In *Proc. 6th Int. Symposium on Large Spatial Databases (SSD'99)*, Hong Kong, China, volume 1651 of *Lecture Notes in Computer Science (LNCS)*, pages 207–226. Springer, 1999.
- [5] A. Badel, J. Mornon, and S. Hazout. "Searching for Geometric Molecular Shape Complementarity using Bidimensional Surface Profiles". *Journal of Molecular Graphics*, 10:205–211, 1992.
- [6] S. Berchtold, D. Keim, and H.-P. Kriegel. "Using Extended Feature Objects for Partial Similarity Retrieval". *VLDB Journal*, 6(4):333–348, 1997.
- [7] S. Berchtold and H.-P. Kriegel. "S3: Similarity Search in CAD Database Systems". In *Proc. ACM SIGMOD Int. Conf. on Management of Data (SIGMOD'97)*, Tucson, AZ, pages 564–567, 1997.
- [8] M. Connolly. "Shape Complementarity at the Hemoglobin a1b1 Subunit Interface". *Biopolymers*, 25:1229–1247, 1986.
- [9] C. Faloutsos, R. Barber, M. Flickner, J. Hafner, et al. "Efficient and Effective Querying by Image Content". *Journal of Intelligent Information Systems*, 3:231–262, 1994.
- [10] C. Faloutsos, M. Ranganathan, and Y. Manolopoulos. "Fast Subsequence Matching in Time-Series Databases". In *Proc. ACM SIGMOD Int. Conf. on Management of Data (SIGMOD'94)*, Minneapolis, MN, pages 419–429, 1994.
- [11] J. Hafner, H. Sawhney, W. Equitz, M. Flickner, and N. W. "Efficient Color Histogram indexing for Quadratic Form Distance Functions.". *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 17(7):729–736, 1995.
- [12] W. Higgins. "Automatic Analysis of 3-D and 4-D Radiological Images". grant application to the Department of Health and Human Services, December 1990.
- [13] H. Jagadish. "A Retrieval Technique for Similar Shapes". In *Proc. ACM SIGMOD Int. Conf. on Management of Data (SIGMOD'94)*, pages 208–217, 1991.
- [14] G. Kastenmüller, H.-P. Kriegel, and T. Seidl. "Similarity Search in 3D Protein Databases". In *Proc. German Conf. on Bioinformatics (GCB'98)*, Köln, Germany, 1998.
- [15] A. Kaufman. "An Algorithm for 3D Scan-Conversion of Polygons". In *Proc. Eurographics*, pages 197–208, 1987.
- [16] D. Keim. "Efficient Geometry-based Similarity Search of 3D Spatial Databases". In *Proc. ACM SIGMOD Int. Conf. on Management of Data (SIGMOD'99)*, Philadelphia, PA, pages 419–430, 1999.
- [17] F. Korn, N. Sidiropoulos, C. Faloutsos, E. Siegel, and Z. Protopapas. "Fast Nearest Neighbor Search in Medical Image Databases". In *Proc. 22th Int. Conf. on Very Large Databases (VLDB'96)*, pages 215–226, 1996.
- [18] H.-P. Kriegel, T. Schmidt, and T. Seidl. "3D Similarity Search by Shape Approximation". In *Proc. 5th Int. Symposium on Large Spatial Databases (SSD'97)*, Berlin, Germany, volume 1262 of *Lecture Notes in Computer Science (LNCS)*, pages 11–28. Springer, 1997.
- [19] J. McQueen. "Some Methods for Classification and Analysis of Multivariate Observations". In *5th Berkeley Symp. Math. Statist. Prob.*, volume 1, pages 281–297, 1967.
- [20] R. Mehrotra and J. Gary. "Feature-Based Retrieval of Similar Shapes". In *Proc. 9th Int. Conf. on Data Engineering, Vienna, Austria*, pages 108–115, 1993.
- [21] T. Möller and E. Haines. *Real-Time Rendering*. A K Peters, Natick, MA, 1999.
- [22] W. Niblack, R. Barber, W. Equitz, M. Flickner, E. Glasmann, D. Petkovic, P. Yanker, C. Faloutsos, and G. Taubin. "The QBIC Project: Querying Images by Content Using Color, Texture, and Shape". In *SPIE 1993 Int. Symposium on Electronic Imaging: Science and Technology Conference 1908, Storage and Retrieval for Image and Video Databases*, San Jose, CA, 1993.
- [23] R. Schneider, H.-P. Kriegel, B. Seeger, and S. Heep. "Geometry-based Similarity Retrieval of Rotational Parts". In *Proc. Int. Conf. on Data and Knowledge Systems for Manufacturing and Engineering, Gaithersburg, MD*, pages 150–160, 1989.
- [24] L. Vincent. "New Trends in Morphological Algorithms". In *SPIE Proceedings on Non-linear Image Processing II*, San Jose, CA, 1991.