Eurographics Symposium on Point-Based Graphics (2004)
M. Alexa, S. Rusinkiewicz, (Editors)

Shape Segmentation and Matching from Noisy Point Clouds

Tamal K. Dey[1†] and Joachim Giesen[2‡] and Samrat Goswami[3§]

[1,3]The Ohio State U. [2] ETH, Zürich

Abstract

We present the implementation results of a shape segmentation technique and an associated shape matching method whose input is a point sample from the shape. The sample is allowed to be noisy in the sense that they may scatter around the boundary of the shape instead of lying exactly on it. The algorithm is simple and mostly combinatorial in that it builds a single data structure, the Delaunay triangulation of the point set, and groups the tetrahedra to form the segments. A small set of weighted points are derived from the segments which are used as signatures to match shapes. Experimental results establish the effectiveness of the method in practice.

Categories and Subject Descriptors (according to ACM CCS): I.3.3 [Computer Graphics]: Modeling3D Shape Matching, Point-Based Graphics.

1. Introduction

A spurt of research activities in extracting shape information from its point sample have ensued in recent years because of the flexibility offered by points as input [ABCO*01, ACK01, PKKG03]. In this paper we address the problem of segmenting a three dimensional shape into identifiable 'features' from point clouds and then using them for shape matching. In particular, we focus on noisy point clouds where the sample points are allowed to scatter around the shape boundary.

Many applications including object recognition, classification, matching, tracking need to solve the problem of shape segmentation, see for example [AG96, BM02, JH99, KT03, TC92]. Different structures such as shock graphs [SKK01], medial axis [LK01], Reeb graphs [HSKK01], mesh partition [KT03] and shape distributions [OFCD01] have been proposed for the problem. All of these techniques require an input mesh and thus are not applicable to point cloud data unless one reconstructs a surface out of it. Surface reconstruction is often a costly step and more importantly is not yet an established robust procedure for noisy data. We bypass an explicit surface reconstruction step. Our algorithm computes a Delaunay triangulation of the point sample and determines the tetrahedra that have to be clustered together to form segments of the shape interior. In other words, instead of building multiple data structures, such as an approximate surface mesh and then other derived structures like medial axis and Reeb graphs, the algorithm acts only on a single data structure, the Delaunay triangulation of the point sample.

The segmentation algorithm has two phases. The first phase pre-processes the point set using the idea of Dey and Goswami [DG04]. It filters Delaunay tetrahedra whose union approximates the shape interior. The second phase further partitions them into segments using the concepts developed in [DGG03]. It mimicks a topological segmentation of the continuous shape. This segmentation partitions the space using Morse structures defined by a distance function.

We exploit this Morse theoretic segmentation for matching shapes based on the principle that similar shapes have similar such segmentation. We derive a signature of a shape from its segmentation and match these signatures. From all segments we select only a few significant segments and represent each one with a weighted point where the weight is the volume of the segment. Then, the shape matching problem boils down to matching two small weighted point sets instead of matching large point sets derived from the shape boundaries [HKR93]. Further, representative points can be precomputed and kept in a database thus reducing both space and time requirements for matching. We carry out these steps

† email: tamaldey@cis.ohio-state.edu
‡ email: giesen@inf.ethz.ch
§ email: goswami@cis.ohio-state.edu

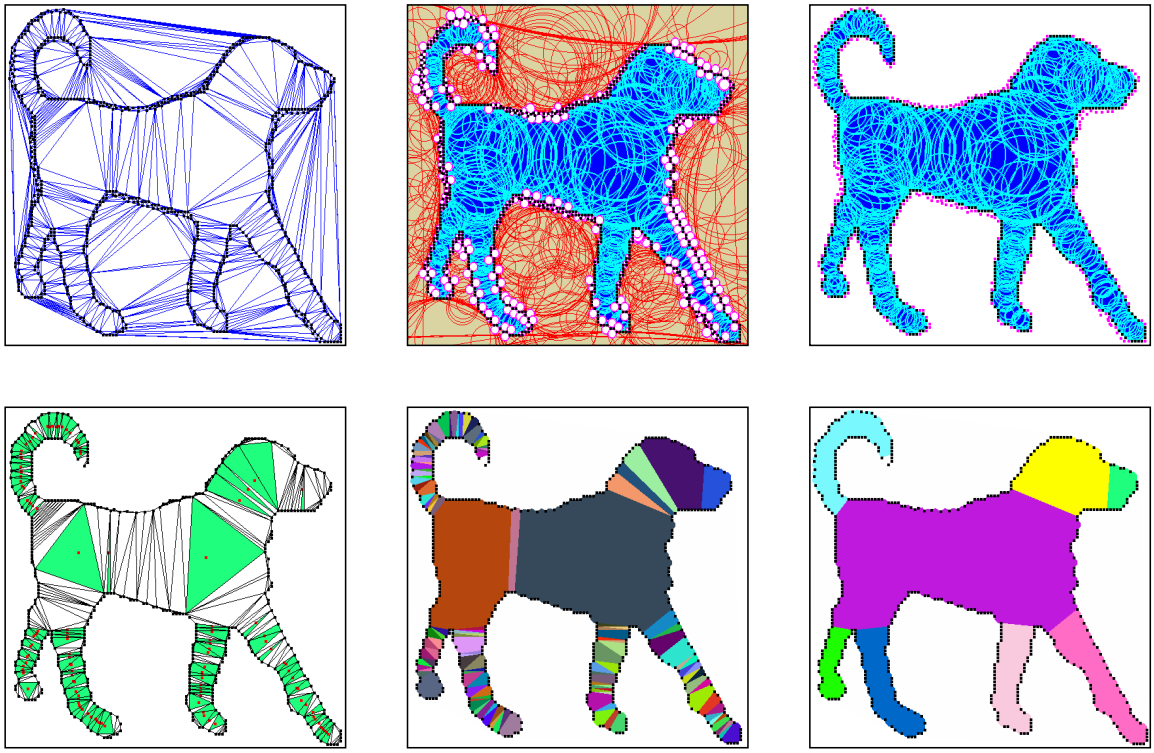


Figure 1: *Steps of the Algorithm: Top row: Delaunay triangulation of a noisy point sample (left), Delaunay balls where small balls are shaded white (middle), union of inner big Delaunay balls (right). Bottom row: inner simplices containing maxima Voronoi vertices are shaded (left), segmentation by stable manifolds of the maxima (middle), segments after merging (right).*

so that the entire matching process remains invariant to rotation, translation, mirroring and scaling.

2. Point processing

Let P be a point sample, possibly noisy, derived from the boundary $\partial\Sigma$ of a shape Σ embedded in $\mathbb{R}^3$. Assume that the surface $\partial\Sigma$ is compact and smooth. We will denote the Delaunay triangulation of P by $\mathrm{Del}\,P$ and its dual, the Voronoi diagram, by $\mathrm{Vor}\,P$. A Voronoi cell for a point $p \in P$ is denoted V_p. Each Voronoi vertex is the center of a ball that circumscribes a dual Delaunay tetrahedron. All such balls are called *Delaunay balls*.

In the point processing phase, the algorithm filters Delaunay tetrahedra from $\mathrm{Del}\,P$ whose union approximates Σ. To understand the rationale behind the method, first assume that P is noise-free. For a point $p \in P$, the inner (outer) pole of p is the Voronoi vertex lying inside (outside respectively) Σ and is farthest from p among all other vertices of V_p. The Delaunay balls centering the poles are called *polar balls*. It is known by a result of Amenta, Choi and Kolluri [ACK01] that the polar balls centering the inner (outer) poles approximate Σ (complement of Σ respectively). It is easy to compute the poles from the Voronoi diagram $\mathrm{Vor}\,P$. However, in absence of Σ, one needs a mechanism to separate the inner poles from the outer ones in order to obtain an approximation of Σ. Here the following *intersection depth* property comes to the rescue. Two inner (outer) polar balls cricumscribing adjacent tetrahedra intersect deeply while an inner and an outer polar ball intersect only in a shallow manner. The depth of intersection is measured by the angle at which the boundaries of the balls intersect. A depth-first search starting from an unbounded polar ball, which is guaranteed to be outer, can collect all of them if the walk moves from an outer polar ball to an adjacent polar ball only if they intersect deeply.

The above algorithm does not work when P is noisy. A main reason for this is that the intersections between polar balls do not follow the intersection depth property. We circumvent this difficulty by observing that, even under noise, where the points are not perturbed more than a small fraction of the local feature size, some of the Delaunay balls behave like polar balls as in the noise-free case. We identify them by their relative size compared to the nearest neighbor distances. Let B be a Delaunay ball cricumscribing four points $p_i, i = 1,..,4$. We say B is *big* if the radius of B is more than $\rho > 0$ times bigger than the average nearest neighbor distances of any of p_i. The average is taken over three nearest

neighbors and ρ is 0.5 in our implementation. After identifying the big Delaunay balls, we partition them using the intersection depth property as in the noise-free case. A Delaunay ball is *inner* (*outer*) if its center lies inside (outside respectively) Σ. It is proved by Dey and Goswami [DG04] that, under a reasonable noise model, big Delaunay balls satisfy the intersection depth property and the union of inner big Delaunay balls approximate Σ. Therefore, a depth-first walk starting from an unbounded outer Delaunay ball collects all big outer Delaunay balls. The rest of the big Delaunay balls are inner. The tetrahedra circumscribed by these inner big Delaunay balls approximate Σ. The top row of Figure 1 illustrates this phase for a two dimensional shape.

3. Segmentation

In this phase we use the technique proposed by Dey, Giesen and Goswami [DGG03] for segmentation. We briefly sketch the method here for completeness.

3.1. Continuous shape

First we describe a segmentation of Σ using the distance function $h : \mathbb{R}^3 \to \mathbb{R}$ where

$$h(x) = \inf_{p \in \partial\Sigma} \|p - x\|^2 \text{ for all } x \in \mathbb{R}^3.$$

We would define a vector field $v : \mathbb{R}^3 \to \mathbb{R}^3$ that assigns to every point $x \in \mathbb{R}^3$ the direction in which h increases the most. If h is smooth at x then $v(x)$ coincides with the normalized gradient $\nabla h(x)/\|\nabla h(x)\|$. In our case h is smooth everywhere except at the medial axis of Σ. However, it turns out that one can still define the direction of the steepest ascent of h at any point on the medial axis [DGG03]. This allows to define a vector field where $v(x)$ is a unit vector pointed in the direction of the steepest ascent of h at x. The vector field v induces a flow $\phi : [0,\infty) \times \mathbb{R}^3 \to \mathbb{R}^3$ such that the right derivative at every point $x \in \mathbb{R}^3$ matches with the vector, see Grove [Gro93].

Given $x \in \mathbb{R}^3$ and an induced flow ϕ, the curve $\phi_x : [0,\infty) \to \mathbb{R}^3$, $t \mapsto \phi(t,x)$ is called the *orbit* of x. A point x is a *fixpoint* of ϕ if $\phi(t,x) = x$ for all $t \geq 0$. Basically, the orbit of a point is the curve it will follow if it were let move along the steepest ascent of h. The fixpoints of ϕ are the critical points of h. The critical points where h increases in all directions are the minima of h. These are the points of $\partial\Sigma$. The critical points where h decreases in all directions are the maxima of h. The rest of the critical points are *saddles*. In Figure 2, the points a and c are maxima, b is a saddle.

The stable manifold $S(x)$ of a critical point x is the set of all points that flow into x, i.e.,

$$S(x) = \{y \in \mathbb{R}^3 : \lim_{t \to \infty} \phi_y(t) = x\}.$$

The stable manifolds of all critical points partition $\mathbb{R}^3$. In particular, if we take the closure of the stable manifolds of the maxima, the stable manifolds of saddles and minima constitute its boundary. We segment Σ with the closure of the stable manifolds of the maxima of h. Figure 2 shows two segments produced by this segmentation.

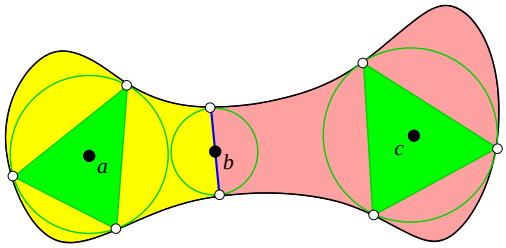


Figure 2: *The stable manifold of the saddle b separates those of the maxima a and c producing two segments.*

3.2. Discretized shape

Now we mimic the above segmentation of Σ using the discrete sample P of $\partial\Sigma$. The distance function is defined with respect to the sample points as follows:

$$h(x) = \min_{p \in P} \|x - p\| \text{ for all } x \in \mathbb{R}^3.$$

It turns out that the critical points for this function are the points where the Delaunay and their dual Voronoi objects intersect. This has been observed before [EFL98]. The sample points are minima. They are the intersection of themselves and their dual Voronoi cells. The Voronoi vertices that are contained in their dual Delaunay tetrahedra are the maxima. There are two types of saddles in $\mathbb{R}^3$. The points where a Delaunay edge intersects its dual Voronoi facet are saddle points of type 1 and the points where a Delaunay triangle intersects its dual Voronoi edge are saddles of type 2. We define segments as the closure of the stable manifolds of the maxima mimicking the continuous case. Each of such a segment is bounded by the stable manifolds of the saddle points of type 1 and 2 and the minima. The stable manifolds of the saddle points are a subset of Delaunay objects (edges) in 2D. The same is not true in 3D. Although they can be computed exactly [GJ02], we avoid costly numerical computations by approximating the stable manifolds of the maxima with only Delaunay objects using the Delaunay flow idea of Edelsbrunner, Facello and Liang [EFL98].

Let σ_1, σ_2 be two Delaunay tetrahedra sharing a triangle t. We say σ_1 *flows into* σ_2 denoted by $\sigma_1 < \sigma_2$ if σ_1 and its dual Voronoi vertex lie on the opposite sides of the plane of t. For an illustration see the triangles surrounding the big green triangle in the hip of the DOG in the lower left picture of Figure 1. All these three triangles flow into the green triangle containing a maximum. It follows from the definition that if $\sigma_1 < \sigma_2$, then the radius of the Delaunay ball

of σ_1 is smaller than the radius of the Delaunay ball of σ_2. Thus, the transitive closure $<^*$ of $<$ is acyclic. For a maximum x, one can collect all tetrahedra that are flowing to x by the transitive closure of $<^*$. This is the method which was proposed to compute the pockets in two dimensional shapes by Edelsbrunner et al. [EFL98]. In three dimensions we face a difficulty because a Delaunay tetrahedron may flow into two other Delaunay tetrahedra. Therefore, the approximated stable manifolds of maxima computed by the above method may not be disjoint. To overcome this problem we change the flow relation $<$ to $\tilde{<}$ as follows. Define the *strength* of a tetrahedron σ as the largest value of the distance function among all maxima that it flows into. We say $\sigma_1 \tilde{<} \sigma_2$ if (i) $\sigma_1 < \sigma_2$ and (ii) there is no other tetrahedron σ_3 with $\sigma_1 < \sigma_3$ and the strength of σ_3 is larger than σ_2. Now we define a segment $G(x)$ for a maximum x as

$$G(x) = \bigcup_{\sigma' \tilde{<}^* \sigma} \sigma'$$

where σ is the tetrahedron containing x. One can turn this definition into an algorithm very easily. First, we determine the maxima Voronoi vertices. They are the Voronoi vertices contained in their dual Delaunay tetrahedra. Then, we sort these maxima in decreasing order of their h-values. This value is given by the radius of the Delaunay ball circumscribing the dual tetrahedron of the maximum Voronoi vertex. We process the maxima in this sorted order and for each maximum x collect the tetrahedra that flow into it. This is done by a depth first search starting from the tetrahedron dual to x and then including a tetrahedron in the collection if it flows into one of the tetrahedra already in the collection. Middle picture in the lower row of Figure 1 shows the segmentation obtained by this process for a two dimensional shape.

3.3. Merging

As one can observe from Figure 1, the segments obtained by the above algorithm may be too fine because of sampling artifacts. Many maxima may be introduced due to discretization. We merge small segments to coarsen the segmentation. A segment is judged *prominent* if the h-value at the maximum is significantly larger than the h-value at the circumcenters of the triangles on the segment boundary. In other words, prominent segments are 'peaky' when seen in the graph of the distance function h. We merge all segments that are not prominent with some adjacent segments. Precisely, we say two segments $G(x_1)$ and $G(x_2)$ are ρ-mergable if they share a triangle t and the circumradii of the tetrahedra containing x_1 and x_2 are no more than ρ times the circumradius of t. In the implementation we merge any two 1.5-mergable segments.

Final segments of a shape are the ones obtained after the merging process. The rightmost picture in the lower row of Figure 1 shows the segmentation after merging. In Figure 3 we show the segmentation of some 3D shapes from their point samples.

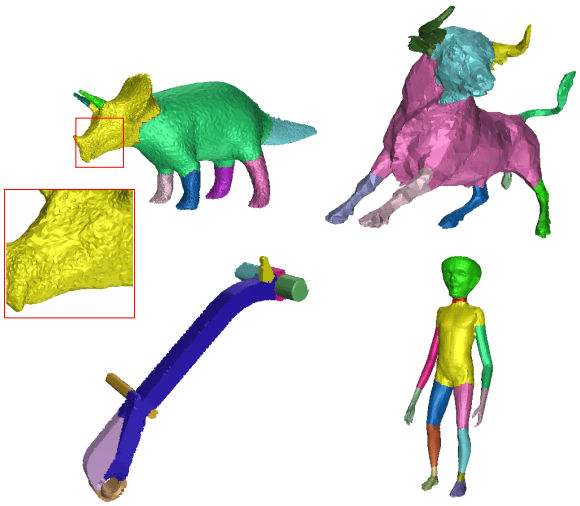


Figure 3: *Segmented shapes: Segments are shown with different colors. The* TRICERATOPS *and the* BULL *point samples are noisy.*

4. Matching

For shape matching we use our segmentation scheme. Based on the principle that similar shapes have a similar segmentation, we generate a signature for a shape from its segmentation and then match it against other signatures. The signatures are a small set of weighted points that represent the segments.

4.1. Signature

Let $G_{P,\Sigma}$ denote the set of segments computed from a point sample P of a shape Σ. To simplify notations we use G_Σ for $G_{P,\Sigma}$. By definition a segment $g \in G_\Sigma$ is a collection of Delaunay tetrahedra. For a Delaunay simplex σ let c_σ and v_σ denote the centroid and volume of σ, respectively. The *representative point* g^* of a segment g and its weight $\hat{g}$ are defined as

$$\hat{g} = \Sigma_{\sigma \in g} v_\sigma, \quad g^* = \frac{\Sigma_{\sigma \in g} (c_\sigma \cdot v_\sigma)}{\hat{g}}.$$

That is, the weight of g is its volume and its representative point is the weighted average of the centroids of all $\sigma \in g$, weight being the volume of each simplex. Given a segmentation G_Σ of a shape Σ, the signature $sign(\Sigma)$ is defined as a set of weighted points as follows.

$$sign(\Sigma) = \{(g^*, \hat{g}) \mid g \in G_\Sigma\}.$$

4.2. Scoring

The amount of similarity between two shapes is measured by first scaling them with bounding boxes and then scoring

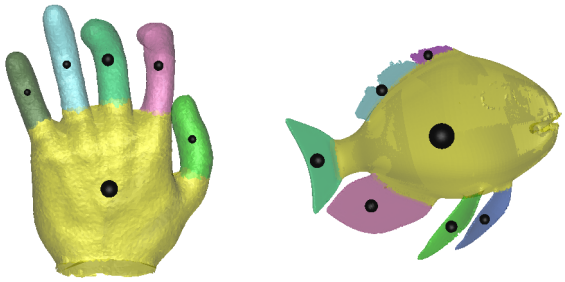
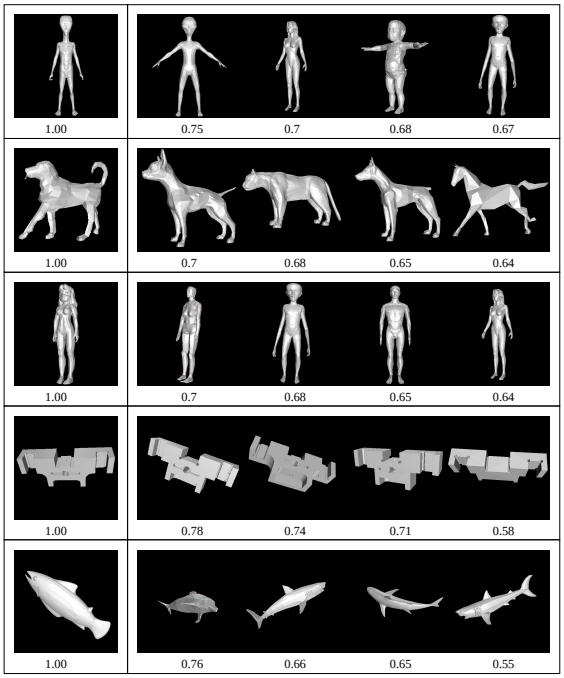


Figure 4: *Representative points enlarged according to the volumes of the corresponding segments.*

the similarity between their signatures. In order to score the similarity between two signatures $sign(\Sigma_1)$ and $sign(\Sigma_2)$, we need to align them first.

Let g^*, h^* be the representative points in $sign(\Sigma_1)$ and $sign(\Sigma_2)$, respectively, with maximum weights. We first translate $sign(\Sigma_2)$ so that g^*, h^* coincide. Then an alignment is obtained by rotating $sign(\Sigma_2)$ so that a line segment between h^* and another point of $sign(\Sigma_2)$ aligns with a line segment between g^* and another point in $sign(\Sigma_1)$. Certainly, there are $\Theta(mn)$ alignments possible where $|sign(\Sigma_1)| = m$ and $|sign(\Sigma_2)| = n$. Since m, n are typically small (less than ten), checking all alignments can be done fast.

For each alignment we compute a score based on the matching of the weighted points. Both a similarity measure (positive) and a dissimilarity measure (negative) are taken into account while computing the score. The maximum of all the scores is taken to be the amount of similarity and corresponding transformations give the best alignment.



	CAD	Pig	Tiger	Dog	Boy	Alien
#pts	11K	37K	44K	55K	91K	120K
sec.	19.8	23.2	39.9	47.5	104.9	145.5

Figure 5: *Matching result: Models with five best scores are shown for each query in a row. Timings for segmentation in seconds for some models are shown.*

5. Results and comparisons

We used the robust and fast Delaunay triangulation code of CGAL to implement the segmentation and the matching algorithms. The experiments were done on 2.8 Ghz Pentium 4 machine with 1 GB RAM. As the examples show, the segmentation mostly respects the so called features. We created a database of signatures from approximately 300 point cloud data mostly collected from different web-sites and also created from 3D models. Some example matching results and segmentation timings are shown in Figure 5. Matching of a query shape over the entire database took less than a second. Figure 6 shows the similarity matrix for our method on approximately 200 shapes divided into 17 categories. Only the five best scores for each query are shaded according to their values.

We find it difficult to compare our technique with other matching algorithms as all of them assume a surface mesh as an input. One novelty of our algorithm is that we do not build any extra data structure other than the Delaunay triangulation of the input point set. Also, there is no costly computation such as approximating geodesic distances as by Hilaga et al. [HSKK01]. The shape distribution method of Osada et al. [OFCD01] assumes an input mesh. For comparisons we adapted it to point clouds as follows. We use the D2 metric, that is, we compute distances between pairs of randomly selected points. Random selection took care of point density to have a fair comparison. This technique is quite effective for most of the models, however sometimes it creates some anomalies. Figure 7 shows one such example for which our method works properly.

6. Conclusions

In this paper we showed that a surface reconstruction technique for noisy point clouds can be combined with a topological technique for segmentation to obtain a robust method for feature identification of shapes from their point samples possibly corrupted with noise. The entire method is direct and simple in that it deals with a single data structure,

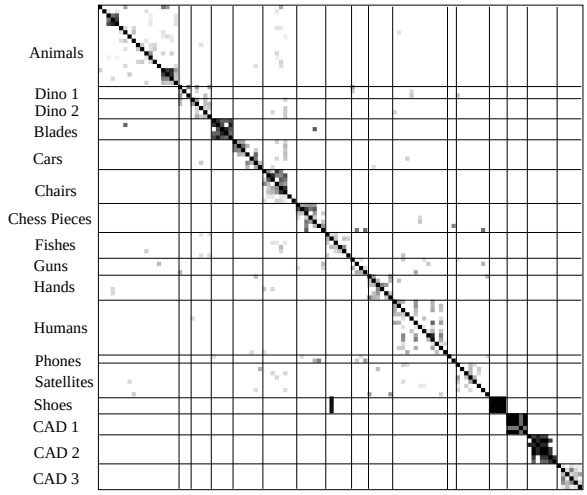


Figure 6: *Similarity matrix.*

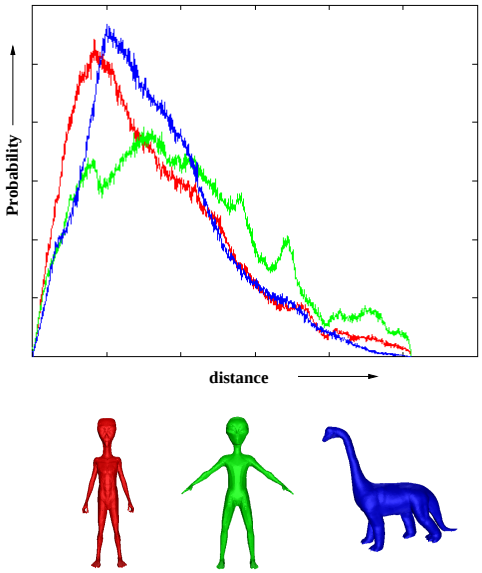


Figure 7: *In the shape distribution method the probability distribution plot of D2 distances shows a better match between the left alien and the dynosaur. Our method matches the two aliens better than with the dynosaur.*

namely the Delaunay triangulation of the point sample, and determines the tetrahedra to be clustered to form the feature segments. The feature segments obtained by the method can be used effectively for shape matching.

Some questions remain open. Our method is suitable for segmenting and hence matching volumes. It does not work for surfaces that have boundaries. Some applications need partial matchings where surfaces with boundaries need to be considered. Also, the method is not suitable for matching deformable objects. For example, a crawling human will be treated differently from the same human with no such crawl. Notice that the segmentation will dissect the features correctly in both cases though the matching may not compensate for the deformations. We are currently investigating these issues.

Acknowledgements. This work is partially supported by NSF CARGO grant DMS-0138456 and the ARO grant DAAD19-02-1-0347.

References

[ABCO*01] ALEXA M., BEHR J., COHEN-OR D., FLEISHMAN S., LEVIN D., SILVA C. T.: Point set surfaces. In *Proc. IEEE Visualization* (2001), pp. 21–28.

[ACK01] AMENTA N., CHOI S., KOLLURI R.: The power crust, union of balls, and the medial axis transform. *Comput. Geom. Theory Appl. 19* (2001), 127–153.

[AG96] ALT H., GUIBAS L. J.: Discrete geometric shapes: matching, interpolation, and approximation: a survey. *Tech. report B 96-11, EVL-1996-142, Institute of Computer Science, Freie Universität Berlin* (1996).

[BM02] BELONGIE S., MALIK J.: Matching with shape contexts. *IEEE Trans. PAMI 24* (2002), 509–522.

[DG04] DEY T. K., GOSWAMI S.: Provable surface reconstruction from noisy samples. In *Proc. 20th Annu. Sympos. Comput. Geom.*, (2004).

[DGG03] DEY T. K., GIESEN J., GOSWAMI S.: Shape segmentation and matching with flow discretization. In *Proc. Workshop Algorithms Data Structures, LNCS 2748* (2003), pp. 25–36.

[EFL98] EDELSBRUNNER H., FACELLO M. A., LIANG J.: On the definition and the construction of pockets in macromolecules. *Discrete Apl. Math. 88* (1998), 83–102.

[GJ02] GIESEN J., JOHN M.: The flow complex: a data structure for geometric modeling. In *Proc. 14th Annu. ACM-SIAM Sympos. Discrete Algorithms* (2002), pp. 285–294.

[Gro93] GROVE K.: Critical point theory for distance functions. In *Proc. Sympos. Pure Math.* (1993), vol. 54, pp. 357–385.

[HKR93] HUTTELNOCHER D. P., KLANDERMAN G. A., RUCKLIDGE W. J.: Computing images using the hausdorff distance. *IEEE Trans. PAMI 15* (1993), 850–863.

[HSKK01] HILAGA M., SHINAGAWA Y., KOMURA T., KUNNI T.: Topology matching for fully automatic similarity estimation of 3d shapes. In *Proc. SIGGRAPH* (2001), pp. 203–212.

[JH99] JOHNSON A. E., HEBERT M.: Using spin-images for efficient multiple model recognition in cluttered 3-d scenes. *IEEE Trans. PAMI 21* (1999), 433–449.

[KT03] KATZ S., TAL A.: Hierarchical mesh decomposition using fuzzy clustering and cuts. In *Proc. SIGGRAPH* (2003), pp. 954–961.

[LK01] LEYMARIE F., KIMIA B.: The shock scaffold for representing 3d shape. In *Proc. 4th Internat. Workshop Visual Form., LNCS 2059, Springer-Verlag* (2001), pp. 216–229.

[OFCD01] OSADA R., FUNKHOUSER T., CHAZELLE B., DOBKIN D.: Matching 3d models with shape distribution. In *Proc. Shape Modelling Internat.* (2001).

[PKKG03] PAULY M., KEISER R., KOBBELT L. P., GROSS M. H.: Shape modeling with point-sampled geometry. In *Proc. SIGGRAPH* (2003), pp. 641–650.

[SKK01] SEBASTIAN T. B., KLEIN P. N., KIMIA B.: Recognition of shapes by editing shock graphs. In *Proc. ICCV* (2001), pp. 755–762.

[TC92] TAUBIN G., COOPER D.: *Geometric invariance in computer vision, Ch. Object Recognition Based on Moment (of algebraic) Invariants*. MIT press, 1992.