SHape REtrieval Contest 2007: Watertight Models Track

Daniela Giorgi, Silvia Biasotti, Laura Paraboschi

CNR - IMATI
Via De Marini 6, 16149, Genoa, Italy
{daniela.giorgi,silvia.biasotti,laura.paraboschi}@ge.imati.cnr.it

1 Introduction

Despite the availability of many effective 3D retrieval methodologies, only a handful of commercial products currently incorporate such techniques. A major barrier to the adoption of these techniques in commercial services is the lack of standardized evaluation: it is almost never obvious what is the best shape characterization or the best similarity measure for a given domain. Having a common understanding on 3D shape retrieval would help users to orient themselves to select the retrieved technique most suitable for their own specific needs. In this context, the aim of SHREC is to evaluate the performance of existing 3D-shape retrieval algorithms, in terms of their strengths as well as their weaknesses, using a common test collection that allows for a direct comparison of methods.

The pecularity of 3D media retrieval is given by the existence of many different representations for 3D shapes, from point-set models, to the many types of boundary representations and decomposition models [1]. Each particular representation is suitable to cope with particular application needs. For this reason, after the first successful experience of SHREC 2006, the contest has moved towards a multi-track organization.

In this report we present the results of the Watertight Models Track. Watertight models are object models represented by seamless surfaces, meaning that there are no defective holes or gaps. They turn out to be useful for many applications, such as rapid prototyping or digital manifacturing.

2 Data collection and queries

The collection to search in was made of 400 watertight mesh models, subdivided into 20 classes of 20 elements each. The experiment was designed so that each model was used in turn as a query against the remaining part of the database, for a total number of 400 queries. For a given query, the goal of the track is to retrieve the most similar objects, that is, the whole set of objects of the class it belongs to.

The type and categorization of the models in a database are crucial when testing a retrieval method, and it is difficult to separate out the influence of the dataset in the performance [2]. On the basis of these observations, we built our benchmark, shown in Figure 1. We manually established the ground truth, so that the classes exhibit sufficient and diverse variation, from pose change (e.g. the "armadillo" class) to shape variability in the same semantic group (e.g. the class of vases). All classes are made up of the same number of objects (20), so that *generality* is kept constant for each query [3], thus preventing from giving a different level of importance to different queries. Generality represents the fraction of relevant items to a given query with respect to the irrelevant embedding, i.e. the whole set of non-relevant models in the database; as observed in [3], it is a major parameter in influencing the retrieval performance.

The original models of our database were collected from several web repositories, namely the National Design Repository at Drexel University [6], the AIM@SHAPE repository [7], the Princeton Shape Benchmark [8], the CAESAR Data Samples [9], the McGill 3D Shape Benchmark [10], the 3D Meshes Research Database by INRIA GAMMA Group [11], the Image-based 3D Models Archive.

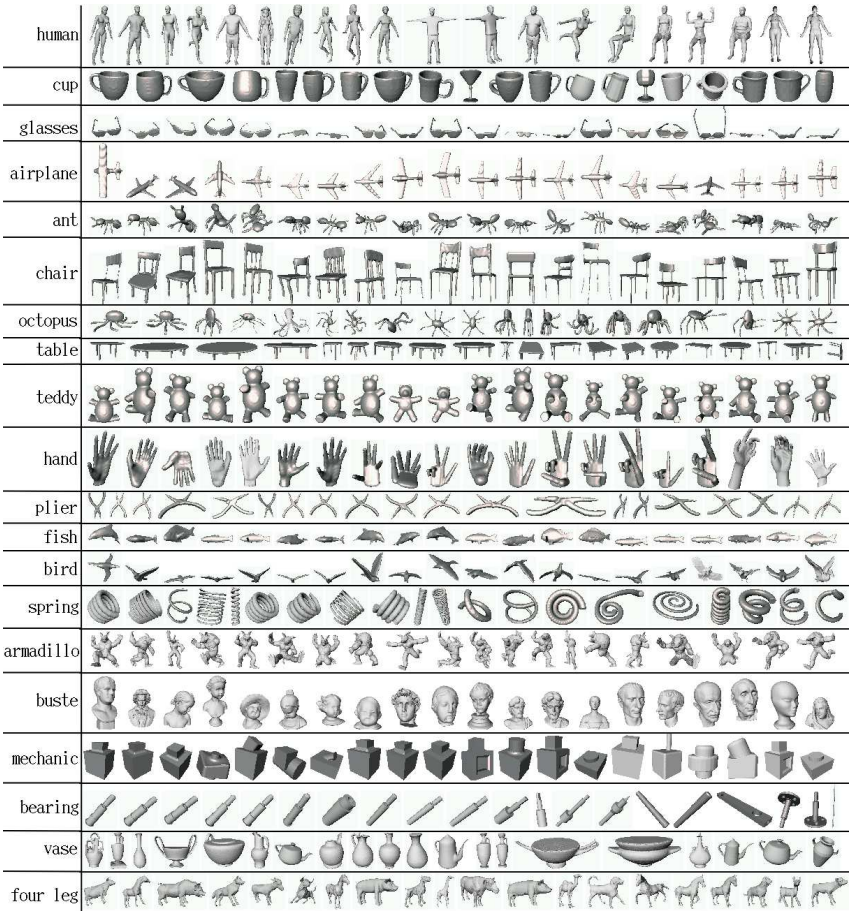

Figure 1: The database that has been used, divided into classes.

3 Participants

Each participant was asked to submit up to 3 runs of his/her algorithm, in the form of 400×400 dissimilarity matrices; each run could be for example the result of a different setting of parameters or the use of a different similarity metric. We remind that the entry (i, j) of a dissimilarity matrix represent the distance between models i and j.

This track saw 5 groups of participants:

1. Ceyhun Burak Akgül, Francis Schmitt, Bülent Sankur and Yücel Yemez, who sent 3 matrices;
2. Mohamed Chaouch and Anne Verroust-Blondet with 2 matrices;
3. Thibault Napoléon, Tomasz Adamek, Francis Schmitt and Noel E. O'Connor with 3 matrices;
4. Petros Daras and Athanasios Mademlis sent 1 matrix;
5. Tony Tung and Francis Schmitt with 3 matrices.

For details on the algorithms and the different runs proposed by the participants, the reader is referred to their papers, included at the end of this report.

4 Performance measures

As observed in section 2, each query has its own set of 20 relevant items. We evaluated all the methods using the standard measures briefly described below.

1. **Precision** and **recall** are two fundamental measures often used in evaluating search strategies. Recall is the ratio of the number of relevant records retrieved to the total number of relevant records in the database, while precision is the ratio of the number of relevant records retrieved to the size of the return vector [4]. In our contest, for each query the total number of relevant records in the database is always 20, that is the size of each class. Starting from here, we evaluate the precision-recall measures for each query, and then average it over each class and over the entire database. An averaged recall value can be calculated through the so-called **average dynamic recall**, defined in our context as $ADR = \frac{1}{20}\sum_{i=1}^{20}\frac{RI(i)}{i}$, where $RI(i)$ indicates the number of retrieved relevant items within the first i retrieved items. $ADR \in [0,1]$ and its best value is $ADR = \frac{1}{20}\sum_{i=1}^{20}\frac{i}{i} = 1$.

2. We compute the **percentage of success** for the **first (PF)** and the **second (PS)** retrieved items, i.e. the probability of the first and second elements in the return vector to be relevant to the given query, and average them over the whole set of queries. For an ideal method $PF = PS = 100\%$.

3. With respect to a query, the **average ranking** is computed averaging the retrieval ranking (i.e. the positions in the return vector of ordered items) of all relevant items. The lower this value, the better the method. The optimal value in our experimental setting is $\frac{1+2+\cdots+20}{20} = 10.5$.

4. The **last place ranking** is defined as $L = 1 - \frac{Rank-n}{N-n}$, where $Rank$ indicates the rank at which the last relevant object is found, n is the number of relevant items ($n = 20$), and N is the size of the whole database ($N = 400$) [5]. In this case, the performance of a method is as good as L is high; in fact $L \in [0,1]$, and the best value, occurring when the last relevant object is in the 20^{th} position, is $L = 1 - \frac{20-20}{400-20} = 1$.

5. There is a series of vectors to describe the performance of retrieval methods that descend from the so called *gain vector* G, which is, in our context, a 400-sized vector such that $G(i) = 1$ if the *i-th* retrieved item is in the same class of the query, 0 otherwise. The ideal gain vector is $IG(i) = 1$, $\forall i = 1, \ldots, 400$. The **cumulated gain vector** is defined as

$$CG(i) = \begin{cases} G(1) & i = 1 \\ CG(i-1) + G(i) & otherwise \end{cases};$$

in our case the ideal vector would be $ICG(i) = 1 \cdot i, \quad i = 1, \ldots, 400$.
The **discounted cumulated gain vector** is

$$DCG(i) = \begin{cases} CG(1) & i = 1 \\ DCG(i-1) + (G(i)/\log(i)) & otherwise \end{cases}.$$

The ideal vector $IDCG$ is obtained using ICG and IG instead of, respectively, CG and G.

We implemented these measures using the software Matlab, version 7.1 (R14), installed on a Intel(R) Pentium(R) 4 CPU 3.00Ghz, 1.00 Gb of RAM.

5 Results and discussion

Most of the participants sent more than one matrix. In what follows, we compare the performance of the participants using their single best run, selected using the previously described measures. Our choice is motivated by reasons of readability of tables and figures; anyway, we report the results of each run in the Appendix. For all participants, the selected run coincides with the best one according to all measures used, except for Agkul et al., that proposed two runs with very similar performances, so that the choice of the best one was not unique; in this case, we choose the best performing one in terms of the area of the precision-recall graph. The comparison of the performance of different runs for the same authors reported in the Appendix allows to evaluate the dependence of each method on different choices of parameters. A general observation is that almost all the methods of the same

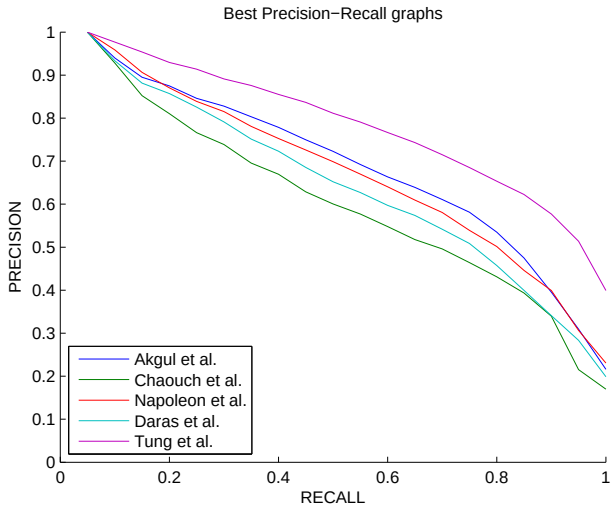

Figure 2: Comparing the best final Precision-recall graphs of each participant.

authors perform more or less the same.

Fig.2 shows the standard **precision-recall graph**, plotting for each participant precision values vs. recall values. We remind that curves shifted upwards and to the right indicate a superior performance.

Numerical values for both the average (w.r.t. all the queries) **precision** and **recall** for return vectors of 20, 40, 60 and 80 items (i.e. 1, 2, 3, and 4 times the number of relevant items to each query) are reported in Table 1 (a) and (b), respectively.

Precision after	20	40	60	80
Akgul et al.	0.626375	0.366062	0.262125	0.205469
Chaouch et al.	0.546250	0.329437	0.241417	0.190938
Napoleon et al.	0.604875	0.366312	0.262750	0.205719
Daras et al.	0.564500	0.346312	0.252208	0.199594
Tung et al.	0.714875	0.414375	0.290958	0.225687
Ideal	1	0.5	0.3	0.25

(a)

Recall after	20	40	60	80
Akgul et al.	0.626375	0.732125	0.786375	0.821875
Chaouch et al.	0.546250	0.658875	0.724250	0.763750
Napoleon et al.	0.604875	0.732625	0.788250	0.822875
Daras et al.	0.564500	0.692625	0.756625	0.798375
Tung et al.	0.714875	0.828750	0.872875	0.902750
Ideal	1	1	1	1

(b)

Table 1: Precision and Recall after 20, 40, 60 and 80 retrieved items.

The **average dynamic recall (ADR)** values for each participants are listed in the following table. As before, these values refer to the average value with respect to all the queries.

	Akgul et al.	Choauch et al.	Napoleon et al.	Daras et al.	Tung et al.
ADR	0.7931	0.7206	0.7795	0.7546	0.8577

Deepening in this kind of analysis, we can also deal with **class precision**, so that it is possible to make consideration not only on the average performance on the given database, but also on the specific class results. In this sense we invite the reader to have a look at the graphs in Fig. 5. It can be seen that the methods can perform in a very different manner when dealing with different classes of objects; for example, while in (a) Chaouch et al. gives the worst result (armadillo class), in (b) it yelds the best one (tables class). Moreover, some classes are uniformly easy to deal with, such as the class of pliers (c), while others are uniformly difficult, as the class of vases (d).

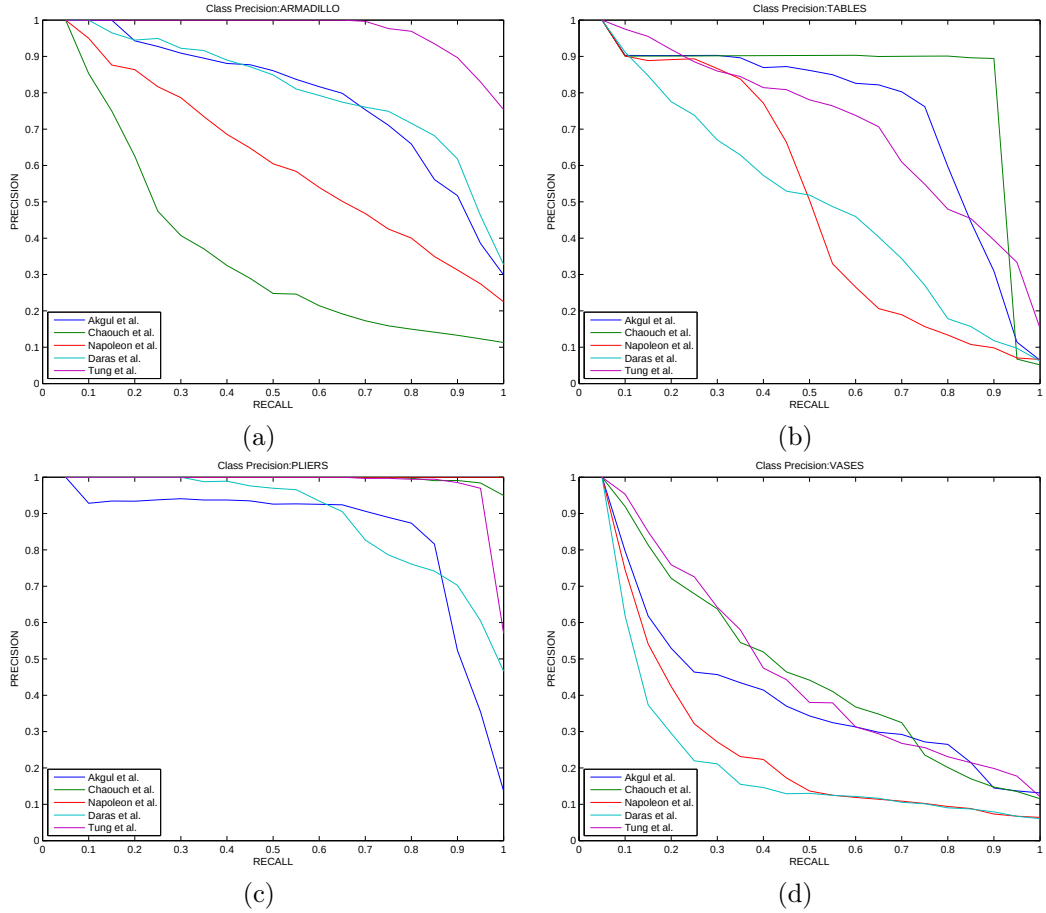

Figure 3: Precision-recall graphs on 4 classes of the database.

Returning to the whole database, we observed that all the methods guarantee the identity property (i.e. $d(Q_i, Q_i) = 0 \; \forall i = 1, \ldots, 400$, Q_i i^{th} model in the database), so that the **percentage of success PF** of the first retrieved item is always 100%:

	Akgul et al.	Choauch et al.	Napoleon et al.	Daras et al.	Tung et al.
PF	100%	100%	100%	100%	100%

,

The **percentages of success PS** w.r.t. the second retrieved item are listed below:

	Akgul et al.	Choauch et al.	Napoleon et al.	Daras et al.	Tung et al.
PS	93.97%	92.81%	95.87%	93.33%	97.68%

.

Concerning the **average ranking** we refer the reader to the histogram in Figure 4(a). Let us remark that in our case the ideal value for the average rank is 10.5, and a lower height of the bars indicate a superior performance.

The **last place ranking** is evaluated in Figure 4(b). The value yield by an ideal method is equal to 1. This measure gives an estimate of the number of the retrieved items a user has to search in order to have a reasonable expectation of finding all relevant items. The higher the bars in the histogram, the lower the number of items to check, meaning better results.

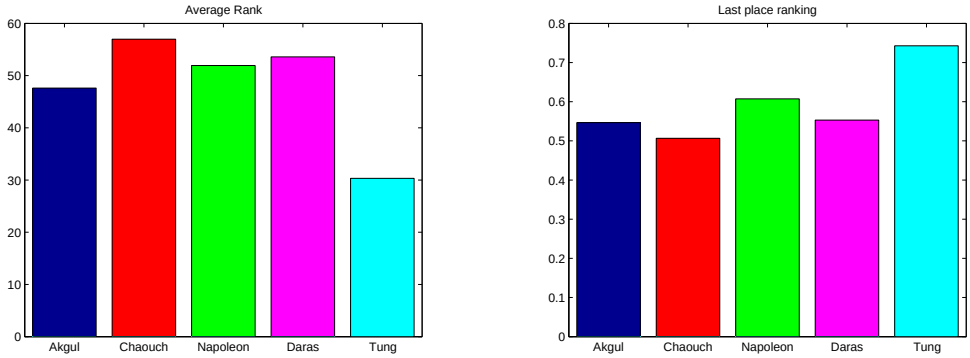

Figure 4: Average ranking (a) and last place ranking (b) of all the methods.

Finally, let us deal with the **mean cumulated** and **mean discounted cumulated gain vectors**, illustrated in Fig. 5. We show the 400-elements vectors (top), as defined in Section 4, and also a detailed view of the behavior of the first 20 components of the vectors (bottom), that is the behavior for the very first part of the return vector.

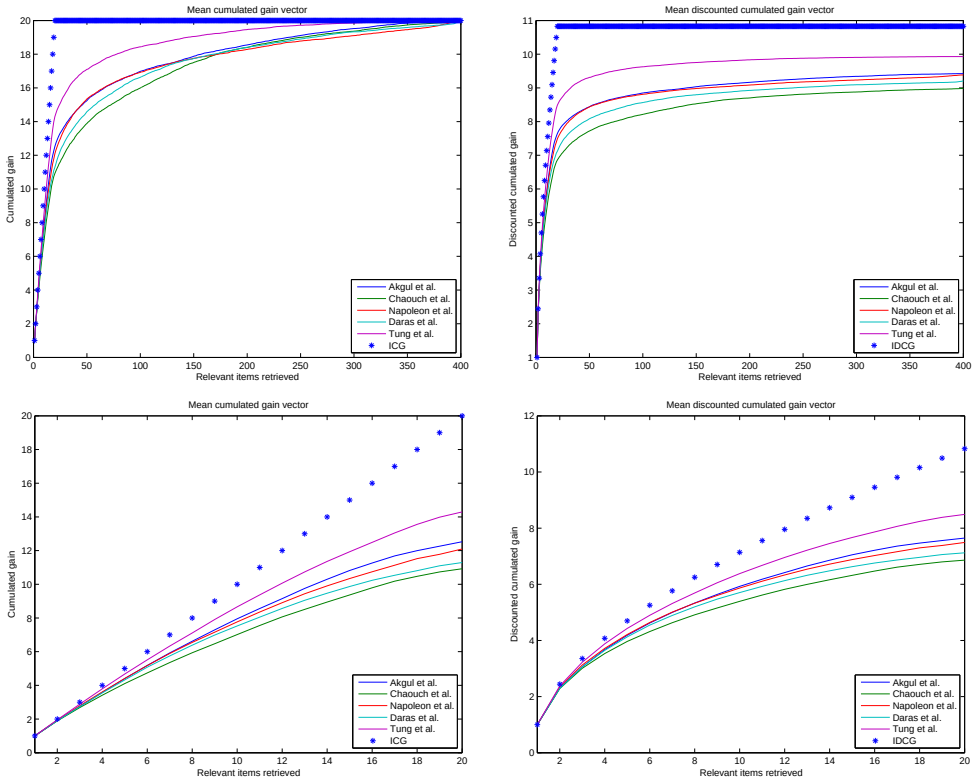

Figure 5: Mean cumulated gain vector (left) and mean discounted cumulated gain vector (right).

6 Conclusions

This paper proposed a comparative analysis of the retrieval performances of 5 different techniques, using a benchmark database of 400 models represented by watertight triangular meshes. Anyway, we warn the reader that, despite the care used in designing a benchmark and evaluating the results, it may happen that a single test collection delivers only a partial view of the whole picture.

References

[1] M. Mäntylä: *An introduction to solid modeling*, Computer Science Press, 1988.

[2] A. Smeulders, M. Worring, S. Santini, A. Gupta and R. Jain: Content-based image retrieval at the end of the early years, *IEEE Transactions on Pattern Analysis and Machine Intelligence* 22(12), 1349–1380, 2000.

[3] D. P. Huijsmans and N. Sebe: How to complete performance graphs in Content-Based Image Retrieval: Add generality and normalize scope, *IEEE Transactions on Pattern Analysis and Machine Intelligence* 27, 245–251, 2005.

[4] G. Salton and M. McGill: *Introduction to modern information retrieval*, McGraw Hill, 1983.

[5] J.P. Eakins, J.M. Boardman and M.E. Graham: Similarity retrieval of trademark images, *Multimedia* 5(2), 53–63, 1998.

[6] http://www.designrepository.org

[7] http://shapes.aim-at-shape.net

[8] P. Shilane, P. Min, M. Kazhdan and T. Funkhouser: The Princeton Shape Benchmark, *Proceedings of SMI 2004: International Conference on Shape Modeling and Applications* 167–178, 2004.

[9] http://www.hec.afrl.af.mil/HECP/Card1b.shtml

[10] http://www.cim.mcgill.ca/ shape/benchMark/

[11] http://www-c.inria.fr/gamma/download/

Appendix

In this Appendix we show separately the results of each single run of the participants. The majority of the participants sent 2 or 3 matrices, except from Daras et al. with only 1 matrix, for whose results the reader is referred to the previous Sections.

Two of the runs sent by Akgul et al., namely the 1st and the 3rd ones, are really similar; globally, the 3rd run seems to be the most performing one, although for some measures it may happen that the 1st run (e.g. with PSs) or even the 2nd run (e.g. last place ranking) give better results. For the other retrieval techniques, it is easier to establish the most performing run.

In any case, we can observe that almost all the methods of the same group perform more or less the same, thus generally revealing a small dependence on the choice of different parameters. A significant difference in the performance obtained by different runs of the same participant can be observed only for run number 3 sent by Napoléon et al., as a consequence of substantial changes in the shape descriptor used (see their paper at the end of this report for further details).

The precision-recall graphs are shown in Fig. 6. Numerical values of precision and recall are reported in Table 2. PFs are 100% for all techniques and runs, while Table 3 yield details about the PSs. The average dynamic recall is illustrated in Table 4. Average ranking and last place ranking are summarized by the histograms of Figure 7. Finally, Figure 8 is related to mean cumulated and mean discounted cumulated gain vectors.

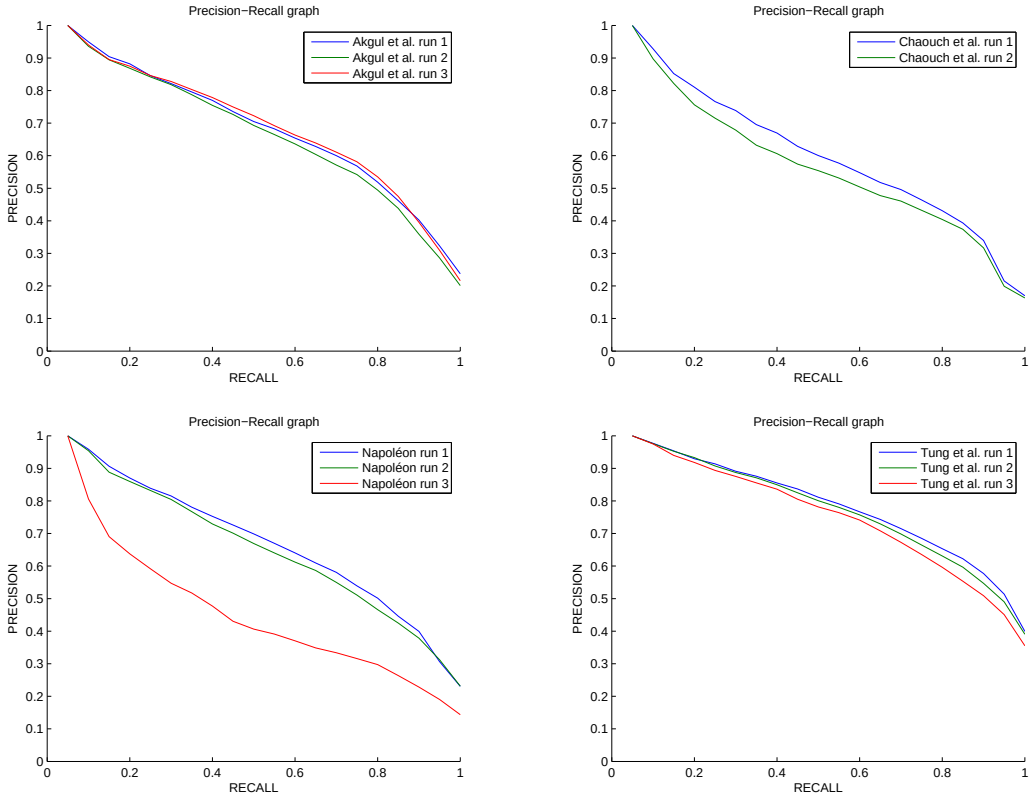

Figure 6: Precision-recall graphs for each participant.

	Precision after				Recall after			
	20	40	60	80	20	40	60	80
Akgul et al. run 1	0.6158	0.3639	0.2621	0.2068	0.6158	0.7277	0.7864	0.8271
Akgul et al. run 2	0.6016	0.3577	0.2566	0.2023	0.6016	0.7155	0.7697	0.8092
Akgul et al. run 3	0.6264	0.3661	0.2621	0.2055	0.6264	0.7321	0.7864	0.8219

	Precision after				Recall after			
	20	40	60	80	20	40	60	80
Chaouch et al. run 1	0.5462	0.3294	0.2414	0.1909	0.5462	0.6589	0.7242	0.7638
Chaouch et al. run 2	0.5125	0.3126	0.2299	0.1840	0.5125	0.6251	0.6896	0.7360

	Precision after				Recall after			
	20	40	60	80	20	40	60	80
Napoleon et al. run 1	0.6049	0.3663	0.2627	0.2057	0.6049	0.7326	0.7882	0.8229
Napoleon et al. run 2	0.5859	0.3601	0.2590	0.2020	0.5859	0.7201	0.7769	0.8081
Napoleon et al. run 3	0.4001	0.2658	0.2030	0.1663	0.4001	0.5316	0.6089	0.6650

	Precision after				Recall after			
	20	40	60	80	20	40	60	80
Tung et al. run 1	0.7149	0.4144	0.2910	0.2257	0.7149	0.8287	0.8729	0.9027
Tung et al. run 2	0.7035	0.4121	0.2905	0.2259	0.7035	0.8241	0.8715	0.9035
Tung et al. run 3	0.6831	0.4022	0.2867	0.2231	0.6831	0.8044	0.8601	0.8922

Table 2: Precision and recall after 20, 40, 60 and 80 retrieved items, for each run of the participants.

	PS
Akgul et al. run 1	94.96%
Akgul et al. run 2	93.61%
Akgul et al. run 3	93.97%
Chaouch et al. run 1	92.81%
Chaouch et al. run 2	89.73%
Napoleon et al. run 1	95.87%
Napoleon et al. run 2	95.42%
Napoleon et al. run 3	80.59%
Tung et al. run 1	97.68%
Tung et al. run 2	97.58%
Tung et al. run 3	97.49%

Table 3: Percentage of success of the second items retrieved, computed for each run.

	ADR
Akgul et al. run 1	0.7888
Akgul et al. run 2	0.7786
Akgul et al. run 3	0.7931
Chaouch et al. run 1	0.7206
Chaouch et al. run 2	0.6831
Napoleon et al. run 1	0.7795
Napoleon et al. run 2	0.7646
Napoleon et al. run 3	0.5770
Tung et al. run 1	0.8577
Tung et al. run 2	0.8496
Tung et al. run 3	0.8354

Table 4: (a) Average dynamic recall, computed for each run.

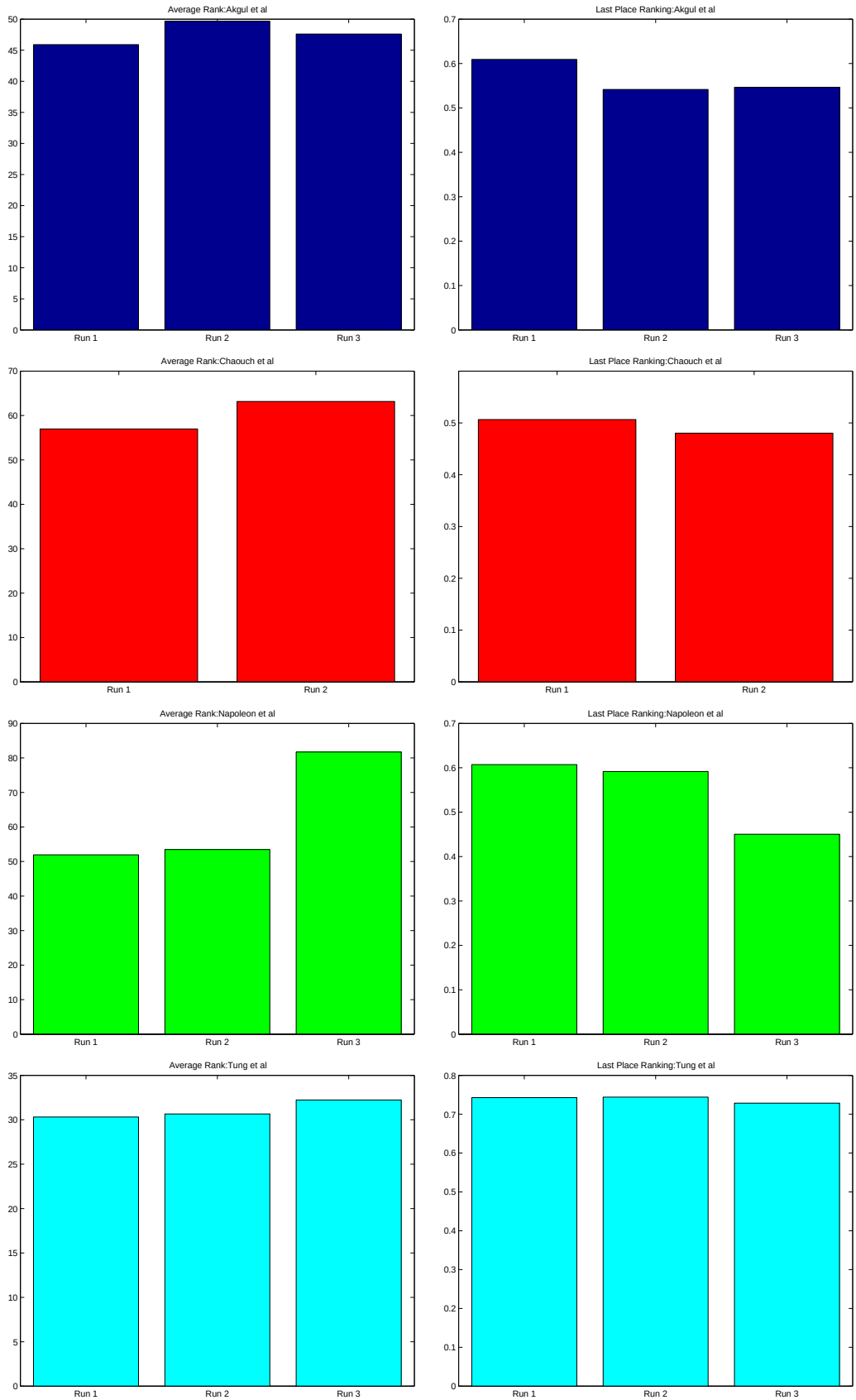

Figure 7: Average ranking (left) and last place ranking (right), for each run.

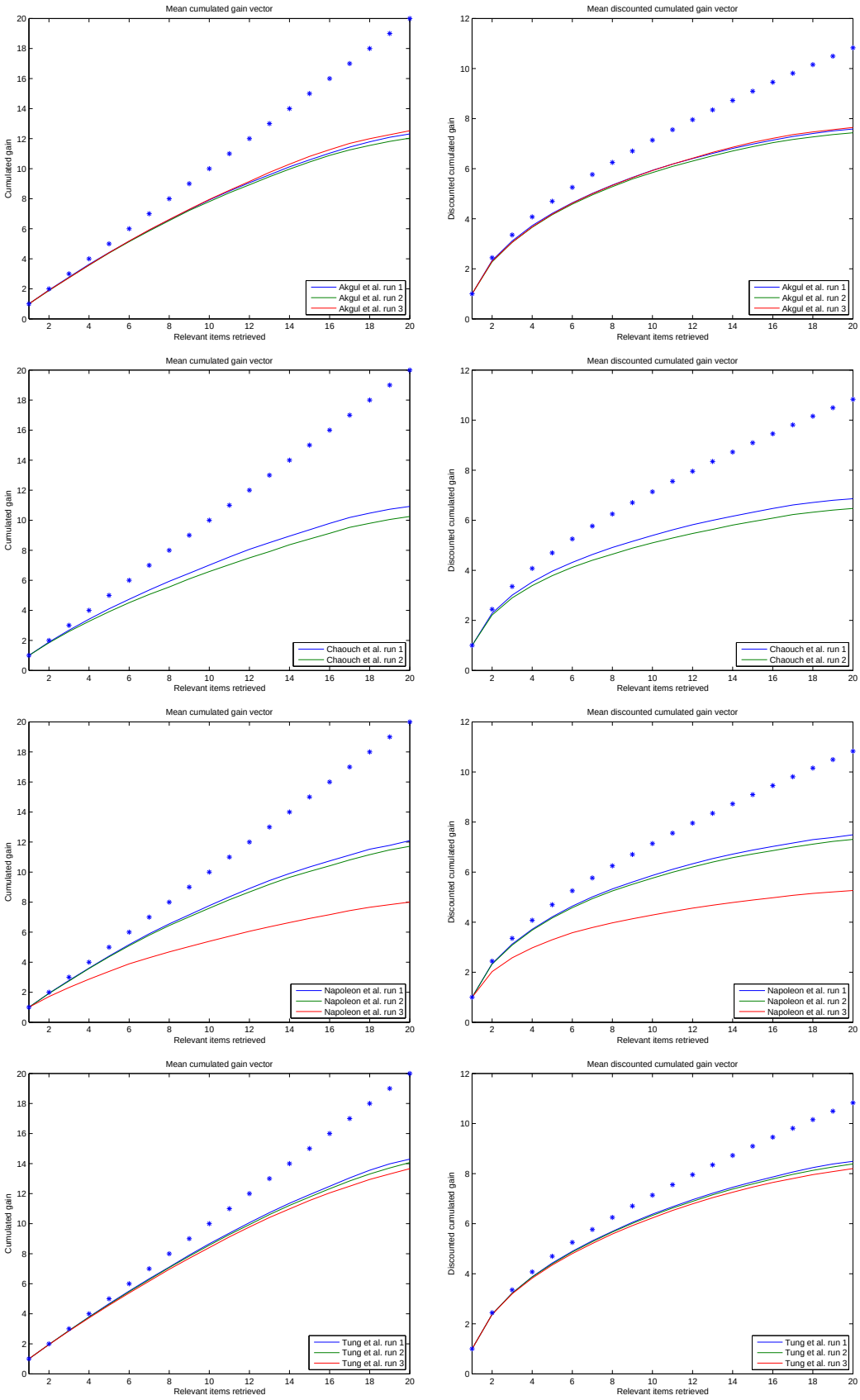

Figure 8: Cumulated gain vectors (left) and discounted cumulated gain vectors (right) within the first 20 items retrieved, for each run.

Multivariate Density-Based 3D Shape Descriptors

Ceyhun Burak Akgül[1,2], Francis Schmitt[1], Bülent Sankur[2], and Yücel Yemez[3]
[1]Boğaziçi University Electrical and Electronics Eng. Dept., 34342 Bebek, Istanbul, Turkey
[2]GET - Télécom Paris - CNRS UMR 5141, France
[3]Koç University Computer Eng. Dept., 34450 Sarıyer, Istanbul, Turkey

Abstract

In this brief communication, we provide an overview of the density-based shape description framework in and the details of the runs we have submitted to the *Watertight Models Track* of the *SHREC'07*.

1. Introduction

Density-based shape description is a framework to extract 3D shape descriptors from local surface features characterizing the object geometry [1, 2]. The feature information is processed with the kernel methodology for density estimation (KDE) [3, 4] and the probability density function (pdf) of the local feature is estimated at chosen target points. The shape descriptor vector is then simply a discretized version of this probability density. This density-based approach provides a mechanism to convert local shape evidences, using KDE, into a global shape description. Our recent work on density-based shape descriptors [1, 2] for 3D object retrieval has proven that this scheme is both computationally rapid and effective compared to other state-of-the-art descriptors. In this brief communication, we provide an overview of the density-based shape description framework in Section 2 and the details of the runs we have submitted to the *Watertight Models Track* of the *SHREC'07* in Section 3.

2. Overview of the Description Framework

A density-based descriptor of a 3D shape is defined as the discretized pdf of some surface feature *S* taking values within a subspace $\mathcal{R}_S$ of $\mathbb{R}^m$. The feature *S* is local to the surface patch and treated as a random variable. For an object O_l represented as a triangular mesh, evaluating the feature *S* at each triangle and/or vertex, we can obtain a set of observations about *S*, that we call *the source set*, denoted as $\{s_k \in \mathcal{R}_S\}_{k=1}^K$. Let $f(\cdot|O_l)$ be the pdf of *S*. Using the source set $\{s_k \in \mathcal{R}_S\}_{k=1}^K$, the value of this pdf at an arbitrary *m*-dimensional point *t* (which is in the range space $\mathcal{R}_S$ of the feature *S*) can be estimated by the following KDE equation [3]:

$$f(t|O_l) = \sum_{k=1}^{K} w_k |H|^{-1} \mathcal{K}\left(H^{-1}(t - s_k)\right) \tag{1}$$

where $\mathcal{K}: \mathbb{R}^m \to \mathbb{R}$ is a kernel function, H is a $m \times m$ matrix composed of a set of design parameters called *bandwidth* parameters, and w_k is the importance weight associated with the *k*th observation s_k. Suppose that we have specified a finite set of points $\bar{\mathcal{R}}_S = \{t_n \in \mathcal{R}_S\}_{n=1}^N$, called *the target set*, within $\mathcal{R}_S$. The density-based descriptor $\mathbf{f}_{S|O_l}$ for the object O_l (with respect to the feature *S*) is then simply an *N*-dimensional vector whose entries consist of the pdf values computed at the target set $\bar{\mathcal{R}}_S = \{t_n \in \mathcal{R}_S\}_{n=1}^N$, that is, $\mathbf{f}_{S|O_l} = \left[f(t_1|O_l), \ldots, f(t_N|O_l)\right]$. To convert this general framework into a practical description scheme, we have to address the following issues:

(i) <u>*Feature Design:*</u> Which surface features should be used to capture local characteristics of the surface? (See Section 2.1.)

(ii) *Target Selection:* How to determine the targets $t_n \in \overline{\mathcal{R}}_S, n = 1,...,N$ at which we will evaluate the KDE equation in (1)? (See Section 2.2.)

(iii) *Density Estimation:* How to choose the kernel function $\mathcal{K}$ and how to set the bandwidth parameter matrix H in (1) and how to evaluate this equation in a computationally efficient manner? (See Section 2.3.)

2.1. Feature Design

In the following, we assume that the 3D object is embedded on a canonical reference frame of $\mathbb{R}^3$ whose origin coincides with the center of mass of the object. The reference frame can be computed using the continuous PCA approach [4].

1. Radial distance R measures the distance of a surface point *Q* to the origin (centroid). Though not an effective shape feature all by itself; when coupled to other local surface features, it helps to manifest their distribution at varying radii.

2. Radial direction $\hat{\mathbf{R}}$ is a unit-norm vector $(\hat{R}_x, \hat{R}_y, \hat{R}_z)$ collinear with the ray traced from the origin to a surface point *Q*. This unit-norm vector is obviously scale-invariant.

3. Normal direction $\hat{\mathbf{N}}$ is simply the unit normal vector at a surface point and represented as a 3-tuple $(\hat{N}_x, \hat{N}_y, \hat{N}_z)$. Similar to the radial direction $\hat{\mathbf{R}}$, the normal $\hat{\mathbf{N}}$ is scale invariant.

4. Radial-normal alignment A is the absolute cosine of the angle between the radial and normal directions and is computed as $A = \left|\langle \hat{\mathbf{R}}, \hat{\mathbf{N}} \rangle\right| \in [0,1]$. We can alternatively remove the absolute value and redefine *A* as $A = \langle \hat{\mathbf{R}}, \hat{\mathbf{N}} \rangle \in [-1,1]$ for meshes with reliable orientation. This feature measures crudely how the surface deviates locally from sphericity. For example, if the local surface approximates a spherical cap, then the radial and normal directions align, and the *alignment A* approaches unity.

5. Tangent-plane distance D stands for the absolute value of the distance between the tangent plane at a surface point and the origin. *D* is related to the radial distance *R* by $D = RA$.

6. The shape index SI provides a local categorization of the shape into primitive forms such as spherical cap and cup, rut, ridge, trough, or saddle. In the present work, we consider the parameterization proposed in [5] given by

$$SI = \frac{1}{2} - \left(\frac{2}{\pi}\right) \arctan\left(\frac{\kappa_1 + \kappa_2}{\kappa_1 - \kappa_2}\right),$$

where κ_1 and κ_2 are the principal curvatures at the surface point. *SI* is confined within the range $[0,1]$ and not defined when $\kappa_1 = \kappa_2 = 0$ (planar patch).

Our previous work [1, 2] has shown that the density-based framework is more effective when we consider the joining of the local features described above, to obtain even higher dimensional features. In the present work, we consider three such instances:

Radial feature S_1 is obtained by augmenting the scale-invariant radial direction vector $\hat{\mathbf{R}}$ with the rotation-invariant radial distance *R*. The resulting 4-tuple $S_1 \triangleq (R, \hat{R}_x, \hat{R}_y, \hat{R}_z)$ can be viewed as an alternative to Cartesian coordinate representation of the surface point. In this parameterization, however, the distance and direction information are decoupled. With this decoupling, the range of individual features

can be determined independently. In fact, $\hat{\mathbf{R}}$ lies on the unit 2-sphere, and the scalar R lies on the interval $(0, r_{\max})$, where $r_{\max}$ depends on the size of the surface.

Tangent plane feature S_2 is obtained by joining the tangent plane distance D with the normal direction $\hat{\mathbf{N}}$, providing a 4-dimensional vector $S_2 \triangleq (D, \hat{N}_x, \hat{N}_y, \hat{N}_z)$ that corresponds to the representation of the local tangent plane. As in the radial case, this representation also separates the distance and direction information concerning the tangent plane.

Finally, we define a third feature $S_3 \triangleq (R, A, SI)$, which aims at encoding the interaction between the radial and normal directions through the alignment feature A and at adding further local surface information through the shape index SI. Again the radial distance R augments the two-tuple (A, SI) to capture the characteristics of the shape at different distances from the center of the object.

In our runs, we will use the discretized pdfs of all these three features as descriptors and combine their individual discrimination capabilities (see Section 3).

2.2. Target Selection

The targets sets for our local features occur as the Cartesian products of their individual constituents. For instance, the S_1-feature is composed of a scalar feature $R \in (0, r_{\max})$ and a unit-norm 3-vector $\hat{\mathbf{R}} \in \mathcal{S}^2$. Accordingly to determine the target set $\bar{\mathcal{R}}_{S_1}$, we first uniformly sample the interval $(0, r_{\max})$ to obtain a distance set $\bar{\mathcal{R}}_R$, then partition the unit 2-sphere using the octahedron subdivision scheme described in [1] to obtain a direction set $\bar{\mathcal{R}}_{\hat{\mathbf{R}}}$, and finally taken their Cartesian products to obtain $\bar{\mathcal{R}}_{S_1} = \bar{\mathcal{R}}_R \times \bar{\mathcal{R}}_{\hat{\mathbf{R}}}$. Note that $r_{\max}$ depends on the type of scale normalization applied to the object (see Section 3). The target set for $S_2 \triangleq (D, \hat{N}_x, \hat{N}_y, \hat{N}_z)$ can be obtained likewise. Finally the target set for $S_3 \triangleq (R, A, SI)$ is given by $\bar{\mathcal{R}}_{S_3} = \bar{\mathcal{R}}_R \times \bar{\mathcal{R}}_A \times \bar{\mathcal{R}}_{SI}$, where both $\bar{\mathcal{R}}_A$ and $\bar{\mathcal{R}}_{SI}$ are uniformly sampled versions of the interval $[0,1]$ since both A and SI share the same unit-interval as range.

2.3. Density Estimation

It is known that the particular functional form of the kernel does not significantly affect the accuracy of the estimator [3]. In our scheme, we choose the Gaussian kernel since there exists a fast algorithm, the fast Gauss transform (FGT) [6], to rapidly evaluate large KDE sums in $O(K+N)$ instead of $O(KN)$-complexity of direct evaluation, where K is the number of sources and N is the number of targets.

The setting of the bandwidth matrix H has been shown to be critical for accurate density estimation [3], which in turn affects shape discrimination and retrieval performance [1]. The optimal bandwidth for KDE depends on the unknown density itself [3], making the appropriate choice of the bandwidth parameter a challenging problem. Since the density-based paradigm relies on the premise that the features of similar shapes induce similar probability distributions, the optimal set of bandwidth parameters is expected to be different for each shape class. In content-based retrieval, since we do not have this kind of information, we can set the bandwidth parameter at either object-level or at database-level. In [1], it has been experimentally demonstrated that database level setting yields better discrimination performance by as much as 11%. Accordingly, we can set the bandwidth parameter by averaging all object-level bandwidths given by Scott's rule [3] or by averaging the covariance matrix of the observations over the objects. This averaging process has some intuitive plausibility as it eliminates object-level details and provides us with bandwidth parameters that regularize class-level information.

3. Experimental Details

Prior to descriptor computation, all models have been normalized so that descriptors are translation, rotation, flipping and scale invariant. For translation invariance, the object's center of mass is considered as the origin of the coordinate frame. For rotation and flipping invariance, we applied the continuous PCA algorithm [4]. For isotropic scale invariance, we calculate a scale factor so that the average point-to-origin distance is unity.

Details about the runs that we submit for the Watertight Track are as follows:

Run 1.

- We reorient the mesh so that the angle between radial and normal directions at a surface point is always acute.
- We calculate a bandwidth matrix by averaging the covariance matrices of the observations over the meshes.
- We use three descriptors:
  1. 1024-point pdf of $S_1 \triangleq (R, \hat{R}_x, \hat{R}_y, \hat{R}_z)$
  2. 1024-point pdf of $S_2 \triangleq (D, \hat{N}_x, \hat{N}_y, \hat{N}_z)$
  3. 576-point pdf of $S_3 \triangleq (R, A, SI)$ where the alignment A is defined as $A = \left|\langle \hat{\mathbf{R}}, \hat{\mathbf{N}} \rangle\right| \in [0,1]$.
- We use the L_1-metric to obtain three distance measures that we sum up to obtain a final dissimilarity value.

Run 2.

- We keep the orientation of the mesh.
- The remaining setting is the same as the Run 1.

Run 3.

- The setting is the same as the Run 2 except that the alignment A is defined as $A = \langle \hat{\mathbf{R}}, \hat{\mathbf{N}} \rangle \in [-1,1]$.

4. References

[1] Ceyhun Burak Akgül, Bülent Sankur, Yücel Yemez, Francis Schmitt, "Density-Based 3D Shape Descriptors", EURASIP Journal on Advances in Signal Processing, vol. 2007, Article ID 32503, 16 pages, 2007.

[2] Ceyhun Burak Akgül, Bülent Sankur, Francis Schmitt, Yücel Yemez, "Multivariate Density-Based 3D Shape Descriptors", accepted for Shape Modeling International 2007, Lyon, France, June 13-15, 2007.

[3] D. W. Scott, *Multivariate Density Estimation*, Wiley Interscience, 1992.

[4] D. V. Vranic, *3D Model Retrieval*, PhD. Thesis, University of Leipzig, 2004.

[5] C. Dorai and A. K. Jain, "COSMOS – A Representation Scheme for 3D Free-Form Objects", *IEEE Trans. on PAMI*, 19(10): 1115-1130, October 1997.

[6] C. Yang, R. Duraiswami, N. A. Gumerov, and L. Davis, "Improved Fast Gauss Transform and Efficient Kernel Density Estimation", *IEEE Int. Conf. on Comp. Vision (ICCV'03)*, 1: 464, 2003.

2D/3D Descriptor based on Depth Line Encoding

Mohamed Chaouch and Anne Verroust-Blondet *
INRIA Rocquencourt
Domaine de Voluceau, B.P. 105
78153 Le Chesnay Cedex, FRANCE
Email: mohamed.chaouch, anne.verroust@inria.fr

1 Introduction

For Watertight Models Track of SHREC'07 (SHape REtrieval Contest 2007 organized by the Network of Excellence AIM@SHAPE), we tested our new 2D/3D approach based on depth lines (DLA) with two different similarity measures.
Our method is detailed in the article "3D Model Retrieval based on Depth Line Descriptor" [CVB07] with the following abstract: *"In this paper, we propose a novel 2D/3D approach for 3D model matching and retrieving. Each model is represented by a set of depth lines which will be afterward transformed into sequences. The depth sequence information provides a more accurate description of 3D shape boundaries than using other 2D shape descriptors. Retrieval is performed when dynamic programming distance (DPD) is used to compare the depth line descriptors. The DPD leads to an accurate matching of sequences even in the presence of local shifting on the shape. Experimentally, we show absolute improvement in retrieval performance on the Princeton 3D Shape Benchmark database."*

2 Method

Our 3D shape retrieval system compares 3D models based on their visual similarity using depth lines extracted from depth images:
- The process first normalizes and scales 3D model into a bounding box.
- Then, it computes the set of $N \times N$ depth-buffer images associated to the six faces of the bounding box.
- The system then generates $2 \times N$ depth lines per image, considering each depth image as a collection of N horizontal and N vertical depth lines.
- Finally, each depth line is encoded in a set of N states called sequence of observations.

The shape descriptor consists in the set of $6 \times 2 \times N$ sequences, with $N = 32$.
Please see our paper for further details.

3 Experimental results

We submitted two runs:

1. In run 1 (DLA_DPD), to perform retrieval results, we tested the dynamic programming distance (DPD) because it tolerates some local shifting on the shape. We used here the Needleman-Wunsch algorithm [NW70].
2. In run 2 (DLA_HD), to compare the depth line descriptors, we used the Hamming distance (HD) which returns the number of corresponding state positions that differ.

*This work has been supported in part by the DELOS NoE on Digital Libraries (EU IST NoE G038-507618).

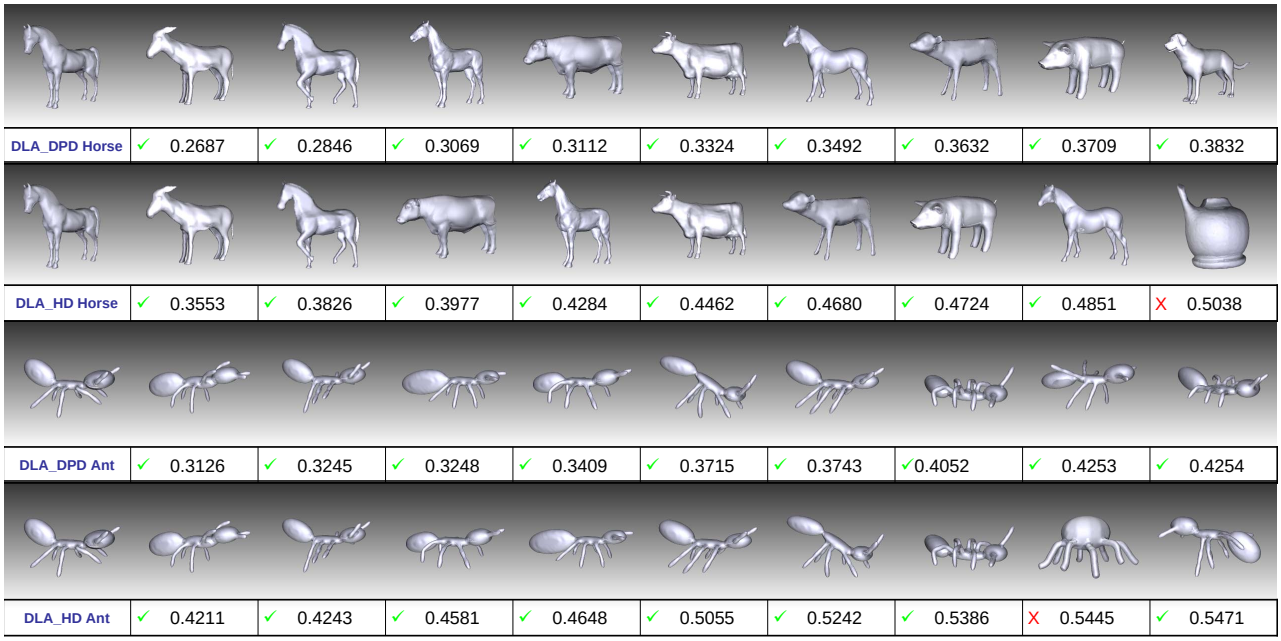

Figure 1: Examples of similarity search. For each query (from "Four-legged animals class" and "Ants class"), we show the top 9 objects matched with DLA approach (using DPD and HD). The similarities between the query models and the retrieved models are given below corresponding images. ✓ and x indicate that the retrieved models belong or don't belong to the query's class, respectively.

References

[CVB07] M. Chaouch and A. Verroust-Blondet. 3D model retrieval based on depth line descriptor. In *IEEE International Conference on Multimedia & Expo (ICME'07)*, Beijing, China, July 2007.

[NW70] S. Needleman and C. Wunsch. A general method applicable to the search for similarities in the amino acid sequence of two proteins. *Journal of Molecular Biology*, 48(3):443–453, 1970.

Multi-view 3D retrieval using silhouette intersection and multi-scale contour representation

Thibault Napoléon
ENST - CNRS UMR 5141
France
napoleon@enst.fr

Tomasz Adamek
CDVP
Dublin University
Ireland
adamekt@eeng.dcu.ie

Francis Schmitt
ENST - CNRS UMR 5141
France
schmitt@enst.fr

Noel E. O'Connor
CDVP
Dublin University
Ireland
oconnorn@eeng.dcu.ie

***Abstract*— We describe in this paper two methods for 3D shape indexing and retrieval of Watertight models proposed by the SHREC'07 Watertight track of the CNR-IMATI. The first one is a 2D multi-view method based on silhouettes intersections. The second one is another a 2D multi-view method which extracts the contour convexities and concavities at different scale levels. In this second approach the optimal matching of two shape representations is achieved using dynamic programming. This two methods show interesting results with a compromise between advantages and weaknesses of each one.**

I. Introduction

We proposed two methods for the Watertight track proposed by the CNR-IMATI for the 3D Shape Retrieval Contest 2007. Each one is based on a multi-view approach which keeps 3D object coherence by considering simultaneously a set of images in specific view directions. The various silhouettes of an object being strongly correlated, using a set of them help to better discriminate one object among others.

First of all, we have to get a robust normalization of the object pose and object scale in order to remain invariant to various geometrical transformations (translation, rotation, scaling). We used a Principal Continuous Component Analysis [1][2] and the smallest enclosing sphere [3] to solve these problems.

The first method is based on silhouettes intersection. We captured a set of views of an object and we extract its silhouette n each view. The distance between two silhouettes is chosen as equal to the number of pixels that are not common to the two silhouettes intersection. The distance between two objects is just defined as the sum of the distance between their two sets of silhouettes.

The second approach, is based on the contour convexities and concavities of the silhouettes of the 3D model presented in [4]. We capture a set of views of an object and we extract a normalized contour for each view. We build then a multi-scale shape representation, where for each contour point we store information about the convexity/concavity at different scale levels. We search the optimal match between two objects by computing the distance between their contour points.

The section 2 presents the normalization method for the 3D models. The section 3 describes the intersection methods. The section 4 presents the contour convexities and concavities approach. And the section 5 presents experimental results.

II. Model normalization

The first step before computing the distance between two 3D models is to find for them their best pose and scale in which they can be compared. To get a robust normalization of the object pose and object scale in order to remain invariant to various geometrical transformations (translation, rotation, scaling), we have to find a center, a scale factor and a pose for each object.

For the center and the scale, we use the smallest enclosing sphere S [3]. The normalization then becomes :

$$x = \frac{x - c_x(S)}{d(S)},\ y = \frac{y - c_y(S)}{d(S)}\ and\ z = \frac{z - c_z(S)}{d(S)}$$

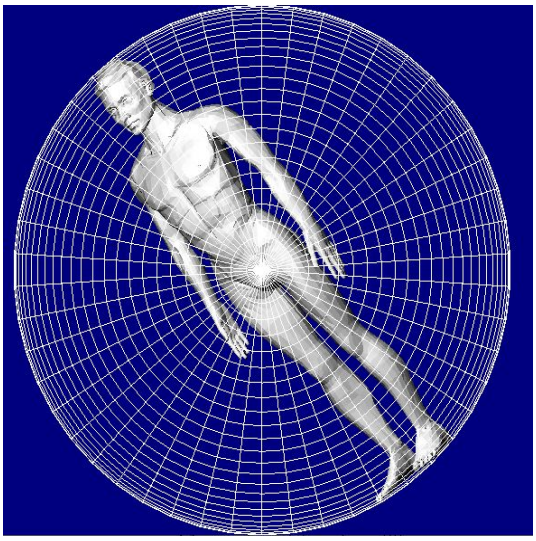

Fig. 1. *Smallest enclosing sphere.*

where $d(S)$ is the diameter of the smallest enclosing sphere and $c_i(S),\ i = x,\ y,\ z$ are the i-th coordinates of its centre. The main advantages of the smallest enclosing sphere are that it is fast to calculate and it allows maximizing the object size inside the unit sphere. This has for consequence to also maximize the object silhouette in any view direction, with the guaranty that the silhouette remains inside the unit disc inscribed in the

image domain associated to this view (no risk of accidental cropping of the silhouette).

It remains the problem of solving the normalization of the object pose. For this, we use the Continuous Principal Component Analysis [1][2]. This approach allows us to define and orientate the three principal axis of an object in a robust way and at a very reasonable computation cost.

III. Intersection descriptor

A. Signature extraction

The first step before computing a distance between two 3D models is to find for them their best pose and scale in which they can be compared. So, we normalize the object using the model normalization presented in the previous section.

in 2D/3D methods 3D object coherence is reinforced by considering simultaneously a set of images in specific view directions. The various silhouettes of an object being strongly correlated, using a set of them help to better discriminate one object among others (view [5] for more information). For this, we can use any set of view directions regularly distributed in space. We consider here the three orthogonal views along the oriented principal axis with parallel projections. Three views instead of a higher number of views allow reducing the descriptor size.

We choose an image size of 256x256 for each silhouette. This resolution gives a good precision and a reasonable computation time. To keep the maximum information from a silhouette, the simplest way is just to keep its image. We will then compare two silhouettes by superposing them and comparing their intersection with their union. A silhouette being a binary image we can store it in a compressed format without loss, which is fast to read and to decode when comparing silhouettes. The signature of an object is then simply constituted by the three compressed silhouettes corresponding to the three oriented principal directions.

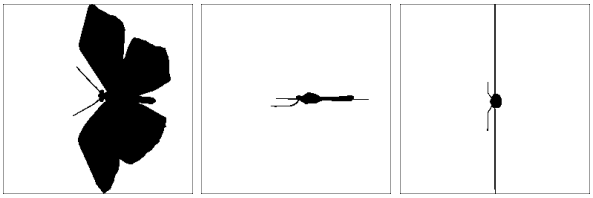

Fig. 2. *Three silhouettes of an object.*

B. Signature matching

The distance between two objects is defined as the distance between their two sets of silhouettes. The three silhouettes of each set being sorted according the three principal axis, this distance is then just defined as the sum of the distances of the three pairs of silhouettes, one pair per axis. The distance between two silhouettes is chosen as equal to the number of pixels that are non common to the two silhouettes, i.e. the difference between the areas of the silhouettes union and the silhouettes intersection. This measure can be done very efficiently directly on the compressed files by using a simple run length compression. The distance between two objects is then straightforward, simpler and fast to compute. To answer a query we just measure its distance to every database models and sort the list accordingly.

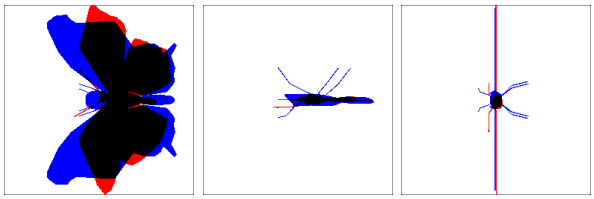

Fig. 3. *Intersections of two objects, in black the parts of the two objects, in blue the parts of the first one and in red the parts of the second one*

Fig. 4. *15 first results for shape retrieval using the intersection method. We can see the query Watertight model on the top left.*

IV. Contour convexities and concavities descriptor

A. Signature extraction

Like in the previous method, we find for each object the best pose and scale in which they can be compared by using the model normalization presented in section 2.

Similarly the 3D object coherence is reinforced by considering simultaneously a set of images in specific view directions.

We choose also an image size of 64x64 or 256x256 for each silhouette. The first one is not very precise but a small resolution reduces the descriptor size and the computation time. In an other way, a better resolution gives more precision with no noise but a bigger descriptor size and computation time. For the descriptor of a silhouette, we extract information about the convexity/concavity at 10 scale levels for each contour point [4]. The representation can be stored in the form of a 2D matrix where the columns correspond to contour points (contour parameter u) and the rows correspond to the different scale levels σ. The position (u, σ) in this matrix contains information about the degree of convexity or concavity for the u contour point at scale level σ. The simplified boundary contours at different scale levels are obtained via a curve evolution process. We represent each size normalized contour C with 100 contour points. It should be noted that we use

the same number of contour points for each shape to be compared. Let the contour C be parameterized by arc-length $u: C(u) = (x(u), y(u))$, where $u \in [0, N]$. The coordinate functions of C are convolved with a Gaussian kernel ϕ_σ of width $\sigma \in \{1, 2...\sigma_{max}\}$. The resulting contour, C_σ, becomes smoother with increasing value of σ, until finally the contour becomes convex.

We propose a very simple measure for the convexity and concavity of the curve. The measure is defined as the displacement of the contour between two consecutive scale levels. If we denote the contour point u at scale level σ as $p(u, \sigma)$, the displacement of the contour between two consecutive scale levels $d(u, \sigma)$ at point $p(u, \sigma)$ can be defined as the Euclidian distance between position of $p(u, \sigma)$ and $p(u, \sigma - 1)$.

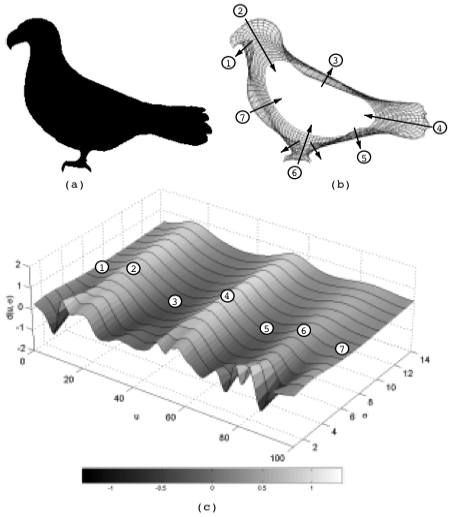

Fig. 5. *Example of extracting the MCC shape representation: (a)-original shape image, (b)-filtered versions of the original contour at different scale levels, (c)-final MCC representation for 100 contour points at 14 scale levels.*

B. Signature matching

When comparing two contours A and B, it is necessary to examine the distance between each contour point of both contours. If two contour points u_A and u_B are represented by their multi-scale features $d_A(u_A, \sigma)$ and $d_B(u_B, \sigma)$ respectively, then the distance between the two contour points can be defined as:

$$d(u_A, u_B) = \frac{1}{K} \sum_{\sigma=1}^{K} |d_A(u_A, \sigma) - d_B(u_B, \sigma)|$$

where K is the number of scale (here 10).

As part of the matching process, the best correspondence between contour points must be determined. We use a dynamic programming method with an $N * N$ distance table to conveniently examine the distances between corresponding contour points on both shapes. The columns represent contour points of one shape representation and the rows represent the contour points of the other. Each row/column entry in the table is the distance between two corresponding contour points calculated according to the previous equation.

Finding the optimal match between the columns corresponds to finding the lowest cost diagonal path through the distance table - see the example in Figure 6 where the contours' feature vectors are illustrated as grey levels along each axis.

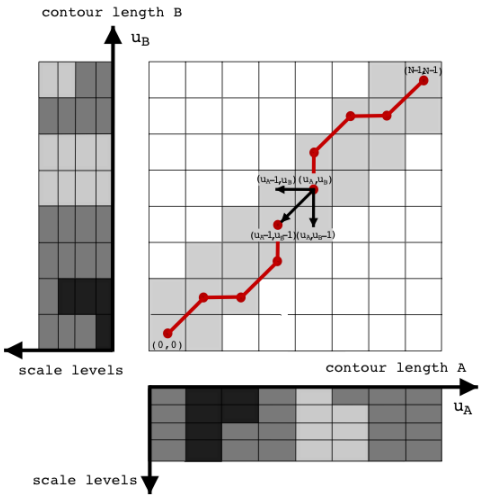

Fig. 6. *Illustration of matching two MCC representations by dynamic programming.*

Finally the distance between two objects is just defined as the distance between their two sets of silhouettes. The three silhouettes of each set being sorted according the three principal axis, this distance is then just defined as the sum of the distances of the three pairs of silhouettes, one pair per axis.

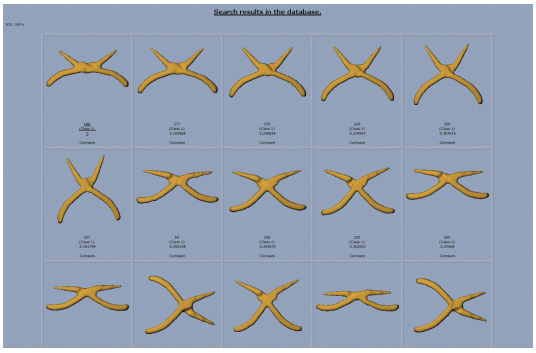

Fig. 7. *15 first results for shape retrieval using the contour convexities and concavities method. We can see the query Watertight model on the top left.*

V. Experimental results

The results presented here was obtained by using three orthogonal silhouettes aligned with the principal axis as described above and with a 256x256 pixels resolution for each silhouette. For the contour convexities and concavities method, we uses a normalized contour with 100 points and

10 scale levels.

We propose three runs for the SHREC'07 Watertight track:

- Run 1: The contour convexities and concavities descriptor, with 3 silhouettes aligned with the principal axis and a resolution of 256x256 pixels for each silhouettes.
- Run 2: The contour convexities and concavities descriptor, with 3 silhouettes aligned with the principal axis and a resolution of 64x64 pixels for each silhouettes.
- Run 3: The multi-view intersection descriptor, with 3 silhouettes aligned with the principal axis and a resolution of 256x256 pixels for each silhouettes.

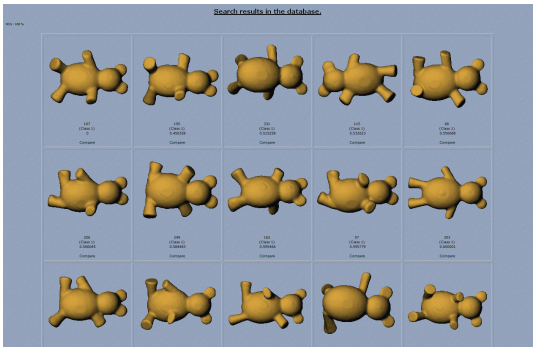

Fig. 8. *15 first results for the Watertight query 107.off using the contour convexities and concavities method with 256x256 pixels resolution and 3 silhouettes.*

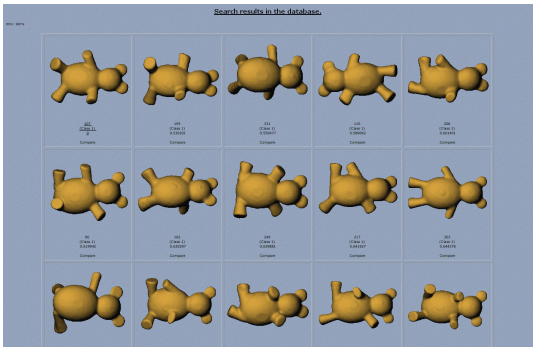

Fig. 9. *15 first results for the Watertight query 107.off using the contour convexities and concavities method with 64x64 pixels resolution and 3 silhouettes.*

The contour convexities and concavities descriptor results are better than the multi-view intersection descriptor results and the resolution of the silhouette have no effect. It is more robust with small variations of the shape and with the mirrored silhouettes problem. But the computation time for a query object of the two method is very different: with the contour convexities and concavities method the CPU time is $\sim$ 20 s and with the multi-view method $\sim$ 0,06 s.

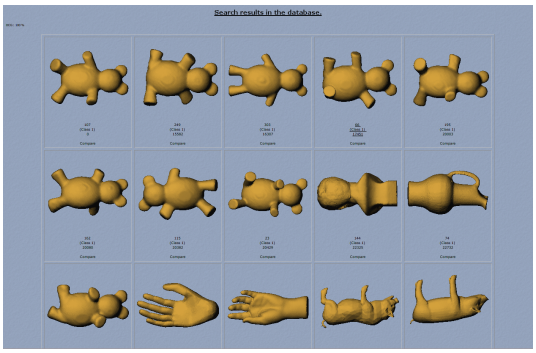

Fig. 10. *15 first results for the Watertight query 107.off using the intersection method with 256x256 pixels resolution and 3 silhouettes.*

VI. Conclusion

We have presented here two different methods on the Watertight SHREC'07 track. We observe that we obtain good results for the two approaches. We can notice that the intersection method is not very robust with small deformations of an object and the contour convexities and concavities method needs more important computation time. This weakness could be reduced by optimizing the source code.

VII. Acknowledgments

The authors wish to acknowledge the support of the European Commission under the FP6-027026-K-SPACE contract.

References

[1] D. V. Vranic, "3d model retrieval," Ph.D. dissertation, University of Leipzig, 2004.

[2] D. V. Vranic, D. Saupe, and J. Richter, "Tools for 3d-object retrieval: Karhunen-loeve transform and spherical harmonics," in *Proceedings of the IEEE 2001 Workshop Multimedia Signal Processing*, J.-L. Dugelay and K. Rose, Eds., Budapest, Hungary, Sept. 2001, pp. 271–274.

[3] K. Fischer and B. Gärtner, "The smallest enclosing ball of balls: Combinatorial structure and algorithms," *International Journal of Computational Geometry and Applications (IJCGA)*, vol. 14, pp. 341–387, 2004.

[4] T. Adamek and N. E. O'Connor, "A multiscale representation method for nonrigid shapes with a single closed contour," *IEEE Trans. Circuits Syst. Video Techn*, vol. 14, no. 5, pp. 742–753, 2004.

[5] D.-Y. Chen, X.-P. Tian, Y.-T. Shen, and M. Ouhyoung, "On visual similarity based 3D model retrieval," *Comput. Graph. Forum*, vol. 22, no. 3, pp. 223–232, 2003.

The Spherical Trace Transform

Petros Daras Dimitrios Tzovaras Apostolos Axenopoulos
Dimitrios Zarpalas Athanasios Mademlis Michael G. Strintzis

1 Introduction

The large amount of the available 3D models and their increasingly important role for many areas such as medicine, engineering, architecture, graphics design etc, showed up the need for efficient data access in 3D model databases. The important question arises is how to search efficiently for 3D objects into many freely available 3D model databases. A query by content approach seems to be the simpler and more efficient way. The method is briefly presented in the sequel. A detailed description of the method can be found in [1].

2 Descriptor Extraction Method

Every 3D object is expressed in terms of a binary volumetric function. In order to achieve translation invariance, the mass center of the 3D object is calculated and the model is translated so as its center of mass coincides with the coordinates system origin. Scaling invariance is also accomplished, by scaling the object in order to fit inside the unit sphere. Then, a set of concentric spheres is defined. For every sphere, a set of planes which are tangential to the sphere is also defined. Further, the intersection of each plane with the object's volume provides a spline of the object, which can be treated as a 2D image. Next, 2D rotation invariant functionals F are applied to this 2D image, producing a single value. Thus, the result of these functionals when applied to all splines, is a set functions defined on every sphere whose range is the results of the functional. Finally, a rotation invariant transform T is applied on these functions, in order to produce rotation invariant descriptors. For the needs of the SHREC, the implemented functionals F are the 2D Krawtchouk moments, the 2D Zernike Moments and the Polar Fourier Transform, while the T function is the Spherical Fourier Transform.

3 Matching

Firstly, the descriptors are normalized so as their absolute sum is equal to 1. Then, the matching is based on the Minkowski $L-1$ distance.

References

[1] D.Zarpalas, P.Daras, A.Axenopoulos, D.Tzovaras, and M.G.Strintzis: 3D Model Search and Retrieval Using the Spherical Trace Transform, *EURASIP Journal on Advances in Signal Processing*, Volume 2007

Shape retrieval of 3D watertight models using aMRG

Tony Tung Francis Schmitt
Telecom Paris, CNRS UMR 5141, France
{tony.tung, francis.schmitt}@enst.fr

Abstract

This paper presents an application of the augmented Multiresolution Reeb graph (aMRG) [3] for shape retrieval of 3D watertight models. The method is based on a Reeb graph construction which is a well-known topology based shape descriptor. With multiresolution property and additive geometrical and topological informations, aMRG has shown its efficiency to retrieve high quality 3D models [4]. The SHREC3D 2007 watertight database is composed of different classes of deformed models (sunglasses, humans, ants, etc.). We propose to evaluate the aMRG with new topological features [5] to classify the database. Our experiments show interesting results.

1. Introduction

Our paper proposes a scheme to retrieve similar shapes in a database of watertight 3D models using a Reeb graph based approach. As the database is composed of classes of deformed objects with same topology, the Reeb graph suits well to describe their shape. It is built using a function μ based on the mesh connectivity. The surface of the object is divided in regions according to the values of μ, and a node is associated to each region. The graph structure is then obtained by linking the nodes of the connected regions. Then a multiresolutional Reeb graph can be constructed hierarchically, based on a coarse-to-fine approach node merging [1]. Keeping advantage of the multiresolutional representation, the augmented multiresolution Reeb graph (aMRG) [3] is an enhanced Reeb graph which includes topological, geometrical and visual (color or texture) information in each graph node. Therefore similarity between two aMRGs can be computed to retrieve the most similar nodes. Its efficiency has been tested on high quality model database from museums [4]. Recently, the aMRG was adapted for full topology matching of human models in 3D video sequence [5]. We propose to evaluate the aMRG with the new topological features to classify the database. As shown in Figure 1, the method is able to retrieve similar 3D models. Three different runs are proposed for the Shape Retrieval Contest (SHREC) for watertight models 2007. The database contains 400 watertight models. The next section discusses work presents the aMRG. Section 3 presents experimental results.

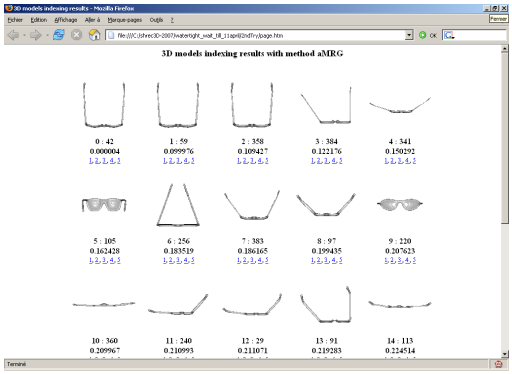

Figure 1. **Shape matching using aMRG.** The aMRG is a topology based descriptor. As is, it is powerful to retrieve similar models with various deformations and same topology. Similar sunglasses can be retrieved, even deformed. Here, the query is the model on top-left. Distance to query is shown under each compared models. The table presents the most similar models to the query.

2. Overview of the aMRG

According to the Morse theory, a continuous function defined on a closed surface characterizes the topology of the surface on its critical points. Therefore, a Reeb graph can be obtained assuming a continuous function μ calculated over the 3D object surface.

In the SHREC3D for watertight models 2007, models are defined by their surface and represented as 3D triangular meshes with vertices located in a Cartesian frame. We chose the function μ proposed in [1], which is defined as the integral of the geodesic distance $g(\mathbf{v}, \mathbf{p})$ from $\mathbf{v}$ to the other points $\mathbf{p}$ of the surface:

$$\mu(\mathbf{v}) = \int_{\mathbf{p} \in S} g(\mathbf{v}, \mathbf{p}) dS. \quad (1)$$

This function μ has the property to be invariant to rotations.

Its integral formulation provides a good stability to local noise on surface and gives a measure of the eccentricity of the object surface points. A point with a great value of μ is far from the center of the object. A point with a minimal value of μ is close to the center of the object. Therefore μ is normalized to $\mu_N = \frac{\mu_{max}-\mu_{min}}{\mu_{max}}$, so that values of μ keep an information on the distance to the object center. The corresponding Reeb graph is then obtained by iteratively partitioning the object surface into regular intervals of μ_N values and by linking connected regions. For each interval, a node is associated to each different set of connected triangles.

To construct a Reeb graph of R levels of resolution, μ_N is subdivided into 2^R intervals from which the object surface is partitioned at the highest level of resolution. Afterwards, using a hierarchical procedure, Reeb graphs of lower resolution levels are obtained by merging intervals by pairs [1]. The multiresolutional aspect results from the dichotomic discretization of the function values and from the hierarchical collection of Reeb graphs defined at each resolution (cf. Figure 2).

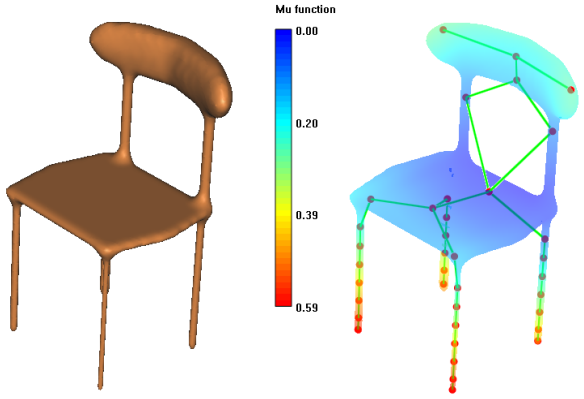

Figure 2. **Multiresolution Reeb graph.** (a) shows a 3D model. (b) shows values of function μ on the surface, with Reeb graphs at resolution $r = 4$. The graph structure contains topological and geometrical information.

The original approach, mainly based on the 3D object topology, is not accurate enough to obtain satisfying matching. Therefore in [3, 4], the multiresolution Reeb graph has been augmented by merging global and local geometric properties and visual properties extracted from the object surface region $S(m)$ associated to each node m. Topological aspects of the graph matching procedure was extended, and similarity calculation with the new features was adapted. The different features are:

- a statistic measure of the extent of $S(m)$,
- a statistic of the Koenderink shape index for local curvature estimation on $S(m)$,
- a statistic of the orientation of the triangle normals associated to $S(m)$,
- a statistic of the texture/color mapped on $S(m)$ (not used here).

The choice of these attributes has been guided by the litterature on 3D content-based retrieval [2]. The result is a flexible multiresolution and multicriteria 3D shape descriptor including merged topological, geometrical and colorimetric properties.

In order to obtain a better control of the node matching, graph topology was exploited in [5]. Topological features can be deduced by the edge orientations given by μ values. In addition, multiresolution gives valuable information to characterize the global shape of models. The shape description is intuitive and completely topological. Its efficiency has been tested as described in the next section.

3. Experiments

aMRG calculations were performed on a laptop with Pentium(R) M processor 1.60 GHz and RAM 512 Mo. aMRG were computed with resolution up to $r = 5$. The computation time depends on the number of vertices, and the complexity of the model. For example, aMRG of watertight model 1.off which contains 15000 vertices was computed in ~ 20 s. And aMRG of model 3.off which contains 6833 vertices was computed in ~ 12 s.

The calculation of the function μ remains the most time consuming, even with the Dijkstra coding scheme of $O(NlogN)$ complexity on N vertices. Our experiments have pointed out the importance of the choice of the function μ. The invariance to rotation properties was necessary for the watertight model database.

For the SHREC3D for watertight models 2007, three runs were proposed with different aMRG resolution:

- Run1: similarity are computed till resolution $r = 3$
- Run2: similarity are computed till resolution $r = 4$
- Run3: similarity are computed till resolution $r = 5$

Results are slightly similar. The main difference is the computation time which is a little bit longer for higher resolution. The matching procedure based only on topological information is well adapted for non oriented object. Using geometrical information can sharpen the retrieval in some case. Computation time for retrieval of a query on the watertight model database of 400 models at $r = 5$ is ~ 6 s. Computation time for a query at $r = 3$ is ~ 3 s. Figure 3 presents a query results on a monster model. aMRG resolution is 3. Similar models are retrieved, even with strong deformations. Figure 4 presents a query results on an ant model. aMRG resolution is 3. The retrieval seems efficient.

Figure 5 presents a query results on a chair model. aMRG resolution is 4. Chairs and tables are retrieved. Figure 6 presents a query results on a teddybear model. aMRG resolution is 4. Teddybears are well retrieved, even models with arms or legs missing.

4. Conclusion

We have presented an application of the augmented Multiresolution Reeb graph for the SHREC3D of watertight models 2007. As the watertight models have remarkable topology, we obtained interesting results with our approach. A full topological approach was used for the graph matching procedure. Three runs at different aMRG resolution were proposed. The weight of the different attributes can still be optimized to refine the similarity evaluation. In general, it is difficult to classify deformed models as a strong deformed model can be considered as close to another class.

References

[1] M. Hilaga, Y. Shinagawa, T. Kohmura, and T. L. Kunii. Topology matching for fully automatic similarity estimation of 3d shapes. *ACM SIGGRAPH*, pages 203–212, 2001. 1, 2

[2] J. Tangelder and R.C.Veltkamp. A survey of content based 3d shape retrieval methods. *IEEE International Conference on Shape Modeling and Applications (SMI)*, pages 145–156, 2004. 2

[3] T. Tung and F. Schmitt. 3d shape matching using reeb graph. *IEEE International Conference on Shape Modeling and Applications (SMI)*, pages 157–166, 2004. 1, 2

[4] T. Tung and F. Schmitt. The augmented multiresolution reeb graph approach for content-based retrieval of 3d shapes. *International Journal of Shape Modeling (IJSM)*, pages 91–120, 2005. 1, 2

[5] T. Tung, F. Schmitt, and T.Matsuyama. Topology matching for 3d video compression. *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, 2007. 1, 2

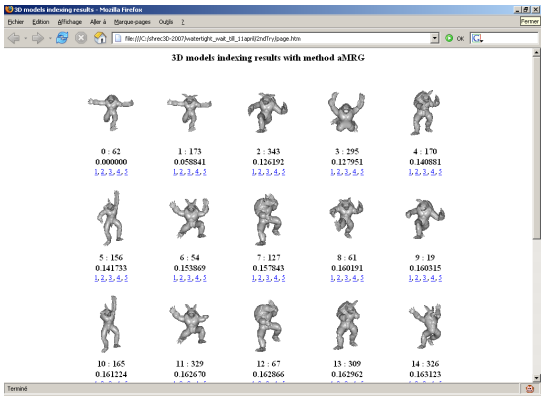

Figure 3. **Query 62 with run 1.** aMRG resolution is 3. Similar models are retrieved, even with strong deformations.

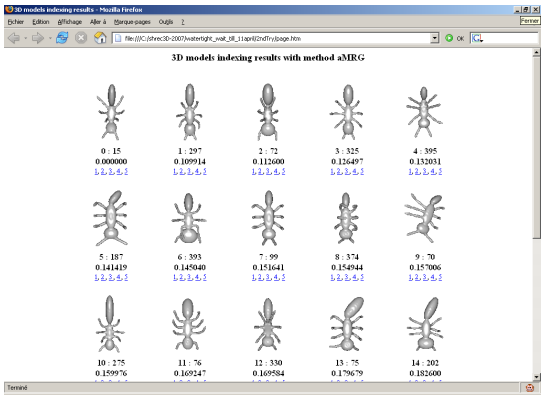

Figure 4. **Query 15 with run 1.** aMRG resolution is 3. Ants models from the database are well retrieved.

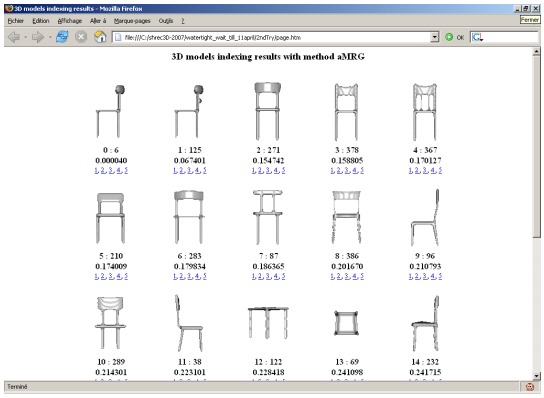

Figure 5. **Query 6 with run 2.** aMRG resolution is 4. Chairs and tables are retrieved.

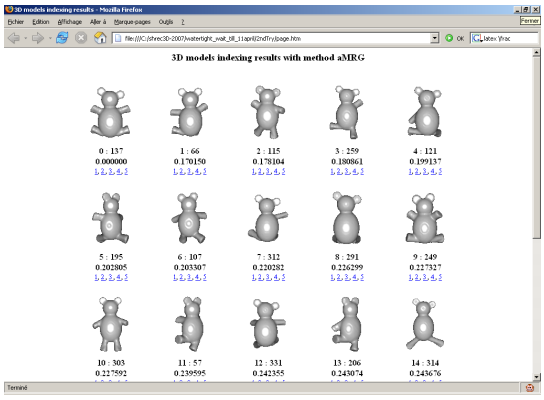

Figure 6. **Query 137 with run 2.** aMRG resolution is 4. Teddybears are well retrieved, even models with arms or legs missing.