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

Thibault Napoléon
Telecom Paris
CNRS UMR 5141
75013 Paris, France
napoleon@enst.fr

Tomasz Adamek
CDVP
Dublin City University
Dublin 9, Ireland
adamekt@eeng.dcu.ie

Francis Schmitt
Telecom Paris
CNRS UMR 5141
75013 Paris, France
schmitt@enst.fr

Noel E. O'Connor
CDVP
Dublin City University
Dublin 9, Ireland
oconnorn@eeng.dcu.ie

Abstract

We describe in this paper two methods for 3D shape indexing and retrieval that we apply on two data collections of the SHREC - SHape Retrieval Contest 2007: Watertight models and 3D CAD models. Both methods are based on a set of 2D multi-views after a pose and scale normalization of the models using PCA and the enclosing sphere. In all views we extract the models silhouettes and compare them pairwise. In the first method the similitude measure is obtained by integrating on the pairs of views the difference between the areas of the silhouettes union and the silhouettes intersection. In the second method we consider the external contour of the silhouettes, extract their convexities and concavities at different scale levels and build a multiscale representation. The pairs of contours are then compared by elastic matching achieved by using dynamic programming. Comparisons of the two methods are shown with their respective strengths and weaknesses.

1 Introduction

We proposed two methods for the 3D Shape Retrieval Contest 2007. Each one is based on a multi-view approach which keeps 3D model coherence by considering simultaneously a set of 2D images in specific view directions. The various silhouettes of a model being strongly correlated, using a set of them help to better discriminate one model among others.

First of all, we have to get a robust normalization of the model pose and model scale in order to remain invariant to various geometrical transformations (translation, rotation, scaling). We used a Principal Continuous Component Analysis [4][5] and the smallest enclosing sphere [3] to solve these problems.

The first method is based on silhouettes intersection. We capture a set of views of a model and we extract its silhouette in 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 models is defined as the sum of the distances between their two sets of silhouettes.

The second approach is based on a multiscale representation of the external closed contour of non rigid 2D shapes presented in [1]. We capture a set of views of a model and for each view we extract and normalize the external border of the silhouette, and we buid its multi-scale shape representation where for each contour point we store information on the convexities and concavities at different scale levels. We then search the optimal elastic match between each pair of silhouettes by minimizing the distance between matched contour points and we integrate the distance over the silhouettes pairs.

Section 2 presents the normalization method for the 3D models. Section 3 describes the intersection methods and section 4 presents the contour convexities and concavities approach. Experimental results are shown in section 5.

2 Model normalization

Before comparing 3D models we need to proceed to a robust normalization of their pose and scale in order to remain invariant to various geometrical transformations (translation, rotation, scaling). For the center and the scale, we use the smallest enclosing sphere *S* [3] (see Figure 1). The normalization then becomes:

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

where $d(S)$ is the diameter of *S* and $c_i(S),\ i = x,\ y,\ z$ are the i-th coordinates of its centre. The use of the smallest enclosing sphere has several advantages: it is fast to calculate, it allows maximizing the model size inside the unit sphere

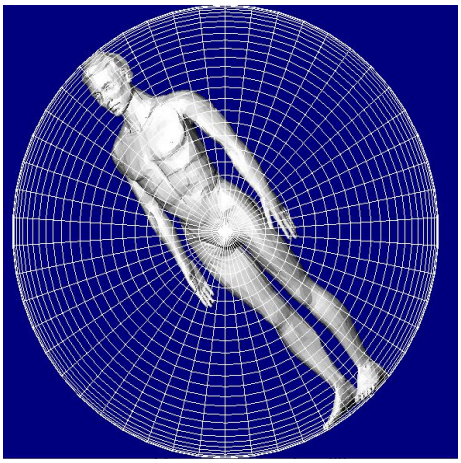

Figure 1. *Smallest enclosing sphere.*

and then its silhouette size 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).

For the normalization of the model pose we use the Continuous Principal Component Analysis [4][5] which defines and orientates the three principal axis of a model in a robust way and at a very reasonable computation cost.

3 Intersection descriptor

3.1 Signature extraction

The various silhouettes of a model being strongly correlated, using a set of them help to better discriminate one model among others [2]. For this, we can use any set of view directions regularly distributed in space. We consider here simply the three orthogonal views along the oriented principal axis with parallel projections. A higher number of views could be used but we limit it to 3 to keep a reduced size for the shape descriptors.

We choose an image size of 256x256 for each silhouette. This resolution gives a good tradeoff between precision and computation time. To keep the maximum information from a silhouette, the simplest way is just to keep its image (see Figure 2). We will then compare two silhouettes by superposing them and comparing their intersection with their union. A silhouette being a binary image we can stored it in a lossless compression format fast to read and to decode when comparing silhouettes. The signature of a model is then simply constituted by the three compressed silhouettes corresponding to the three oriented principal directions.

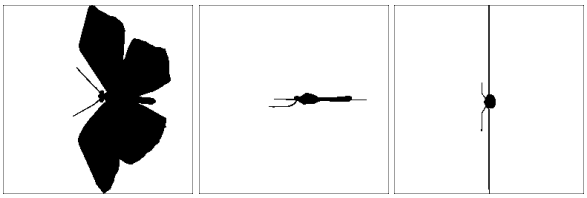

Figure 2. *Three silhouettes of a model.*

3.2 Signature matching

The distance between two models 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 (see Figure 3). This measure can be computed very efficiently directly on the files compressed with a simple run length encoding. The distance computation between two models is then straightforward and fast. To answer a query we just measure its distance to every database models and sort the list accordingly (see Figure 4).

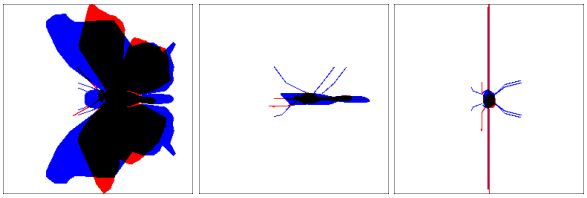

Figure 3. *Intersections of two models. In black their common parts, in blue the parts of the first model and in red the parts of the second one*

4 Contour convexities and concavities descriptor

4.1 Signature extraction

We use 256x256 silhouettes for both the CAD and Watertight models tracks. We also test 64x64 silhouettes on the Watertight models: this smaller image resolution allows a reduction of the descriptor size and of the computation time at the price of a stronger sampling noise leading to a lower retrieval precision. The descriptor of a silhouette contour C is obtained by normalizing the contour length

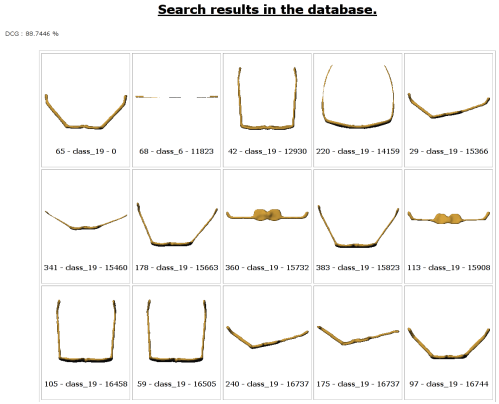

Figure 4. *15 first results for shape retrieval using the intersection method on the Watertight model database. The query 65.off is on the top left.*

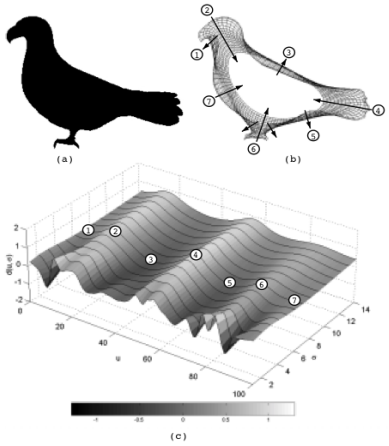

Figure 5. *Example of extracting the MCC shape representation: (a)-original shape image, (b)-filtered versions of the original contour at different scale levels, (c)-final MCC representation for 100 contour points at 14 scale levels.*

with 100 sampled contour points and by extracting convexity/concavity information at each sampled contour point and at 10 scale levels [1]. 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 contour point u at scale level σ. The simplified boundary contours at different scale levels are obtained via a curve evolution process. It should be noted that we use the same number of sample contour points at each scale. Let the contour C be parameterized by arclength 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 bandwidth $\sigma \in \{1, 2...\sigma_{max}\}$. The resulting contour C_σ becomes smoother with increasing σ value, until finally the contour becomes convex (see Figure 5).

We propose a very simple measure for the convexity/concavity of the curve. It 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 $d(u, \sigma)$ of the contour between two consecutive scale levels at point $p(u, \sigma)$ can be defined as the Euclidian distance between positions of $p(u, \sigma)$ and $p(u, \sigma - 1)$.

4.2 Signature matching

When comparing two contours A and B, it is necessary to examine the distance between each sampled 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 Figure 6 where the contours feature vectors are illustrated as grey levels along each axis).

The three silhouettes of each set being sorted according to the three principal axis, the distance between two models is just defined as the sum of the distances of the three pairs of silhouettes, one pair per axis (see Figure 7).

5 Experimental results

5.1 Experimental results for Watertight track

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

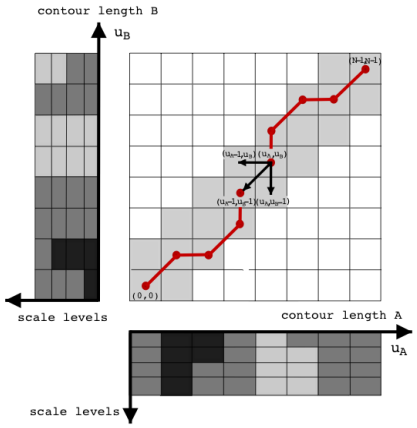

Figure 6. *Matching of two MCC representations by using dynamic programming.*

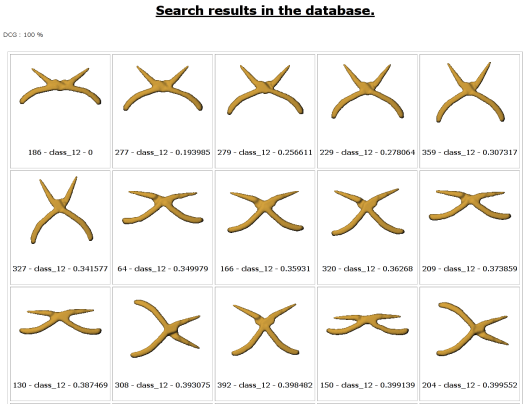

Figure 7. *15 first results for shape retrieval using the contour convexities and concavities method on the Watertight model database. The query 186.off is on the top left.*

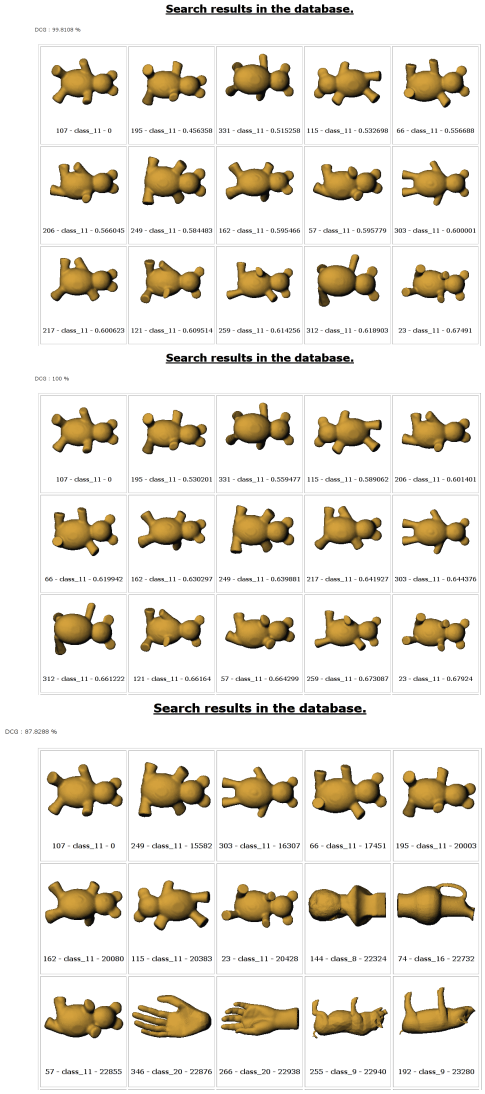

Figure 8. *15 first results for the Watertight query 107.off. Top: run 1, Middle: run 2, Bottom: run 3.*

• 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.

The contour convexities and concavities descriptor provides better results than the multi-view intersection descriptor, see Figure 8 (top and bottom). The two different image resolutions produce practically the same results, see Figure 8 (middle). We present in Table 1 the classwise DCGs for each run. We observe that run 1 and run 2 perform perfectly results for the *plier* class. For the contour convexities and concavities descriptor the worst DCGs are obtained for *spring* and *vase* classes. These two classes contains models with heterogeneous shapes. For the multi-view intersection descriptor the worst DCG is obtained for *armadillo* and *octopus*.

5.2 Experimental results for CAD track

We propose two runs for the SHREC'07 CAD models track:

• Run 1: The contour convexities and concavities de-

Class	DCG Run 1	DCG Run 2	DCG Run 3
airplane	92.9	93.6	66.5
ant	91	89.7	46.7
armadillo	82.4	80.3	46.3
bearing	86.1	80.1	80.8
bird	87.1	89.7	61.6
buste	87	85.9	66.3
chair	92	93.4	87.7
cup	77.2	77.1	73.7
fish	94.3	93.2	76.9
four leg	90.4	85.5	68.1
glasses	90.8	91	92.5
hand	81.8	77.2	56.6
human	82.8	77.4	68
mechanic	89.1	91	85.6
octopus	79.2	74.2	41.2
plier	100	100	97
spring	55.3	54.5	50.6
table	77.7	81.6	81.9
teddy	95.6	96	80
vase	52.2	53.7	56.3

Table 1. *Classwise DCGs performance for the three runs on the Watertight models.*

scriptor, with 3 silhouettes aligned with the principal axis and a resolution of 256x256 pixels for each silhouettes.

• Run 2: The multi-view intersection descriptor, with 3 silhouettes aligned with the principal axis and a resolution of 256x256 pixels for each silhouettes.

The contour convexities and concavities descriptor provides again better results than the multi-view intersection descriptor (see Figure 9). The first method is more robust against small variations of the shape and against the mirrored silhouettes problem. But the computation time per query model is very different between the two methods: with the contour convexities and concavities descriptor the CPU time per query is $\sim$ 20 s and with the multi-view intersection descriptor $\sim$ 0,06 s.

6 Conclusion

We have tested two different methods on the SHREC'07 contest. We observe that we obtain the best results with the convexities/concavities descriptor. We notice that the intersection method is not very robust with small deformations of the models. The contour convexities and concavities method needs much more computation time, but this weakness could be strongly reduced by optimizing the source code.

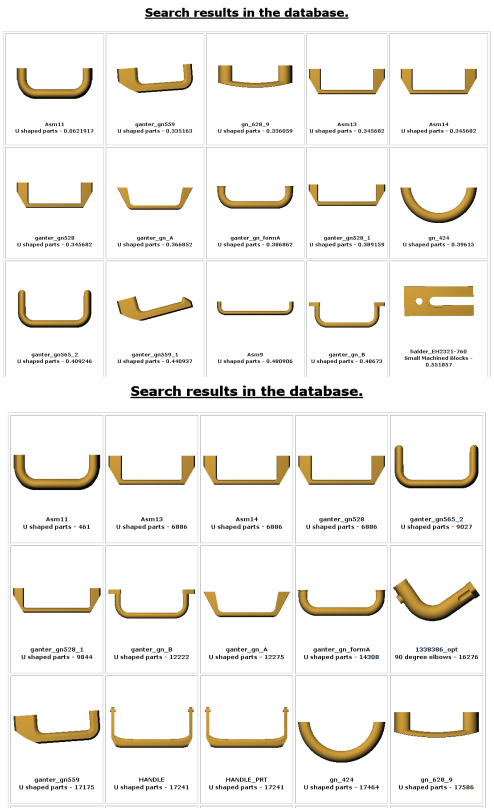

Figure 9. *15 first results for the CAD query 40.stl. Top: run 1, Bottom: run 2.*

7 Acknowledgments

This research was partly supported by the European Commission under contract FP6-027026-K-SPACE.

References

[1] 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*, 14(5):742–753, 2004.

[2] D.-Y. Chen, X.-P. Tian, Y.-T. Shen, and M. Ouhyoung. On visual similarity based 3D model retrieval. *Comput. Graph. Forum*, 22(3):223–232, 2003.

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

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

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