Multimedia Tools and Applications manuscript No.
(will be inserted by the editor)

Improving 3D Similarity Search by Enhancing and Combining 3D Descriptors

Benjamin Bustos · Tobias Schreck · Michael Walter · Juan Manuel Barrios · Matthias Schaefer · Daniel Keim

Received: date / Accepted: date

Abstract Effective content-based retrieval in 3D model databases is an important problem that has attracted much research attention over the last years. Many individual methods proposed to date rely on calculating global 3D model descriptors based on image, surface, volumetric, or structural model properties. Descriptors such as these are then input for determining the degree of similarity between models. Traditionally, the ability of individual descriptors to perform effective 3D search is decided by benchmarking. However, in practice the data set on which 3D retrieval is to be applied may differ from the characteristics of the respective benchmark. Therefore, statically determining the descriptor to use based on a fixed benchmark may lead to suboptimal results.

We propose a generic strategy to improve the retrieval effectiveness in 3D retrieval systems consisting of multiple model descriptors. The specific contribution of this paper is two-fold. First, we propose to adaptively combine multiple descriptors by forming *weighted descriptor combinations*, where the weight of each descriptor is decided at query time. Second, we enhance the set of global model descriptors to be combined

Benjamin Bustos · Juan Manuel Barrios
Department of Computer Science, University of Chile
Av. Blanco Encalada 2120 3er Piso, 8370459 Santiago, Chile
Tel.: +56-2-9784969
Fax: +56-2-6895531
E-mail: bebustos,jbarrios@dcc.uchile.cl

Tobias Schreck · Michael Walter
Department of Computer Science, Technische Universitaet Darmstadt
Fraunhoferstrasse 5, D-64283 Darmstadt, Germany
Tel.: +49-6151-155125
Fax: +49-6151-155669
E-mail: tobias.schreck,michael.walter@gris.informatik.tu-darmstadt.de

Daniel Keim · Matthias Schaefer
Department of Computer Science, University of Konstanz
Universitaetsstrasse 10, D-78457 Konstanz, Germany
Tel.: +49 7531 88 3064
Fax: +49 7531 88 3065
E-mail: keim,schaefer@dbvis.inf.uni-konstanz.de

by including *partial descriptors* of the same kind in the combinations. Partial descriptors are obtained by applying a given descriptor extractor on the set of parts of a model, obtained by a simple model partitioning scheme. Thereby, more model information is exposed to the 3D descriptors, leading to a more complete object description. We give a systematic discussion of the descriptor combination space involving static and query-adaptive weighting schemes, and based on descriptors of different type and focus (model global vs. partial). The combination of both global and partial model descriptors is shown to deliver improved retrieval precision, compared to policies using single descriptors or fixed-weight combinations. The resulting scheme is generic and can accommodate a large class of global 3D model descriptors.

Keywords 3D similarity retrieval · Descriptor combinations · Object partitioning

1 Introduction

Multimedia retrieval systems have become more and more important due to the fast growing amount of available multimedia data. 3D objects are an important type of multimedia data in many application domains including Computer Aided Design/Computer Aided Manufacturing, Virtual Reality, Molecular Biology, Simulation and Engineering, Visualization, Entertainment. For example, CAD systems allow modeling of industrial designs in 3D which can be evaluated by product engineers and manufactured in subsequent steps. The wealth of designs modeled over time, possibly by many different designers, can in principle be consolidated into large 3D design databases. Given appropriate technology, from such databases it is possible to explore, market, and re-use existing designs. Retrieval by shape similarity is an important aspect of retrieving desired 3D content. This is particularly true in light of heterogeneous content and non-standardized or missing textual meta data, as often observed in practice.

To support shape-based 3D retrieval, a wealth of methods for retrieval in 3D databases has been proposed to date, see for example Tangelder and Veltkamp [34], Bustos et al. [11], and Iyer et al. [19]. Feature-based similarity search methods are among the most popular methods. The basic idea is to extract characteristic numerical attributes (so-called features, or descriptor) of the 3D objects. These descriptors are used as representations of the underlying 3D model content, and similarity between 3D objects is estimated by the mathematical differences between associated descriptors. Although feature-based methods for 3D model retrieval usually are efficient, robust, and easy to implement (see for example, Bustos et al. [10]), these descriptors typically only capture an approximation of the original 3D models. Two observations can be made: (a) there exists a large space for descriptor definition, and (b) the retrieval results obtained for any given descriptor, database, and query, may yield not only relevant, but also irrelevant results. As a consequence, a steadily increasing number of descriptors are proposed by 3D retrieval researchers, and optimization of existing descriptors by parameter tuning and forming of combinations is an important goal.

In this work, we present a generic approach for improving 3D retrieval performance for a given set of global 3D model descriptors. Assuming a pool of available 3D descriptors, we boost the retrieval precision of the given descriptors by (a) enhancing the global description with partial descriptions, and (b) by forming appropriately weighted combinations of the set of descriptors. We present algorithmic schemes for both approaches, and evaluate the effectiveness of this strategy by means of encompassing experiments.

The remainder of this paper is structured as follows. Section 2 discusses related work, after which in Section 3 our basic idea for forming combined global-partial descriptors is introduced. Section 4 then introduces our approach for deriving query-adaptive weights for formation of descriptor combinations. Section 5 evaluates the proposed approach by a series of experiments. Section 6 concludes the paper and discusses options for future work in the area.

2 Related Work

Important progress has been made toward content-based retrieval of 3D content in recent years, with a wealth of different approaches been proposed to date. Recent surveys [8, 19, 34] indicate that the *descriptor-based approach* is highly popular. Descriptors capture specific 3D object properties via compact representations (e.g., feature vectors, histograms, graphs) which, together with a distance function or other associated custom matching scheme, allow us to calculate similarity scores for any pair of 3D objects. Descriptors are usually defined heuristically, often motivated by techniques from Computer Graphics and Geometry Processing, and are often supported by Signal Processing methodology.

3D descriptors may be distinguished by the similarity concept they support. Many descriptors address global geometric similarity under invariance with respect to rigid transformations. Structural similarity notions may be supported by descriptors relying on graph representations, cf. Hilaga et al. [18] and Marini et al. [26]. Bending-invariant notion may be supported by specialized descriptors and matching strategies, cf. Ruggeri and Saupe [30].

In many applications, local similarity concepts are important. First approaches aiming to this end identify local areas of interest, and apply customized description and matching strategies to find correspondences between model parts. Various interest location detectors and matching schemes have been defined, e.g., those introduced in Gal and Cohen-Or [17] and Wessel et al. [38]. There, the main challenge is to identify which interest detector and local description concept best supports local 3D similarity concepts, and for which kinds of queries. A generic problem to solve with local interestingness detectors is to identify the scale at which interestingness is detected. To this end, scale space analysis approaches have been adapted from 2D (cf. Lowe [24]) to the 3D case (cf. Castellani et al. [12]). However, finding semantically meaningful local parts is difficult, and depending on parameterization of the analysis, either too many or too few parts may be reported.

The investigation of specific 3D object properties to exploit for descriptor definition is complemented by work to optimize the effectiveness of given descriptor types. Some of this work relies on a certain amount of supervised information about the databases or user preferences relevant for the retrieval. Research done in this area improves retrieval effectiveness by forming appropriate descriptor combinations, or optimizing certain object preprocessing steps [6, 28, 36]. Also, Machine Learning approaches are increasingly employed to further improve the performance of existing descriptor solutions, e.g., as shown by Akgul et al. [1]. An example is Wessel and Klein [37], where a 3D meta-descriptor is formed from classification probabilities of the models. Also, the *relevance feedback* approach [16] suggests to interactively collect relevance information from the user regarding the list of retrieved objects, and using this feedback to automatically optimize the query evaluation. An example work to this end is given by Leng and Qin [22].

Recently, there has appeared some related work using local features for content-based image retrieval. For example, Rahmani et al. [29] uses a salient-point based approach for performing a “localized” search, that is, only part of the image is relevant for the query. Additionally, Chen et al. [13] proposes another approach based on salient points, that tries to detect regions of interest in the image. These regions are automatically weighted by a pseudo relevance feedback method, and the resultant “weighted visual features are used for the image retrieval. Unfortunately, none of these approaches is directly applicable to 3D model retrieval, because the computation of 3D salient points is still an open problem [3].

This paper proposes a generic scheme to optimize the retrieval effectiveness of given global 3D object descriptors, by combining them with descriptors of the same descriptor type obtained for a simple partitioning of the objects. Weights for combination are found by applying a variant of the semi-supervised approach presented in Bustos et al. [6, 7]. We point out that the global-partial approach is orthogonal to using dynamic weights. For a simplified system configuration, it can well be applied using fixed weights. While reducing a bit in retrieval performance, the method then still improves over existing global-only combinations, and does not require supervised information.

3 Combination of Global and Partial Descriptors

3D descriptors supporting global shape retrieval usually represent an equal sample of local model properties in the description. This may be done by explicit enumeration of local properties (e.g., by Radial Extent Functions described as in Vranic [35]), or by aggregation of local properties sampled in a uniform way (e.g., by Shape Distributions as introduced in Osada et al. [27]). Many approaches for partial shape retrieval first determine a set of local descriptors, and then, perform a custom matching stage to establish local-to-local correspondences for pairs of models under concern, an example being the Spin Image technique introduced by Johnson and Hebert [21]. In the following, we motivate a scheme to combine both global and a simple form of partial descriptors, for support global 3D model retrieval.

3.1 Motivation and Basic Approach

Considering that many global descriptors perform some sort of averaging or sampling over local model properties, we propose to generically combine global descriptors with partial descriptors of the same type. The main motivation is that if two models are globally similar, then they should be similar regarding all of their parts. However, global descriptors, depending on their definition, may fail to capture relevant local object information. Consider e.g., the image-based approach which collects features extracted from certain model views, where the views in turn are collected by rotation of the global model. Due to this rotation, possibly important local model information can be suppressed, because it is not exposed in the projection and therefore, not available to the feature extraction. Adding descriptors for all parts of a partitioning of the model to the baseline global descriptor is expected to make the model description more complete. Specifically, as the parts are made subject to the sample particular preprocessing steps of the given descriptor, which may include normalization for scale and orientation, we expect to capture model information that would otherwise be not be available.

Our approach to use local descriptors is a heuristic. The method in nature is similar to the *Histogram of Oriented Gradients* (HOG) approach for 2D image analysis described in Dalal and Triggs [14]. In the HOG method, an input image is decomposed into a grid of subimages. Gradient descriptors are calculated for each subimage, and integrated to form a joint global image descriptor with equal weight. In our case, we combine the descriptors of all model parts with a descriptor of the whole model. Regarding the proportion by which to balance the global descriptor with the local descriptors, we generally expect to need a larger weight for the global descriptor than for the sum of local descriptors. We expect this because, the data partitioning scheme is non-data adaptive. Specifically, we expect the simple partitioning in some cases to particularly introduce additional uncertainty to the pose normalization steps, that is an integral part of many descriptor extractors. On the other hand, we assume the method to on average, contribute useful description information. As Section 5.2.1 will experimentally demonstrate, in fact moderate fixed weights for the partial descriptors amounting each to roughly 3-5% of the weight for the global descriptor, yield good results. While the optimal weight can be determined by exhaustive benchmarking or dynamic approaches (see Section 5.3), we expect using moderate weights for the local descriptors to be a feasible rule-of-thumb.

3.2 Implementation

According to Bustos et al. [8], descriptor-based 3D object comparison may be described by a multistage process. Objects to be compared are fed into a descriptor extraction pipeline, which applies method-specific object preprocessing and normalization, followed by object abstraction, analysis, and final descriptor generation. The output of this descriptor generation pipeline yields a descriptor for each input model, and descriptors are used for pairwise similarity score calculation (cf. Figure 1(left) for an illustration).

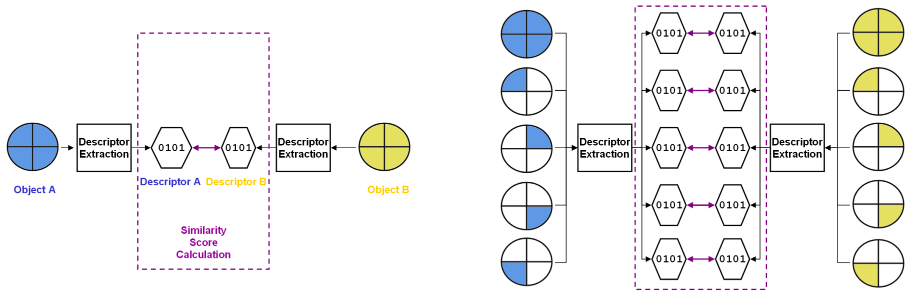


Fig. 1 Block schemes for descriptor-based similarity score calculation, for the standard global case (left), and the combined global-partial case (right).

We extend this process by a preceding simple *object partitioning step*, which yields a partitioning of each of the objects to be compared. To compare two objects, we feed the whole models as well as all of their segments into the descriptor extraction process. For each pair of objects to be compared, we first calculate similarity scores for the corresponding object and object part descriptors. As a last step, the individual scores are aggregated into one final similarity score for the input object pair (cf. Figure 1 (right) for an illustration). Our scheme relies on an existing 1:1 mapping between object segments which is used for segment-wise descriptor comparison. We secure this mapping by (A) applying rotation normalization to the global model prior to partitioning, and (B) applying a uniform, fixed partitioning scheme.

(A) is achieved by applying PCA-based rotation normalization, aligning all objects in a canonical coordinate system based on which in step (B), a simple Voronoi partitioning takes place as follows (assuming triangulated mesh models as input). We form a Voronoi partition based on $x * y$ equally spaced base vectors. The base vectors are obtained by finding x and y uniformly distributed angles θ and ϕ in spherical coordinates. Object triangles are traversed and it is checked if the respective triangle is completely contained in one Voronoi partition. If so, then the given triangle is added to that partitions segment. If not, then the triangle is split up into individual parts corresponding to the intersection of the given triangle with each of the Voronoi partitions. Finally, for each Voronoi partition all associated triangles are collected to form the respective partition segments.

This way of partitioning 3D objects into parts is admittedly simple. Many intelligent, data adaptive segmentation algorithms have been proposed to date. One example is given in Sijbers et al. [33], where a watershed-type segmentation algorithm is proposed for 3D data. Another example is given in Mademlis et al. [25], where meaningful 3D segments are found based on medial surface analysis. We point out that it is not our aim to attempt semantically meaningful object segmentation. Rather, our goal is to expose more of the object geometry to the feature extractor, thereby, arriving at a more complete object description. The simple octant-based partitioning scheme does exactly this. This structurally fixed partitioning scheme, in conjunction with rotation normalization of the objects, provides a straightforward spatial 1:1 mapping between model segments. This removes the need to find partial correspondences, which is a difficult problem in its own. Note that also, the adaptive weighting scheme described and evaluated in the following will help to identify the most promising segments to compare. From experimentation with the combination schemes as reported in Sections 4 and 5, we have observed best results regarding retrieval precision for setting $x = 2$ and $y = 4$, which effectively corresponds to an Octant partition. Throughout the remainder of this paper, we will therefore consider Octant partitioning of objects. Figure 2 illustrates the Octant partitioning scheme applied to a Formula-1 3D object.

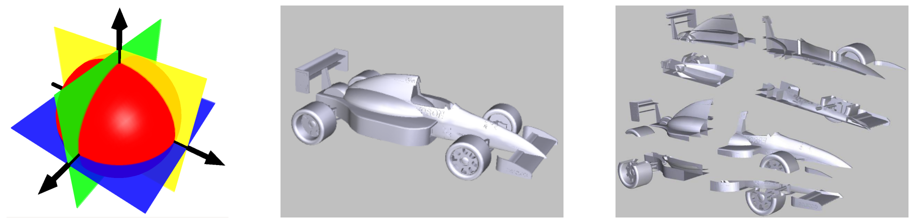


Fig. 2 Illustration of an octant-based partitioning scheme (left), and its application to a 3D object (middle and right).

4 Query-adaptive retrieval

In this section, we propose a *query-adaptive* approach for performing similarity search in 3D databases. That is, we will adapt the similarity measure used to perform the search *depending on the query object.* The hypothesis behind this is that not all feature vectors (FVs) for 3D models are well suited for different query objects. Instead, we propose methods that adaptively combine several FVs to increase the effectiveness of the 3D search engine, by taking into account the actual query object that the user wants to search.

4.1 A static approach

A retrieval performance analysis of FVs for 3D model retrieval presented in Bustos et al. [10], suggests that there are a number of FVs that achieve good average retrieval performance for 3D similarity search, but that there is no clear winner amongst them. Instead, the individual FVs have different strengths and weaknesses, and they represent complementary information regarding the description of 3D objects. From a review of the wealth of 3D descriptors proposed to date [11, 34], we can see that descriptors may differ, among other aspects, by *type of feature* and *representation of feature.* Prominent types of features include image features derived from 2D projections of the 3D models; curvature features obtained by analysis of model surface; and distribution of radial extent of the model. Often, from a given feature type, different representations can be derived. Consider for example constructing a descriptor from 2D projections. A descriptor can be obtained by considering just the contour of the image (e.g., using centroid-distances as properties), or by considering the depth image (e.g., using certain coefficients of the Fourier transform of the respective depth image).

Because FVs capture different aspects and characteristics of the models, an interesting approach is to use *combinations of FVs* for further improving the retrieval effectiveness of the similarity search, thus avoiding the disadvantages of using a single feature, capturing only a single characteristic of an object. As motivation, Figure 3 shows an example of three similarity queries (with the same query object) using two single feature vectors (depth buffer and silhouette, respectively; cf. also Section 5.1) and a combination of both feature vectors. Note that all objects retrieved by the combination of feature vectors are similar to the query object. Another example for the usefulness of combining 3D descriptors is given by the DSR descriptor introduced in Vranic [36] and also used in our study (see Section 5.1.2). The DSR descriptor combines three individual descriptors, two of them being of the same type (contour and depth image features). The combination has been shown to outperform each of the base descriptors (see [36]).

A question that naturally arises is how can different FVs be combined in a 3D similarity search system. A simple concatenation of all available FVs is not advisable because the effectiveness of the similarity search would degrade with the inclusion of FVs irrelevant to the query. Therefore, it is an interesting problem to find whether there are combinations of FVs that are better suited for performing similarity search on certain object classes, or even if there are combinations that dominate others for all types of queries.

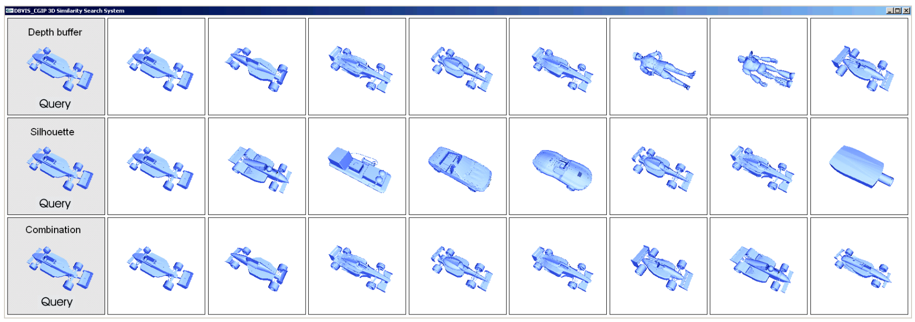


Fig. 3 Example of similarity search using a simple combination of feature vectors

A first, simple approach is to resort to static combinations of FVs. To construct the combinations, we use the sum of the unweighted normalized distances using each FV.

Definition 1 The unweighted normalized combined distance d_c is defined as:

$$\delta_c(q,o) = \sum_{i=1}^{F} b_{c_i} \frac{\delta_i(q,o)}{norm_i}$$

where F is the total number of available FVs, b_{c_i} is a binary variable that indicates whether FV f_i (i.e., the i-th considered FV) is included in combination c, δ_i is the distance function used with FV f_i, and $norm_i$ is a normalization factor for f_i.

The unweighted combination approach treats all FVs of the combination as equally important when determining the similar objects for the query object. The normalization factors are required to make the sum of distances meaningful (e.g., in case the FVs produce points of different dimensionalities). The advantage of using the *sum of distances* as combined distance is that if all δ_i distances are metrics, the sum of distances also holds the properties of a metric. This is very important if efficiency is also an issue: then, any index structure for metric spaces [31, 39] can be used to speed up similarity queries.

A throughful experimental evaluation of this static approach [4] showed that resorting to combinations of FVs is a promising approach to improve the effectiveness of similarity queries in 3D databases. However, a static combination of FVs will not necessarily provide optimal results, because if one of the considered FVs has a very bad effectiveness for the given query object, it will spoil the final result. Then, the problem is to determine which FVs to combine, as the inclusion of FVs irrelevant to q can harm the overall effectiveness of the search system.

In general, different FVs provide the best effectiveness for different query objects. Thus, the similarity search system should give a higher priority to the best FVs for q to perform the similarity query. In the next subsection, we will present a technique for the a priori estimation of the goodness of a FV given a query object.

4.2 Query-adaptive methods

To solve the problems described in the last subsection, we propose to use *adaptive weighting methods* for the combinations of FVs, which aim to give a high weight only to those FVs that are most promising to a given query object. In our proposed approach, the suitability of a FV is estimated against a training (or reference) database before performing the weighted combined query against the actual database.

4.2.1 Entropy impurity measure

We present a measure for the a priori estimation of individual FV performance, namely the *entropy impurity measure*. The values returned by this measure will be used for implementing adaptively weighted combinations of FVs.

Let $\mathbb{U}$ be the universe of valid 3D objects and $\mathbb{S} \subseteq \mathbb{U}$ a set of 3D objects. Let $\mathbb{T} \subseteq \mathbb{U}$ be a *training set of classified objects*, where $\omega_j \subseteq \mathbb{T}$, $1 \leq j \leq M$, is a *model class* of objects (i.e., all objects in ω_j are considered similar), and $\mathbb{T} = \biguplus \omega_j$ (i.e., $\mathbb{T}$ is the disjoint union of the M 3D model classes).

Let $q \in \mathbb{U}$ be a query object. Given a FV f, a *ranking* R^k_{fq} is a list of the k-NN of q from $\mathbb{T}$ with respect to f, sorted in ascending order by the distances to q. Figure 4 shows an example of a ranking for $k = 5$. Three of the k-NN belong to one of the model classes (dark grey) and the other two belong to another model class (light gray). The ranking will be the base for the proposed measure of FV performance.

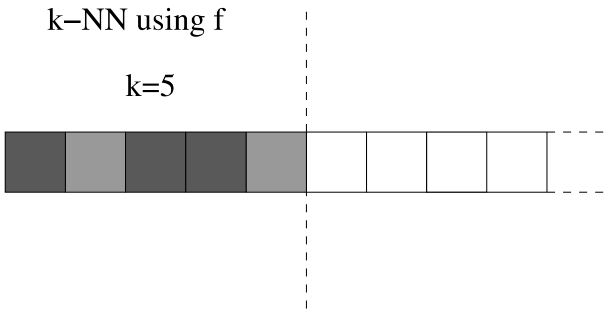


Fig. 4 Example of a ranking R^k_{fq} using $k = 5$

The *entropy impurity* [15] is a well known measure used in the context of decision tree induction. It measures the impurity of a node N of a tree with respect to the elements assigned to N. If all these elements have the same class label then the impurity is 0, otherwise it is a positive value that increases up to a maximum when all classes are equally represented[1].

Let $P^k_{\omega_j}\left(R^k_{fq}\right)$ denote the fraction of objects at the first k positions of R^k_{fq} that belong to class ω_j.

Definition 2 The entropy impurity of a FV f with respect to q is defined as

$$impurity(f,q,k) = -\sum_{j=1}^{M} \begin{cases} P^k_{\omega_j}\left(R^k_{fq}\right) \log_2\left(P^k_{\omega_j}\left(R^k_{fq}\right)\right) & \text{if } P^k_{\omega_j}\left(R^k_{fq}\right) > 0 \\ 0 & \text{otherwise.} \end{cases}$$

[1] Other impurity measures are the *Gini impurity* and the *misclassification impurity* [15], but we obtained the best experimental results by using entropy impurity.

Given a ranking R^k_{fq}, if the k objects belong to the same class then the impurity of FV f is 0; otherwise it is a positive number, with its greatest value ($\log_2 k$) occurring when the number of classes covered by the k objects in R^k_{fq} is maximal. This value aims to measure the coherence of the retrieved objects using FV f. Our hypothesis is that a well suited FV for the given query object will retrieve objects from the training set that belong to the same model class (i.e., its entropy impurity measure is low). On the other hand, if a FV retrieves objects from different model classes, then the answer is not coherent (i.e., its entropy impurity measure is high) and hence the FV is considered to be not suitable for this query object.

Considering the ranking illustrated in Figure 4, it follows that the entropy impurity of FV f is

$$\mathit{impurity}(f, q, 5) = -\left(\frac{3}{5} \cdot \log_2\left(\frac{3}{5}\right) + \frac{2}{5} \cdot \log_2\left(\frac{2}{5}\right)\right) \approx 0.29.$$

A remaining problem is how to obtain a suitable training dataset $\mathbb{T}$ for computing the entropy impurity values. For the case of 3D model retrieval, this problem may be solved by using one (or a subset) of the available standard reference collections for 3D object similarity search as $\mathbb{T}$, like the Princeton Shape Benchmark [32], the Konstanz Database Benchmark [10], or the Engineering Shape Benchmark [23] (see Section 5.1).

4.2.2 Adaptive combinations of feature vectors

The previously defined entropy impurity measure may be used to adaptively weight combinations of FVs, where the weight values depend on the query object. Let $\mathbb{F} = \{f\}$ be the set of available FVs ($|\mathbb{F}| = F$). We can use the performance estimator to combine FVs by computing an *adaptively weighted combination of FVs.* The obtained estimator value for FV f is used to compute its associated weight. After all weights are computed, the combined distance function is computed as the linear weighted combination of the distances using each FV.

As the FV performance estimation decreases with the entropy value and it has a maximum value equal to $\log_2(k)$, we use $\log_2(k) - \mathit{impurity}(\cdot)$ (which returns a value in $[0, \log_2 k]$) as weight for the combination[2].

Definition 3 The *entropy impurity weighted distance* is defined as

$$\delta_c(q, o) = \sum_{i=1}^{F} (\log_2(k) - \mathit{impurity}(f_i, q, k)) \cdot \frac{\delta_i(q, o)}{\mathit{norm}_i}.$$

The combined distance function $\delta_c(q, o)$ is then used to perform the similarity query on the database. Note that this distance function *may change on every query issued to the search system,* as it depends on the actual query object. In case efficiency is also an issue of the search system, there are index structures [5, 9] specifically designed for such adaptive distance functions (also known as *multi-metric* distances).

[2] The original formula introduced in Bustos et al. [6] returned values in the range $(0, 1]$. The new formula presented here ensures that the weight is 0 if the entropy impurity value reaches its maximum.

4.3 Adaptive approach with global-partial features

The previously introduced entropy impurity measure may also be used in conjunction with global-partial features. In this case, the entropy impurity approach estimates the a priori goodness of each of the partial features. This makes sense because it is possible that some of the partial fragments of an objects do carry noise, thus it would be beneficial for the retrieval system to avoid (i.e., give low weights) to those noisy partial features.

For our experiments, we used the same training datasets as for the query-adaptive approach. It must be noticed, however, that the weights given by the entropy impurity method must be scaled down in order to not overweight the partial features. This is necessary to ensure that the global feature has always more impact on the retrieval of similar 3D objects than the partial features.

Finally, it is also possible to mix the adaptive approach with global-partial features and the adaptive combination of different FVs. In this case, weights are computed for each partial feature within each FV and for each FV represented by its global-partial features. In the experimental evaluation, we will show that this final approach allows us to obtain the best overall effectiveness results compared against all single, global-partial, and query-adaptive approaches when used as standalone techniques.

5 Experimental Evaluation

To assess the impact of the developed global-partial combination scheme on retrieval precision, a series of experiments was performed and is presented as follows. Section 5.1 introduces the used descriptors and benchmark data sets, recalling baseline benchmark results. Section 5.2 reports benchmarking results of statically weighted global-partial combinations. Section 5.3 then reports results of adaptively weighted global-partial combinations. Section 5.4 gives an assessment of the results obtained by the different combination strategies, and discusses further options for combination building.

5.1 Preliminaries

5.1.1 Benchmarks, 3D Descriptors, and Experimental Platform

Our experiments are based on two 3D benchmark data sets. The *Konstanz 3D Benchmark* (KNDB) is introduced in Bustos et al. [10] and consists of 1,838 3D mesh models out of which 472 are classified into 55 equivalence classes, serving as queries. The benchmark encompasses generic 3D objects representing real-world objects like humans, vehicles, furniture, etc. By its structure and observed benchmarking results, KNDB shares many similarities with the Princeton Shape Benchmark as described in Shilane et al. [32]; for details please cf. Bustos et al. [10]. The *Purdue Engineering Shape Benchmark* (ESB) is introduced in Jayanti et al. [20] and consists of 866 3D objects representing engineering shapes. We rely on the lowest level of the ESB classification hierarchy, classifying the shape models into 45 equivalence classes. As 3D descriptors, we consider the following three standard 3D descriptors:

- Silhouette images (SIL, 300dim): A descriptor derived from orthogonal projections of the 3D models along the three principal axes of the respective models. The descriptor is obtained from the centroid-distances of the silhouette images.
- Depth images (DBF, 438dim): A descriptor derived from depth images obtained by orthogonal projection of the models along their three principal axes.
- Radial extension function in Spherical Harmonics representation (RSH, 136dim): The radial extent of the model is measured by a uniform sampling scheme, and represented by coefficients of Spherical Harmonics base functions.
- We also consider a hybrid descriptor formed from these descriptors (DSR, 472dim). DSR statically combines the SIL, DBF, and RSH descriptors of fixed dimensionality by vector concatenation.

For implementation details of SIL, DBF, and RSH, please refer to Vranic [35]. DSR is described in Vranic [36]. As distance function, we use the *Manhattan distance* ($\delta(x, y) = \sum_{i=1}^{dim} |x_i - y_i|$) in all our experiments. This distance is a metric, it has been shown to provide good effectiveness in 3D model retrieval [10] and it is cheap to compute.

We performed our experiments on a standard development environment. Specifically, a Windows XP system with Dual Core CPU and 2 GB of main memory was used. The 3D descriptor extractors were implemented in C++. The combination schemes and evaluation precision-recall calculation schemes were implemented in JAVA. Given the size of the 3D object database, number of descriptors, and model partitioning scheme, all experiment data could be held in main memory.

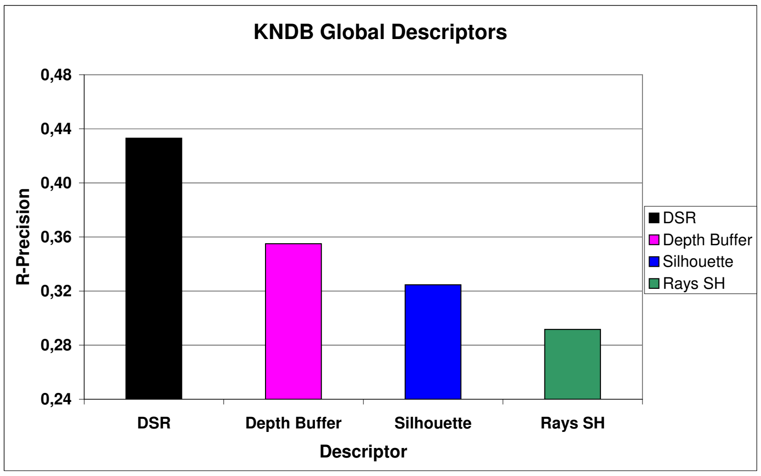
5.1.2 Baseline Benchmark Results

As a first experiment, we compare the retrieval precision performance of the selected descriptors. This results will be used to evaluate the improvements on effectiveness of the proposed approaches based on combinations. For comparing the effectiveness of the different approaches, we use the *R-Precision score* [2], which returns a single value thus making easy to compare different retrieval systems. Given a query object, the R-precision is defined as the precision (number of relevant objects p retrieved by a search divided by the total number of objects retrieved by the search) after retrieving R objects, where R is the number of relevant objects in the collection for the query:

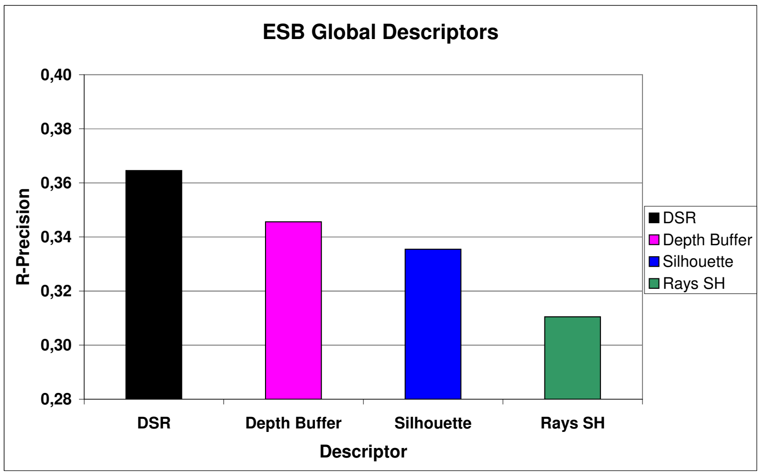
$$\text{R-Precision} = \frac{p}{R}$$

The R-Precision is a single score in the range $[0, 1]$. A search system with a high R-precision value (close to one) means that it has good effectiveness. According to our experience, this score is highly correlated to other retrieval benchmarking schemes. We computed the R-precision scores for all query objects of a given benchmark and descriptor, and then computed the average R-precision value.

Figure 5 plots the average R-Precision scores observed for the DSR, DBF, SIL, and RSH descriptors, on the KNDB and ESB benchmark data sets, respectively. Consistent with previously reported results, Depth Buffer image-based descriptor performs generally well and substantially better than descriptors based on Silhouette images or Radial Extent Functions. DSR, as a static combination of DBF, SIL and RSH, improves substantially over the best performing individual descriptor DBF. According to Vranic [36], the retrieval precision provided by DSR is among the best one can currently expect from any (static) 3D object descriptor. All relative improvements on the effectiveness of our proposed methods (presented in the next subsections) are computed with respect to the results presented in Figure 5.



(a) KNDB Results



(b) ESB Results

Fig. 5 Baseline R-Precision scores of DSR, DBF, SIL, and RSH descriptors, observed on the KNDB and ESB benchmarks.

5.2 Statically Weighted Combinations

We first assess the effect of statically combining global and partial descriptors. Section 5.2.1 gives the setup for forming combinations, and Sections 5.2.2 and 5.2.3 report the result of using global-partial combinations of single and multiple descriptor types, respectively.

5.2.1 Experiment Setup

In accordance with the aggregation model from Section 4.1, we determine the aggregated distance between any two 3D objects as a sum of normalized, weighted distances. The individual distances in this sum are calculated from a set of available descriptors, each descriptor being of a given *descriptor type* and *model focus*. We use as descriptor types DSR, DBF, SIL, and RSH descriptors (cf. Section 5.1). As descriptor focus, we consider full 3D models, and model segments obtained by smoothly segmenting the models into eight parts each, according to Section 3.2. In this subsection, we first apply static weighting for forming the combinations, while in Section 5.3, we will focus on adaptively weighted combinations. Figure 6 summarizes the space of possible descriptor combinations addressed in this paper.

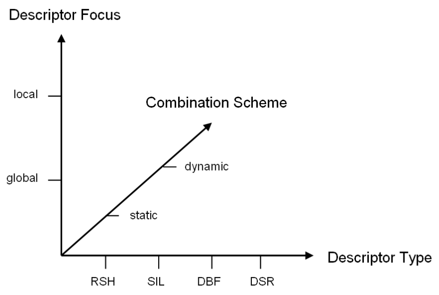


Fig. 6 Experiment space spanned by combining descriptors of different types and focus either statically or adaptively.

For a given descriptor type, 9 individual descriptors (one global and eight partial) contribute to the aggregate distance calculation for each pair of objects. As we like to control the influence of the global and the partial descriptors toward the final combination, we introduce a (static) weight vector $\mathbf{w}_i$ of the following form:

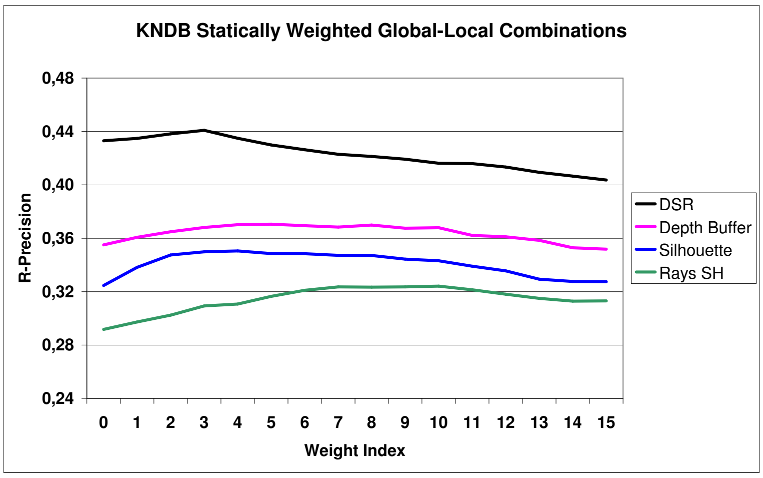
$$\mathbf{w}_i = <1, \frac{i}{100}, \frac{i}{100}, \frac{i}{100}, \frac{i}{100}, \frac{i}{100}, \frac{i}{100}, \frac{i}{100}, \frac{i}{100}> \qquad (1)$$

We consider $\mathbf{w}_i[0]$ to be the weight for the global descriptor, and $\mathbf{w}_i[1], \ldots, \mathbf{w}_i[8]$ to be the weights for the partial descriptors. Note that $\mathbf{w}_0$ corresponds to using the global descriptor only. Weight vectors of index $i > 0$ correspond to combinations of both global and partial descriptors, where each partial descriptor is weighted by $i\%$ relative to the global descriptor.

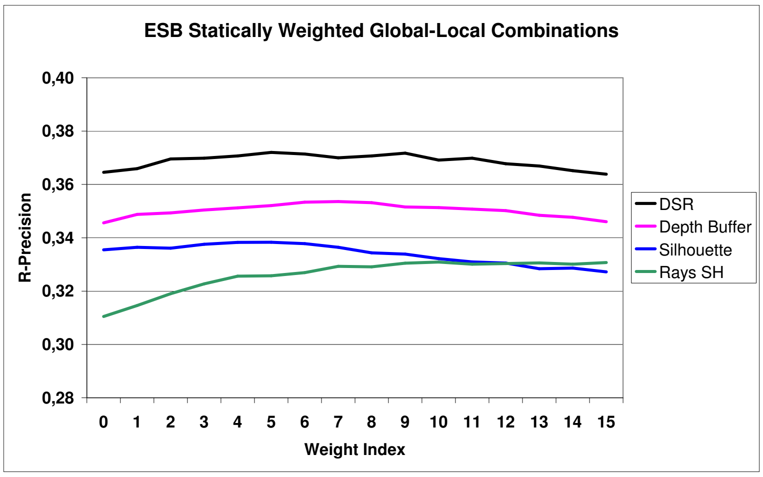
We tested weight vectors for $i = 1, \ldots, 15$ modeling global-partial combinations where the partial descriptors receive low to moderate weighting, relative to the global descriptor. For each weight vector, we evaluate the average R-Precision over all query objects from the KNDB and ESB benchmarks, respectively.

5.2.2 Static Global-Partial Combinations for Individual Descriptor Types

As a baseline experiment, we consider extending global descriptors by descriptors of segments of the models. Figure 7 shows the retrieval precision scores obtained on the KNDB and ESB benchmarks, for combined global and partial 3D descriptors, for each of the base descriptors DSR, DBF, SIL, and RSH. Specifically, the R-precision for increasing partial descriptor weights according to Formula 1 is shown. Table 1 compares the R-Precision values of combinations obtained for the best weight settings with the respective results of using only the global descriptors (i.e., a weight of 0 for the local descriptors, cf. also Section 5.1).



(a) KNDB Results



(b) ESB Results

Fig. 7 R-Precision results for statically weighted combinations of global and partial features.

From these data, we make the following observations. First, for all descriptor types and benchmarks tested, there are combinations of global and local descriptors that *improve* the retrieval precision over using only the base descriptor. This is in accordance with the assumption that combinations of global and partial descriptors can deliver more useful model information (cf. Section 3.1). Second, we observe that while all

Table 1 Results of using best fixed weights (indices given in brackets) for global-partial combinations, and comparison with baseline results of using only the respective global descriptor.

	DSR	Depth Buffer	Silhouette	Rays SH
KNDB (global)	43.31%	35.50%	32.46%	29.17%
KNDB (best i)	44.09%(3)	37.05%(5)	35.05%(4)	32.41%(10)
KNDB (relative improvement)	1.81%	4.37%	7.97%	11.12%
ESB (global)	36.46%	34.56%	33.55%	31.05%
ESB (best i)	37.20 %(5)	35.36%(7)	33.83%(5)	33.09%(10)
ESB (relative improvement)	2.05%	2.31%	0.86%	6.59%

tested descriptor types benefit from this combination scheme, the benefits are higher for descriptor types that show a lower base performance. E.g., RSH as the weakest performing base descriptor is improved by up to 10%, while DSR as the best performing base descriptor is improved up to 2%. This is in accordance with the general *Pareto* principle (increasing marginal costs). Third, we observe that while the beneficial weight intervals are generally low for all descriptor types, the optimal weights are different for each descriptor type (cf. Table 1). For some cases (DSR and DBF in KNDB benchmark, and SIL in ESB benchmark), in increasing local weights we observe performance results even dropping below the base performance.

We conclude that the base combination scheme is in principle effective. Given its simplicity, it can be recommended as a default tuning method to (moderately) boost the retrieval performance of a given 3D retrieval system. Some care needs to be applied in setting the weights for the local descriptors, as too high weights may be ineffective.

We note that the rates of improvement are not tremendously high, in particular for the descriptors with better base performance. Also, there is no common weight setting that would provide optimal improvement for all descriptor types and benchmarks. Therefore, we next consider a scheme for adaptively finding weights for global and local descriptors, with the aim of further improved retrieval performance.

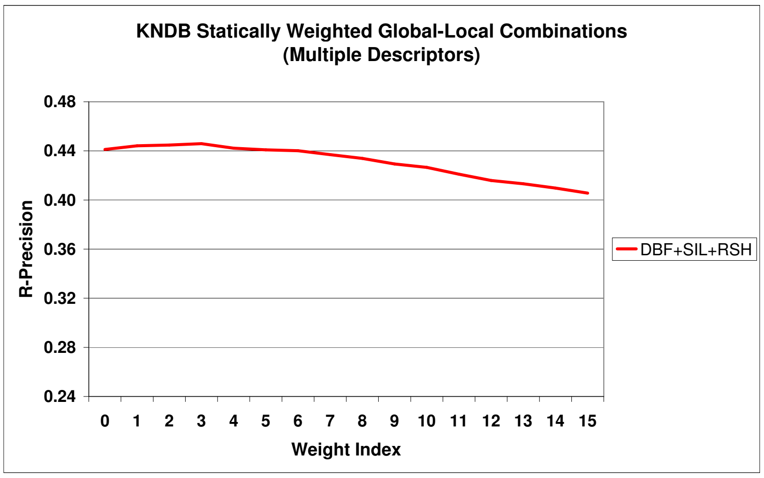
5.2.3 Static Global-Partial Combinations for Multiple Descriptor Types

Our previous studies [6, 7] have shown that the combination of (global) 3D descriptors of different types may improve retrieval precision rates. Specifically, the DSR descriptor (see Vranic [36]), as a static combination of different 3D descriptors, has been shown to provide improved precision rates (cf. Section 5.1.2). We are therefore interested whether the idea of extending a descriptor by global descriptors of the same type also hold across combinations of several descriptor types.

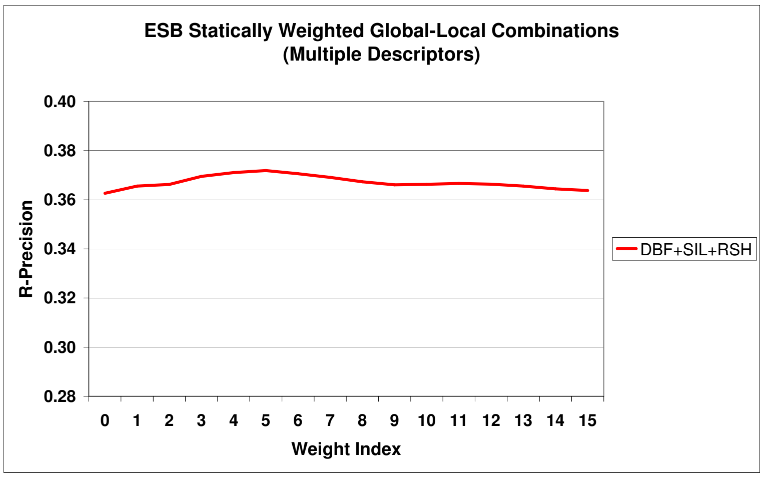
A first indication, that our approach also applies to combinations was already given in Section 5.2.2: The extension of DSR by partial DSR descriptors improved up to 2% in R-Precision, on the ESB and KNDB data sets. To complement this finding, we performed experiments combining the set of DBF, SIL, and RSH descriptors with partial descriptors of the same types. The overall combination was formed by summing up the 27 distance values, given by one gobal and 8 partial descriptors, for each of the three descriptor types (the same static weight vector was used for each feature type).

Figure 8 gives the plot of the resulting R-Precision scores for $i = 0, \ldots, 15$. For both data sets, the global-partial combinations of DBF, SIL, and RSH yield improved R-Precision scores for rather small values for i. R-Precision for KNDB peaks at $i = 3$ with

44.6%, while it does so for ESB at $i = 5$ with 37.19%. Compared to the R-Precision values at $i = 0$, which correspond to the global-only combination of the three desriptor types, this implies an improvement of ca. 1.0% and 2.5% (cf. also Table 2). The order of magnitude of the improvement is comparable to the improvement observed for the globa-partial combinations of DSR alone, which is a vector combination of DBF, SIL, and RSH descriptors.



(a) KNDB Results



(b) ESB Results

Fig. 8 R-Precision results for statically weighted combinations of global and partial features of DBF, SIL, and RSH descriptors.

Table 2 Results of using best fix-weighted global-partial combination of DBF, SIL, and RSH descriptors.

	KNDB	ESB
DBF+SIL+RSH (global)	44.12%	36.27%
DBF+SIL+RSH (best i)	44.59% (3)	37.19% (5)
Relative improvement	1.07%	2.55%

5.3 Adaptively Weighted Combinations

The previous experiments indicate that our approach is basically applicable. However, considering the heuristic nature (using static weight vectors) and simplicity (model partitioning scheme) of the approach, we expect that the improvement potential has not been fully utilized yet. We expect that an appropriate adaptive weighting scheme should be beneficial to this end, as not every (partial or global) descriptor of every type will be equally important for answering every possible 3D query (cf. also Section 4.3).

To test this idea, we applied the weighting scheme introduced in Section 4, to modify the weight used with each descriptor, depending on the given query object. We next present experimental results we obtained. Section 5.3.1 introduces the modified experiment setup, and Sections 5.3.2 and 5.3.3 report results obtained for global-partial combinations for individual descriptors as well as for sets of descriptors.

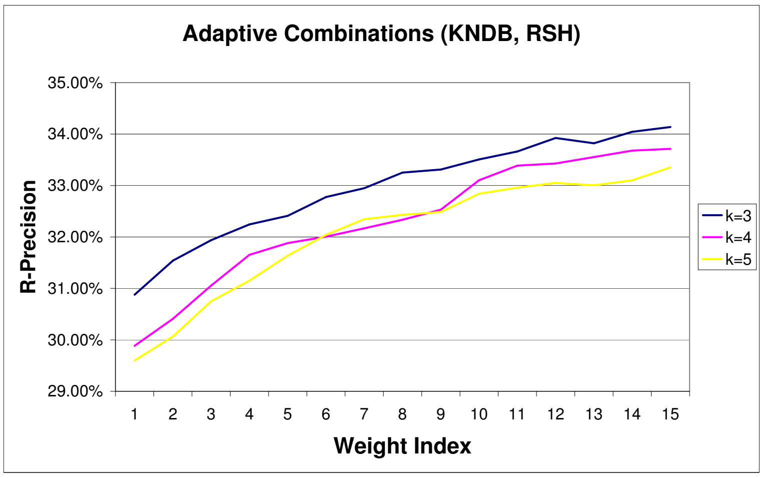
5.3.1 Experiment Setup Modification

The experiment setup from Section 5.2.1 is extended by scaling the static weight vector by adaptive components corresponding to the Entropy Impurity scores calculated according to Section 4.2. Each adaptive weight is an estimation for the suitability of the descriptor for retrieving objects similar to a given query. Specifically, the benchmark data set classification was used as the ground truth[3]. The Entropy Impurity values were computed beforehand and stored in a lookup table. At query time, the basic weight vector $\mathbf{w}$ is modified by multiplying each of its component by the respective Entropy Impurity value of the whole model as well as all its segments, obtaining the final query-adaptive weight vector.

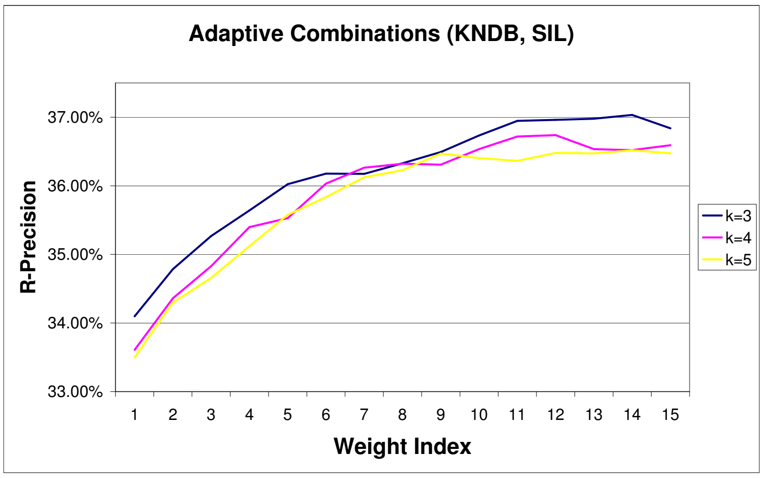
5.3.2 Adaptive Global-Partial Combinations for Individual Descriptor Types

We evaluated the R-Precision scores for global-partial combinations formed for each descriptor type, using adaptive weight vectors obtained for different prefix length parameters k (cf. Section 4.2). Figures 9 and 10 shows the results for the different descriptor types on the KNDB benchmark, for $k = \{3, 4, 5\}$ and $i = \{0, \ldots, 15\}$. We observe that for given k, R-Precision scores increase consistently over the range of base weights. In accordance with the observations made in Bustos et al. [7], we also observe that lower values for k usually yield the better results.

[3] For the partial descriptors, we let the object segments inherit the class label from their corresponding full model. Entropy Impurity was calculated for each fragment object among all other object fragments from the same spatial index.



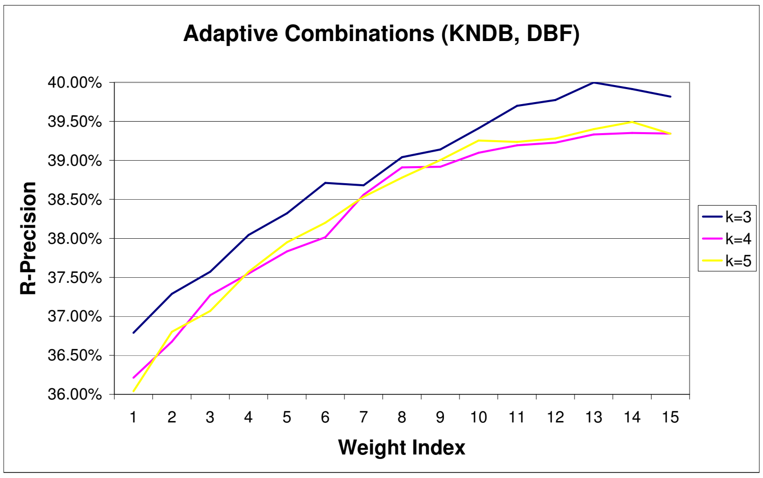
(a) RSH



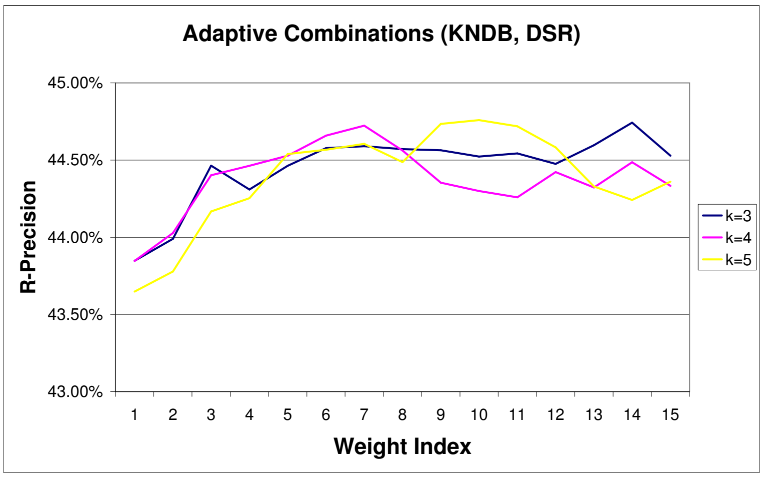
(b) SIL

Fig. 9 R-Precision results for adaptively weighted combinations of global and partial descriptors of type RSH and SIL, obtained on the KNDB benchmark.

Table 3 lists the best R-Precision scores observed for the four descriptor types on the KNDB and ESB benchmarks, and compares them in terms of relative improvement with the global-only results. The general trend is that the comparable weaker descriptors benefit the most from the extension, similar to the observation made in Section 5.2.2 for the statically weighted case. The best R-Precision values observed for the adaptively weighted combinations improve between 3% and 17% over the global-only results, and are consistely better than the statically weighted global-partial combinations.



(a) DBF



(b) DSR

Fig. 10 R-Precision results for adaptively weighted combinations of global and partial descriptors of type DBF and DSR, obtained on the KNDB benchmark.

5.3.3 Adaptive Global-Partial Combinations for Multiple Descriptor Types

Finally, we evaluated the adaptively weighted global-partial combination of the set of descriptor types RSH, SIL, and DBF. Figure 11 shows the results obtained on the KNDB and ESB benchmarks, respectively. For both benchmarks, maximum R-Precision values are reached at rather high base weights for the partial descriptors at or above $i = 10\%$. For KNDB, the combinations perform best at $k = 3$, while for ESB, adaptive weights obtained for $k = 1, 2, 3$ perform rather equally well. Table 4 compares the R-Precision scores obtained for the best weighted combination of global and partial descriptors with the statically-weighted, global-only descriptor combination. Relative improvement amounts to ca. 8% and 6.5% on KNDB and ESB, respectively.

Table 3 Results of using best adaptive weights (parameters given in brackets) for global-partial combinations, and comparison with baseline results of using only the respective global descriptor.

	DSR	Depth Buffer	Silhouette	Rays SH
KNDB (global)	43.31%	35.50%	32.46%	29.17%
KNDB (best i,k)	44.75%(10,5)	40.00%(13,3)	37.03%(13,4)	34.14%(15,3)
KNDB (rel. improvement)	3.35%	12.67%	14.07%	17.03%
ESB (global)	36.46%	34.56%	33.55%	31.05%
ESB (best i,k)	38.02%(10,4)	35.97%(9,4)	34.76%(10,5)	34.13%(15,5)
ESB (rel. improvement)	4.29%	4.07%	3.60%	9.93%

Table 4 Results of using best adaptively weighted global-partial combination of DBF, SIL, and RSH descriptors.

	KNDB	ESB
DBF+SIL+RSH (global)	44.12%	36.27%
DBF+SIL+RSH (best i,k)	47.93% (10,3)	38.64% (13,5)
Relative improvement	8.63%	6.55%

5.4 Summary and Discussion

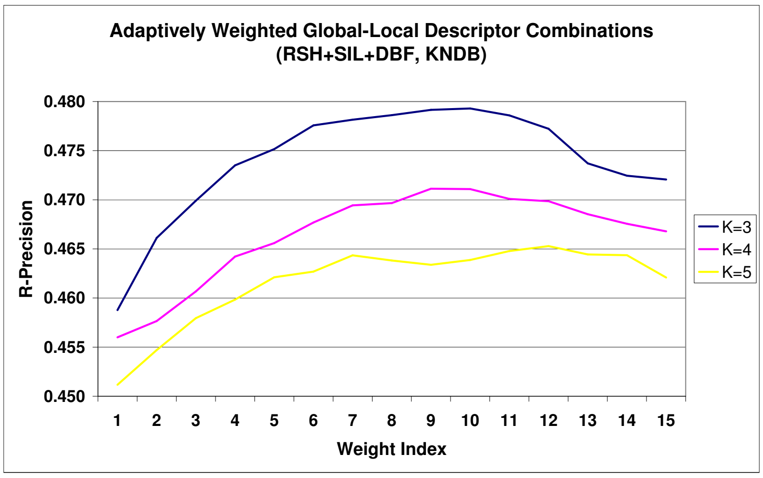
We summarize and discuss the main experimental findings, as well as further considerations.

5.4.1 Summary of the Results

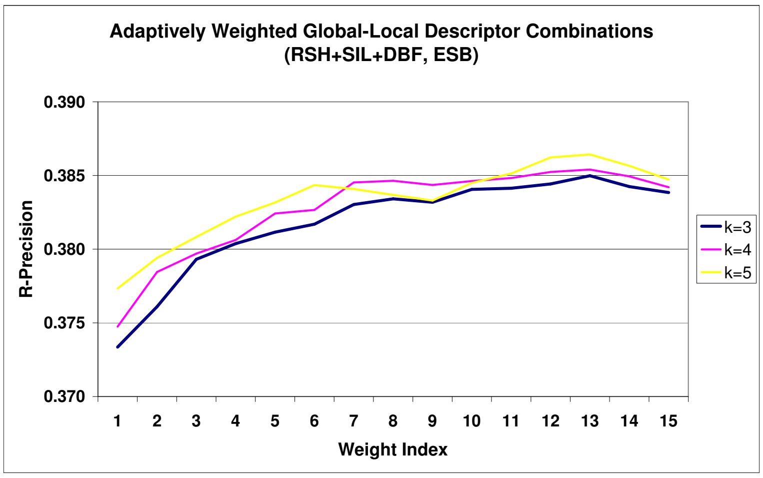
We summarize the results of the static combination experiments as follows.

– We find that *statically combining single global descriptors by partial descriptors of the same type* is a viable approach to improve retrieval precision. Improvements were observed for different descriptor types (RSH, SIL, DBF), and on different benchmarks (generic and engineering 3D shapes). Improvements were observed for combining the global descriptor with eight partial descriptors, each receiving moderate weight, relative to the global descriptor counterpart. Depending on the base retrieval precision, improvements between 2% (Depth Buffer on ESB benchmark) and 11% (Radial Extent Function on KNDB benchmark) were observed.
– Improvement rates of up to 2% were also observed for *statically combining multiple descriptors by partial descriptors*, both for a descriptor representing a fixed vector combination (DSR), as well as combining the RSH, SIL, and DBF descriptors.

We conclude that statically weighted global-partial combinations may serve as a generic approach to improve the retrieval performance of a given descriptor. The approach is inexpensive in the sense that it can be applied without extending a set of existing descriptors, but only a simple model partitioning scheme needs to be applied, the output of which is fed into the existing descriptor extraction pipeline. It is interesting to note that while the observable improvement rates depend on the base performance of the descriptors, we observed still notable improvements for one of the very best performing global static descriptor combinations (DSR).



(a) KNDB Results



(b) ESB Results

Fig. 11 R-Precision results for adaptively weighted combinations of global and partial descriptors of type RSH, SIL, DBF, and DSR in a single combination, obtained on the KNDB benchmark.

Results of the adaptive combination experiments are summarized as follows.

- Application of the Entropy Impurity based descriptor weighting scheme to global-partial combinations of given descriptor type improves one more. Relative improvement rates between 4% (Depth Buffer on ESB benchmark) and 17% (Radial Extent Function on KNDB benchmark) were observed.
- For the DSR static vector combination, adaptive global-partial combinations improved by 3.4% and 4.3% over the global-only DSR descriptor on KNDB and ESB benchmarks, respectively. The best overall retrieval precision rates were observed for adaptively weighted global-partial combinations of the three descriptor types RSH, SIL, and DBF together. Here, the relative improvement compared to the

global-only combination amounted to 8.6% and 6.5% on KNDB and ESB, respectively.

From the result summary, we conclude that the Entropy Impurity-based adaptive combination scheme is effective in appropriately weighing descriptors of different types and model focus to provide substantially improved retrieval precision results. Specifically, the simultaneous adaptive weighting depending on descriptor type *and* descriptor focus (in the case RSH+SIL+DBF) is beneficial. This is seen as the former combination yields better results than adaptively combining global and partial DSR descriptors alone, where the descriptor by definition fixes the weight of each descriptor type contained in the fixed vector combination.

5.4.2 Discussion

An assessment of these results needs to consider the *magnitude* of the obtained improvements and the *cost* of realizing these improvements. We also discuss the applicability of the approach to different classes of 3D descriptors.

Regarding the magnitude aspect, we state that the question of improvement is a relative one, as it depends on the base against which the improvements are achieved. We believe that a fair, yet tough baseline for assessment is the DSR descriptor in its standard, global implementation, providing state-of-the-art retrieval precision. A comparison of the best obtained results with the DSR performance is given in Table 5. According to this table, the best adaptive combination outperforms the standard DSR descriptor by 10% and 6% on the KNDB and ESB benchmark datasets. With regard to the Pareto Principle, we argue that these improvement rates are considerable. Note that if we consider not DSR (which is already a combination) but e.g., the DBF descriptor (one of the best performing single descriptor types) as a baseline, even larger improvement rates (up to 35% on KNDB) would be observed.

Table 5 Summary table listing the best observed R-Precision scores of the various single and combined descriptors, including comparison with the DSR descriptor. * and ** denote the weighted combinations of best parameter settings for i and (i, k), respectively.

Method	RP. KNDB	%DSR	RP. ESB	%DSR
RSH	29.17%	-32.65%	31.05%	-14.84%
SIL	32.46%	-25.04%	33.55%	-7.98%
DBF	35.50%	-18.03%	34.56%	-5.20%
DSR	**43.31%**	**0.00%**	**36.46%**	**0.00%**
DSR*	44.09%	1.81%	37.20%	2.05%
(DBF+SIL+RSH)*	44.59%	2.97%	37.19%	2.01%
DSR**	44.76%	3.35%	38.02%	4.29%
(DBF+SIL+RSH)**	47.93%	10.67%	38.64%	5.99%

Regarding the cost aspect, we distinguish between static and adaptive combinations. We consider the static case relatively inexpensive, as it can be based on most existing 3D descriptor extraction frameworks. It requires a simple model partitioning, and extra time for descriptor extraction (typically done off-line) and distance aggregation (on-line to query time). Regarding the adaptive combination case, the main cost factor involves the provision of a suitable classification used for Entropy Impurity weight calculation. Depending on the type of retrieval system and observation model, these costs could be either low or high. E.g., if class labels can be accumulated from user Relevance Feedback over time, these costs might be low. They might be high, if class labels need to be obtained in any other way, e.g., manually, or by applying machine learning algorithms. In any case, our framework supports both operation modes, and the best mode may be determined on an application basis.

Our approach works for any 3D descriptor extraction algorithm that is able to extract features from a model part. Example descriptor classes include radial extent, image, and curvature-distribution feature extractors. As the partitioning approach usually returns non-watertight model segments, methods requiring water-tight models are not directly applicable. However, post-processing could applied to the mesh partitions, securing water-tightness. In principle, the method could also be applied to graph-based descriptors. However, many graph-based descriptors analyze whole models. Also, it is not straightforward how to form combinations of partial graph representations. Therefore, more research is needed to assess the applicability of our approach with respect to graph-based descriptors.

6 Conclusions and Future Work

We presented a generic approach to improve retrieval precision in 3D model retrieval systems consisting of a set of model descriptors. The approach is based on combining descriptors of different object focus (specifically: global and partial 3D object focus) to improve retrieval precision. Additionally, the scheme accommodates the combination of descriptors of different types, and supports statically or adaptively weighted combinations. An implementation consisting of a simple 3D object partitioning scheme and an adaptive weight determination scheme based on Entropy Impurity was proposed. Based on extensive experiments, the approach was shown to outperform one of the very best 3D descriptors, as evaluated on two different 3D benchmarks.

These results are promising, and lead us to consider the following future research. A straightforward step is to research additional object partitioning schemes. Currently, adaptive weighting is employed to over or underweight segment descriptors obtained by a fixed object partitioning scheme. More intelligent partitioning algorithms would be expected to yield better partitioning results. Further, our distance calculation compares partial descriptors for fixed spatial positions of the segments, relative to the global object. More flexible matching approaches can be expected to deliver improved results, including support for partial similarity notions.

Finally, many other approaches for determining suitable weights for forming global-partial descriptor combinations are possible. Specifically, methods not requiring supervised information should be developed, lowering the costs for realizing improved retrieval precision. Again, more intelligent partitioning algorithms could be expected to also yield estimations on the meaningfulness of the segments produced, which in turn could serve as weights for forming descriptor combinations.

Acknowledgment

We thank Dejan Vranić and Dietmar Saupe for providing the base 3D descriptor extractors used in our work.

References

1. Akgül CB, Sankur B, Yemez Y, Schmitt F (2008) Similarity score fusion by ranking risk minimization for 3D object retrieval. In: Proc. Eurographics Workshop on 3D Object Retrieval (3DOR'08), Eurographics Association, pp 41–48
2. Baeza-Yates R, Ribeiro-Neto B (1999) Modern Information Retrieval. Addison-Wesley
3. Bronstein A, Bronstein M, Bustos B, Castellani U, Crisani M, Falcidieno B, Guibas L, Kokkinos I, Murino V, Sipiran I, Ovsjanikovy M, Patane G, Spagnuolo M, Sun J (2010) Shrec 2010: robust feature detection and description benchmark. In: Proc. Eurographics Workshop on 3D Object Retrieval (3DOR'10), Eurographics Association, pp 79–86
4. Bustos B (2006) Index structures for similarity search in multimedia databases. PhD thesis, Department of Computer and Information Science, University of Konstanz
5. Bustos B, Skopal T (2006) Dynamic similarity search in multi-metric spaces. In: Proc. 8th ACM SIGMM International Workshop on Multimedia Information Retrieval (MIR'06), ACM Press, pp 137–146
6. Bustos B, Keim D, Saupe D, Schreck T, Vranić D (2004) Automatic selection and combination of descriptors for effective 3D similarity search. In: Proc. IEEE 6th International Symposium on Multimedia Software Engineering (ISMSE'04), IEEE Computer Society, pp 514–521
7. Bustos B, Keim D, Saupe D, Schreck T, Vranić D (2004) Using entropy impurity for improved 3D object similarity search. In: Proc. IEEE International Conference on Multimedia and Expo (ICME'04), IEEE, pp 1303–1306
8. Bustos B, Keim D, Saupe D, Schreck T, Vranić D (2005) Feature-based similarity search in 3D object databases. ACM Computing Surveys 37(4):345–387
9. Bustos B, Keim D, Schreck T (2005) A pivot-based index structure for combination of feature vectors. In: Proc. 20th Annual ACM Symposium on Applied Computing, Multimedia and Visualization Track (SAC-MV'05), ACM Press, pp 1180–1184
10. Bustos B, Keim D, Saupe D, Schreck T, Vranić D (2006) An experimental effectiveness comparison of methods for 3D similarity search. International Journal on Digital Libraries, Special issue on Multimedia Contents and Management in Digital Libraries 6(1):39–54
11. Bustos B, Keim D, Saupe D, Schreck T (2007) Content-based 3D object retrieval. IEEE Computer Graphics and Applications, Special Issue on 3D Documents 27 (4):22–27
12. Castellani U, Cristani M, Fantoni S, Murino V (2008) Sparse points matching by combining 3D mesh saliency with statistical descriptors. Computer Graphics Forum 27(2):643–652
13. Chen J, Ma R, Su Z (2010) Weighting visual features with pseudo relevance feedback for CBIR. In: Proc. 9th ACM International Conference on Image and Video Retrieval (CIVR'10), ACM, pp 220–227
14. Dalal N, Triggs B (2005) Histograms of oriented gradients for human detection. In: CVPR '05: Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05) - Volume 1, IEEE Computer Society, Washington, DC, USA, pp 886–893, DOI http://dx.doi.org/10.1109/CVPR.2005.177

15. Duda R, Hart P, Stork D (2001) Pattern Classification, 2nd edn. Wiley-Interscience, New York
16. Frakes W, Baeza-Yates RA (1992) Information Retrieval:Data Structures & Algorithms. Prentice-Hall
17. Gal R, Cohen-Or D (2006) Salient geometric features for partial shape matching and similarity. ACM Transactions on Graphics 25(1):130–150
18. Hilaga M, Shinagawa Y, Kohmura T, Kunii T (2001) Topology matching for fully automatic similarity estimation of 3D shapes. In: Proc. ACM International Conference on Computer Graphics and Interactive Techniques (SIGGRAPH'01), ACM Press, pp 203–212
19. Iyer N, Jayanti S, Lou K, Kalyanaraman Y, Ramani K (2005) Three-dimensional shape searching: state-of-the-art review and future trends. Computer-Aided Design 37:509–530
20. Jayanti S, Kalyanaraman Y, Iyer N, Ramani K (2006) Developing an engineering shape benchmark for cad models. Computer-Aided Design 38(9):939–953
21. Johnson A, Hebert M (1999) Using spin images for efficient object recognition in cluttered 3D scenes. IEEE Transactions on Pattern Analysis and Machine Intelligence 21(5):433–449
22. Leng B, Qin Z (2008) A powerful relevance feedback mechanism for content-based 3d model retrieval. Multimedia Tools and Applications 40(1):135–150
23. Lou K, Prabhakar S, Ramani K (2004) Content-based three-dimensional engineering shape search. In: Proc. International Conference on Data Engineering (ICDE'04), IEEE Computer Society, pp 754–765
24. Lowe DG (2004) Distinctive image features from scale-invariant keypoints. International Journal on Computer Vision 60(2):91–110
25. Mademlis A, Daras P, Tzovaras D, Strintzis MG (2007) On 3D partial matching of meaningful parts. In: Proc. IEEE International Conference on Image Processing (ICIP'07), IEEE, vol 2, pp 517–520
26. Marini S, Spagnuolo M, Falcidieno B (2007) Structural shape prototypes for the automatic classification of 3D objects. Computer Graphics and Applications 27(4):28–37
27. Osada R, Funkhouser T, Chazelle B, Dobkin D (2002) Shape distributions. ACM Transactions on Graphics 21(4):807–832
28. Papadakis P, Pratikakis I, Theoharis T, Passalis G, Perantonis S (2008) 3D object retrieval using an efficient and compact hybrid shape descriptor. In: Proc. Eurographics Workshop on 3D Object Retrieval (3DOR'08), Eurographics Association, pp 9–16
29. Rahmani R, Goldman SA, Zhang H, Cholleti SR, Fritts JE (2008) Localized content-based image retrieval. IEEE Transactions on Pattern Analysis and Machine Intelligence 30:1902–1912
30. Ruggeri M, Saupe D (2008) Isometry-invariant matching of point set surfaces. In: Proc. Eurographics Workshop on 3D Object Retrieval (3DOR'08), Eurographics Association, pp 17–24
31. Samet H (2005) Foundations of Multidimensional and Metric Data Structures (The Morgan Kaufmann Series in Computer Graphics and Geometric Modeling). Morgan Kaufmann Publishers Inc., San Francisco, CA, USA
32. Shilane P, Min P, Kazhdan M, Funkhouser T (2004) The Princeton shape benchmark. In: Proc. International Conference on Shape Modeling and Applications (SMI'04), IEEE Computer Society, pp 167–178

33. Sijbers J, Verhoye M, Scheunders P, der Linden AV, Dyck DV, Raman E (1997) Watershed based segmentation of 3D MR data for volume quantization. Magnetic Resonance Imaging 15(6):679–688
34. Tangelder JWH, Veltkamp RC (2008) A survey of content based 3D shape retrieval methods. Multimedia Tools and Applications 39(3):441–471
35. Vranic D (2004) 3D model retrieval. PhD thesis, University of Leipzig, Germany
36. Vranic D (2005) DESIRE: a composite 3D-shape descriptor. In: Proc. IEEE International Conference on Multimedia and Expo (ICME'05), IEEE, Los Alamitos, CA, USA, pp 962–965.
37. Wessel R, Klein R (2010) Learning the compositional structure of man-made objects for 3d shape retrieval. In: Proc. Eurographics Workshop on 3D Object Retrieval (3DOR'10), Eurographics Association, pp 39–46
38. Wessel R, Novotni M, Klein R (2006) Correspondences between salient points on 3D shapes. In: Proc. GI Workshop on Vision, Modeling, and Visualization (VMV'06), pp 365–372
39. Zezula P, Amato G, Dohnal V, Batko M (2005) Similarity Search: The Metric Space Approach (Advances in Database Systems). Springer-Verlag New York, Inc., Secaucus, NJ, USA