Identifying Flat and Tubular Regions of a Shape by Unstable Manifolds

Samrat Goswami [*]
CS and ICES, U. Texas at Austin,
Austin, TX 78712

Tamal K. Dey [†]
CSE, Ohio State U.,
Columbus, OH 43210

Chandrajit L. Bajaj [‡]
CS and ICES, U. Texas at Austin,
Austin, TX 78712

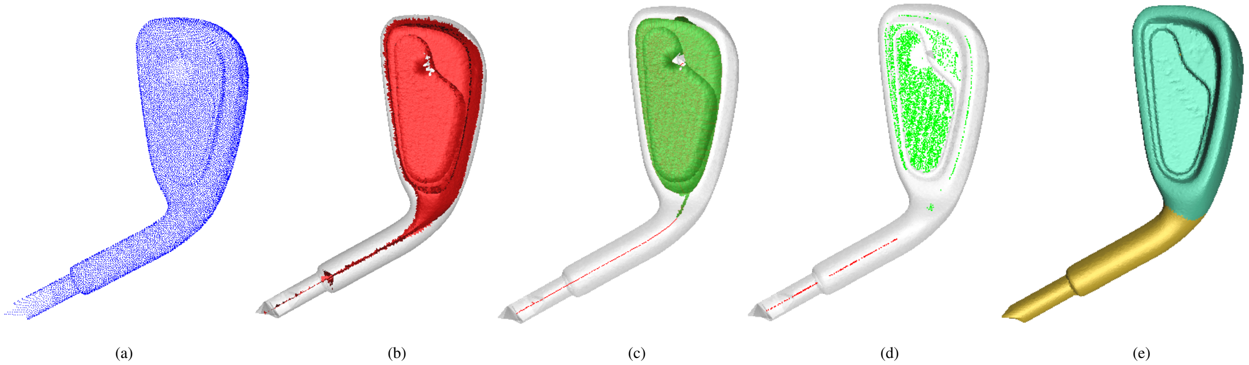

Figure 1: The steps of the algorithm are shown on an example dataset CLUB. Starting with an input set of points sampled from the surface (a), the medial axis in the interior of the shape is computed (b). The algorithm then detects the set of index 1 and index 2 saddle points lying on the interior medial axis and computes the unstable manifold of these saddle points (c). The unstable manifold of an index 1 saddle point is two dimensional (green) and the unstable manifold of an index 2 saddle point is one dimensional (red). The algorithm then collects the local maxima lying on the boundaries of these two types of unstable manifolds and tag them as falling into two different categories. The stable manifolds of these maxima are then used to map the 2-dimensional and 1-dimensional part of the medial axis back to the surface. The flat portion on the surface is colored cyan and the tubular region is colored golden (e).

Abstract

We present an algorithm to identify the flat and tubular regions of a three dimensional shape from its point sample. We consider the distance function to the input point cloud and the Morse structure induced by it on $\mathbb{R}^3$. Specifically we focus on the index 1 and index 2 saddle points and their unstable manifolds. The unstable manifolds of index 2 saddles are one dimensional whereas those of index 1 saddles are two dimensional. Mapping these unstable manifolds back onto the surface, we get the tubular and flat regions. The computations are carried out on the Voronoi diagram of the input points by approximating the unstable manifolds with Voronoi faces. We demonstrate the performance of our algorithm on several point sampled objects.

CR Categories: F.2.2 [Nonnumerical Algorithms and Problems]: Geometrical problems and computations; I.3.5 [Computational Geometry and Object Modeling]: Curve, surface, solid and object representations

Keywords: Point cloud, Voronoi diagram, Delaunay triangulation, Unstable Manifold.

1 Introduction

Problem and motivation. Many applications in shape modeling require to identify the salient features of a given shape. Some of them such as assembly planning, feature tracking, animations, structure elucidation of bio-molecules, human-body modeling benefit from a semantic annotation of the features. One such natural annotation is achieved by classifying the features as 'tubular' and 'flat'. Obviously, this annotation is ambiguous since the feature-space is a continuum resulting into features that cannot be simply classified as tubular or flat. Nevertheless, many designed and organic shapes have pronounced features that are perceived to be tubular and flat. We seek to identify these features using a topological method. The unstable manifolds induced by a shape distance function identify some one- and two-dimensional subsets of the medial axis. The preimage of a function that maps the points on the surface to the medial axis provides an association of the shape to these one- and two-dimensional subsets. The preimage of the one-dimensional subset is called tubular whereas that of the two-dimensional subset is called flat. Our experimental result shows that this classification can be effectively approximated for many datasets in practice.

Previous results. Because of the significance of the problem, quite a few work spanning various approaches have been reported

[*] e-mail:samrat@ices.utexas.edu
[†] e-mail: tamaldey@cse.ohio-state.edu
[‡] e-mail:bajaj@cs.utexas.edu

in the literature. To mention a few, we refer to the curvature based methods of [Várady et al. 1997] and [Mortara et al. 2004a; Mortara et al. 2004b], the fuzzy clustering method of [Katz and Tal 2003], the method based on PCA of surface normals by [Pottmann et al. 2004], the hybrid variational surface approximation by [Wu and Kobbelt 2005] and the Reeb graph approach of [Shinagawa et al. 1996] and [Verroust and Lazarus 2000]. Remarkably the distance function over $\mathbb{R}^3$ which is defined by the distance to the boundary of the shape has not been fully used for feature annotation. In the context of surface reconstruction, topological structures induced by distance functions have been analyzed by Edelsbrunner [Edelsbrunner 2002], Chaine [Chaine 2003] and Giesen and John [Giesen and John 2003]. Chazal and Lieutier [Chazal and Lieutier 2004] and Siddiqi et al. [Siddiqi et al. 1998] have used it for medial axis approximations. Dey, Giesen and Goswami used the topological structures induced by the distance function to segment a shape [Dey et al. 2003]. However, this work stops short of using the topological structures for feature annotations. In this paper we complete this step.

Results. Given a compact surface Σ smoothly embedded in $\mathbb{R}^3$, a distance function h_Σ can be assigned over $\mathbb{R}^3$ that assigns to each point its distance to Σ.

$$h_\Sigma \,:\, \mathbb{R}^3 \to \mathbb{R},\ x \mapsto \inf_{p\in\Sigma} \|x-p\|$$

In applications, Σ is often known via a finite set of sample points P of Σ. Therefore it is quite natural to approximate the function h_Σ by the function

$$h_P \,:\, \mathbb{R}^3 \to \mathbb{R},\ x \mapsto \min_{p\in P} \|x-p\|$$

which assigns to each point in $\mathbb{R}^3$ the distance to the nearest sample point in P.

In this paper, we start with a finite sample P of Σ and identify the index 1 and index 2 saddle points of h_P from the Voronoi diagram Vor$P$ and its dual Delaunay triangulation Del$P$ of P. We then select only the saddle points of both indices which lie on the interior medial axis of Σ and compute their unstable manifolds. The unstable manifold of index 1 saddle points (U_1) are two dimensional whereas those of index 2 (U_2) are one dimensional. Exact computations of U_1 is prone to numerical error. So, we present an algorithm to compute them approximately. We then map the points belonging to U_1 and U_2 back to Σ. The image of U_1 under the mapping gives the flat regions of Σ and that of U_2 gives its tubular regions.

Thus, the main contributions of this paper are:

- Algorithms to compute the unstable manifolds of the index 2 saddles points of h_P exactly and those of the index 1 saddle points approximately,
- Identification of the tubular and flat features of Σ from its point sample P via the unstable manifolds of the saddle points,
- Experimental results exhibiting the performance of our algorithm on several point sampled objects.

The paper is organized as follows. In Section 2 we state some definitions and explain the terms such as Voronoi-Delaunay diagram, induced flow, stable/unstable manifolds etc. In Section 3 we describe the relation between the Voronoi-Delaunay diagram of the point set P and the induced flow. In Section 4 we describe the structure of the unstable manifolds of index 1 and index 2 saddle points and present an algorithm to compute them. In Section 5 we give an algorithm to map the unstable manifolds back to the surface to identify its flat and tubular features. In Section 6 we demonstrate the results of our algorithm on several models ranging from CAD objects to protein molecules. We conclude in Section 7.

2 Preliminaries

2.1 Voronoi-Delaunay Diagram of P

In this paper we always assume the distance metric to be Euclidean unless otherwise stated. For a finite set of points P in $\mathbb{R}^3$, the Voronoi cell of $p \in P$ is

$$V_p = \{x \in \mathbb{R}^3 \,:\, \forall q \in P - \{p\},\, \|x-p\| \le \|x-q\|)\}.$$

If the points are in general position, two Voronoi cells with non-empty intersection meet along a planar, convex Voronoi facet, three Voronoi cells with non-empty intersection meet along a common Voronoi edge and four Voronoi cells with non-empty intersection meet at a Voronoi vertex. A cell decomposition consisting of the *Voronoi objects*, that is, Voronoi cells, facets, edges and vertices is the Voronoi diagram Vor$P$ of the point set P.

The dual of Vor$P$ is the Delaunay diagram Del$P$ of P which is a simplicial complex when the points are in general position. The tetrahedra are dual to the Voronoi vertices, the triangles are dual to the Voronoi edges, the edges are dual to the Voronoi facets and the vertices (sample points from P) are dual to the Voronoi cells. We also refer to the Delaunay simplices as *Delaunay objects*.

2.2 Induced Flow

The distance function h_P induces a flow at every point $x \in \mathbb{R}^3$. This flow has been characterized earlier [Giesen and John 2003]. See also [Edelsbrunner 2002]. For completeness we briefly mention it here.

Critical Points. The critical points of h_P are those points where h_P has no non-zero gradient along any direction. These are the points in $\mathbb{R}^3$ which lie within the convex hull of its closest points from P. It turns out that the critical points of h_P are the intersection points of the Voronoi objects with their dual Delaunay objects.

- *Maxima* are the Voronoi vertices contained in their dual tetrahedra,
- *Index 2 saddles* lie at the intersection of Voronoi edges with their dual Delaunay triangles,
- *Index 1 saddles* lie at the intersection of Voronoi facets with their dual Delaunay edges, and
- *Minima* are the sample points themselves as they are always contained in their Voronoi cells.

In this discrete setting, the index of a critical point is the dimension of the lowest dimensional Delaunay simplex that contains the critical point.

Flow. For every point $x \in \mathbb{R}^3$, let $V(x)$ be the lowest dimensional Voronoi object that contains x and $D(x)$ be its dual. Now *driver* of x, denoted as $d(x)$, is defined as

$$d(x) = \text{argmin}_{y\in D(x)} \|x-y\|$$

The direction of steepest ascent can be uniquely determined by a unit vector in the direction of $x - d(x)$. The critical points coincide with their drivers. Now one can assign a vector v at every x with a zero vector assigned at the critical points. The resulting vector field is not necessarily continuous. Nevertheless, it induces a *flow* in $\mathbb{R}^3$. This flow tells how a point x moves in $\mathbb{R}^3$ along the steepest ascent of h_P and the corresponding path is known as the *orbit* of x. We can also define an *inverted orbit* of x where x moves in the direction of steepest descent.

Stable and Unstable Manifolds. For a critical point c its stable manifold is the set of points whose orbits end at c. The unstable manifold of a critical point c is the set of points whose inverted orbits end at c. The structure and computation of stable manifolds of the critical points of h_P were described in [Giesen and John 2003]. They can be computed from the Delaunay triangulations of the given point sets though they may not be subcomplexes of the Delaunay triangulations. For computational advantages they are also approximated by Delaunay subcomplexes as in [Dey et al. 2003].

We are interested in computing unstable manifolds and their approximations. As the Delaunay and Voronoi diagrams, the structures of stable and unstable manifolds have a duality. Interestingly, one can compute the unstable manifolds and their approximations from the Voronoi diagrams. Here we state some of the facts about the unstable manifolds of the critical points.

1. MAXIMA. The unstable manifold is the local maximum itself.
2. INDEX 2 SADDLES. The unstable manifold of an index 2 saddle point is a polyline starting at the saddle point and ending at a maximum.
3. INDEX 1 SADDLES. The unstable manifold of an index 1 saddle point is a two dimensional surface patch which is bounded by the unstable manifold of index 2 saddle points.
4. MINIMA. The unstable manifold of a local minimum is a three dimensional polytope bounded by the unstable manifold of critical points with higher indices.

In Section 4 the computation of the unstable manifold of index 1 and index 2 saddle points is described.

3 Flow on Voronoi Objects

Before we state the connection between the flow induced by h_P and the Vor-Del diagram of P, we would like to state some facts about the relative position of Voronoi and Delaunay objects. These relative positions can describe the nature of flows in the Voronoi objects. These facts were clearly explained in [Edelsbrunner 2002] for a more general setting of power distance.

Fact 1 *The unoriented normal to the supporting plane of a Voronoi facet is along its dual Delaunay edge and the plane passes through the midpoint of the edge. The Delaunay edge, though, may or may not intersect the dual Voronoi face.*

Figure 2 illustrates the two possibilities that may arise. The left figure corresponds to the situation that results in an index 1 saddle point.

Fact 2 *The supporting line of a Voronoi edge always intersects the plane of the dual Delaunay triangle at its circumcenter and is along its unoriented normal. The Voronoi edge may or may not intersect the interior of the Delaunay triangle.*

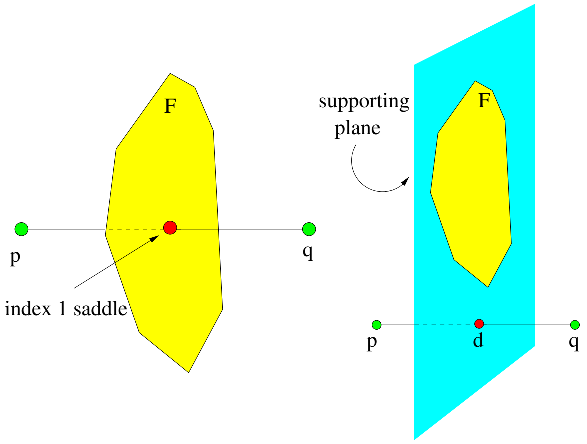

Figure 2: Relative position of a Voronoi facet F with respect to its dual Delaunay edge pq. The left picture shows the creation of an index 1 saddle point. The right picture shows the position of the driver d of F.

Figure 3 lists the four possible scenarios. The bottom right corresponds to the generation of an index 2 saddle point.

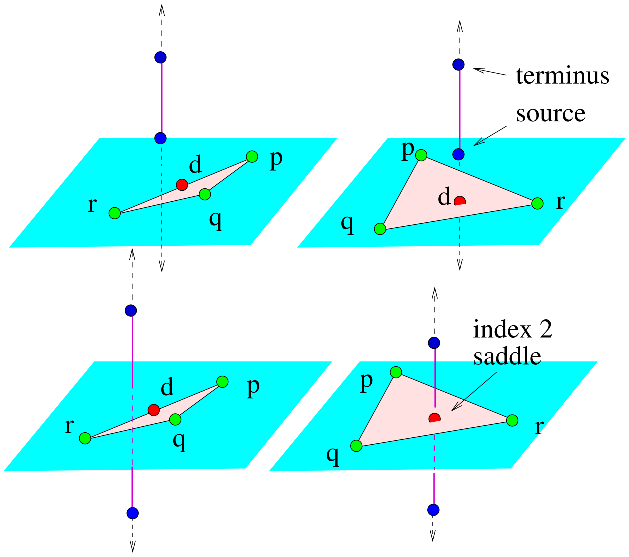

Figure 3: Relative position of a Voronoi edge e with respect to its dual Delaunay triangle pqr. The blue circles denote the two Voronoi vertices defining e. The driver of e is marked d and the supporting plane of triangle pqr is drawn in cyan.

We have already seen that the critical points of h_P can be computed from Vor P and Del P. Also, the driver of a point x comes from the Delaunay object dual to the Voronoi object x lies in. In this context we would like to state the following lemma which is key to the further computations.

Lemma 1 *All interior points of a Voronoi object have the same driver.*

This result can be easily proved by considering all the different cases regarding the dimension of the Voronoi object and its position with respect to its dual Delaunay object.

By Lemma 1 and Facts 1 and 2 we can list the possible position of the drivers of the points lying in the interior of a certain dimensional Voronoi object.

Position of Drivers

Voronoi Cell

For a Voronoi cell V_p, the dual Delaunay object is a singleton set containing the sample point p and therefore all points x in the interior of V_p has p as their driver.

Voronoi Facet

Consider a Voronoi facet in the intersection of V_p and V_q. The dual Delaunay edge is pq and the midpoint of pq is the driver of all x lying in the interior of the Voronoi facet (Figure 2(right)).

Voronoi Edge

Next, consider a Voronoi edge in the intersection of V_p, V_q, V_r. As Fact 2 and Figure 3 indicate, the infinite line segment containing the Voronoi edge may or may not intersect the convex hull of p, q, r leading to two different possibilities

Case 1.1 In case of intersection, the circumcenter of pqr is the driver. Such Voronoi edges will be termed *non-transversal* edges as the flow is along the edge itself. The Voronoi edge has two Voronoi vertices as its endpoints. If both of them are in the same half-space defined by pqr, the closer Voronoi vertex is called *source* and the further one is called *terminus* of the Voronoi edge because the flow is directed from the closer to the further vertex. Figure 3 (top right) illustrates this case.

Case 1.2 If the Voronoi edge does not intersect the affine hull of $p, q,$ and r, the midpoint of the edge opposite to the largest angle of pqr is the driver. These Voronoi edges will be termed as *transversal*. If any point x moving along its orbit hits one such edge, the position of the driver implies that it will enter the Voronoi facet dual to the Delaunay edge opposite to the largest angle in pqr. Such Voronoi facet will be termed *acceptor* facets of that *transversal* Voronoi edge. Figure 4 illustrates the situation.

Voronoi Vertex

The case of Voronoi vertex again requires the analysis of two different cases. We assume, that it is outside its dual tetrahedron because otherwise it is a local maximum and hence is its own driver. Let v be a Voronoi vertex with the dual tetrahedron σ whose four neighbors are $\sigma_i, i = 1 \dots 4$. Further, let the corresponding shared triangles between σ and σ_i be $t_i, i = 1 \dots 4$ where $w_i, i = 1, \dots 4$ is its opposite vertex in σ.

Case 2.1 There is only one triangle t_i of σ for which the Voronoi vertex v and the opposite vertex w_i lie in two different half-spaces defined by t_i. Let e_i be the Voronoi edge between the duals of σ and σ_i. Then, the driver for v (dual to σ) is same as the driver of e_i. In such cases, e_i is termed as the *outgoing* Voronoi edge of v. See top row of Figure 5 for an illustration.

Case 2.2 There are two triangles t_i, t_j of σ for which the Voronoi vertex v and the opposite vertex (w_i and w_j) lie in two different half-spaces defined by the corresponding triangles. Let e_i, e_j be the Voronoi edges defined as in Case 2.1. Note, in this case, both e_i, e_j are the *outgoing* Voronoi edges of v. There are two possibilities that we need to consider further.

Case 2.2.1 Both e_i, e_j are *transversal*: In this case the *acceptor*s of both of them is dual to the Delaunay edge $t_i \cap t_j$ and the corresponding driver is the midpoint of $t_i \cap t_j$. See bottom-left subfigure of Figure 5.

Case 2.2.2 One of e_i, e_j is *transversal*: The driver is same as that of the non-transversal Voronoi edge. See bottom-right subfigure of Figure 5.

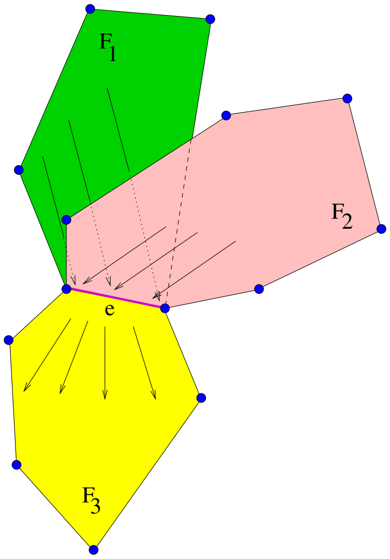

Figure 4: Transversal Voronoi edge e is shown in red with three incident Voronoi facets. Flow direction is shown with arrows. Flow from either of F_1 or F_2 hits e and enters F_3, the acceptor of e.

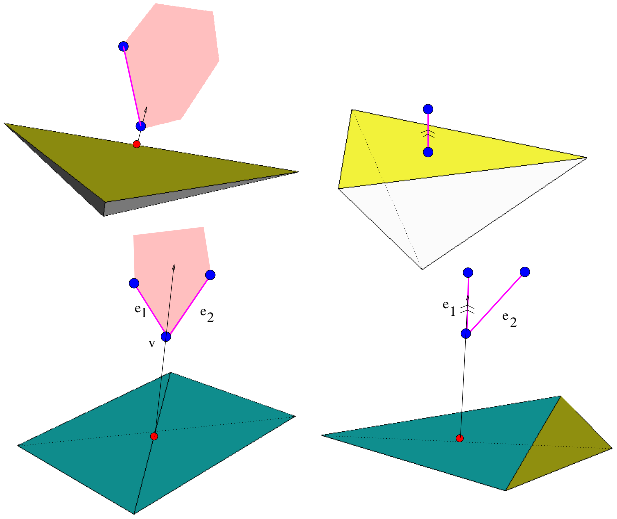

Figure 5: Possible driver positions of a Voronoi vertex v according to the cases 2.1 and 2.2.$(1-2)$. The acceptor Voronoi facet is shown in pink. The flow along a non-transversal Voronoi edge is shown with a double arrow. The driver is shown in red circle.

In this context we state another lemma that is important for subsequent developments.

Lemma 2 *Let F be an acceptor Voronoi Facet for the transversal Voronoi edges $e_1 = (v_1, v_2) \dots e_k = (v_k, v_{k+1})$ around it.*

1. *The Voronoi edges $e_1 \ldots e_k$ form a continuous chain around F.*
2. *The Voronoi vertices $v_2 \ldots v_k$ fall in the category 2.2.1. The Voronoi vertices v_1 and v_{k+1} fall in the category 2.2.2.*
3. *F, $e_1 \ldots e_k$, $v_2 \ldots v_k$ have same driver which is the midpoint of the Delaunay edge dual to F.*

We omit the proofs of all of the above claims.

4 Computing Unstable Manifolds

4.1 Unstable Manifold of Index-2 Saddle Points

In this section we describe the structure and computation of the unstable manifolds of index 2 saddle points.

The unstable manifold of an index 2 saddle point is one dimensional. In our discrete setting it is a polyline with one endpoint at the saddle point and the other endpoint at a local maximum. The polyline consists of segments that are either subsets of non-transversal Voronoi edges or lie in the Voronoi facets. Due to the later case, the polyline may not be a subcomplex of Vor$P$.

Let us consider an index 2 saddle point, c, at the intersection of a Delaunay triangle t with a Voronoi edge e. Let the two tetrahedra sharing f be σ_1, σ_2. The edge e has the endpoints at the dual Voronoi vertices of σ_1 and σ_2, denoted as v_1, v_2 respectively. The unstable manifold $U(c)$ of c, has two intervals - one from c to v_1 and the other from c to v_2. We look at the structure of one of them, say the one from c to v_1, and the other one is similar.

At any point on the subsegment cv_1, the flow is toward v_1 from c. Once the flow reaches v_1, the subsequent flow depends on the driver of v_1. Instead of just looking at v_1, we consider a generic step, where the flow reaches at a Voronoi vertex v and we enumerate the possible situations that might occur depending on the position of driver of v. If v is a local maximum, the flow stops there, as the driver of v is v itself. Otherwise there are two cases to consider.

- v **falls into Case 2.1:** Let the dual tetrahedron be σ and the driver of v is same as that of the Voronoi edge e which is between the dual of σ and one of its neighbors, say σ'. If e is non-transversal, the flow will be along the Voronoi edge e till it hits the Voronoi vertex at the other endpoint (dual to σ'). Otherwise, the flow enters the acceptor Voronoi facet F of e. Due to Lemma 2, the driver of F is same as the driver of e. Therefore the next piece of the unstable manifold can be uniquely determined by the driver of e, say d and the Voronoi vertex v. It is the segment between v and the point where the ray $\overrightarrow{dv}$ intersects a Voronoi edge of F.
- v **falls under Case 2.2.x:** This situation is similar to the one described above. In case of both of the Voronoi edges being *transversal* (Case 2.2.1), the flow enters the acceptor Voronoi facet. In the other case (Case 2.2.2), the flow follows the non-transversal Voronoi edge.

Some segments of $U(c)$ are not along the Voronoi edges. Wherever the flow encounters a transversal Voronoi edge, it seizes to follow the Voronoi edge and enters a Voronoi facet which is acceptor for that Voronoi edge. This calls for the analysis of the flow when it crosses an acceptor Voronoi facet and hits a Voronoi edge. We have already characterized the position of the driver for a Voronoi edge and thereby classified those edges as either transversal or non-transversal. If the current edge intersected by the ray from the driver to v is a non-transversal edge, the flow will follow that Voronoi edge and hit a Voronoi vertex. Otherwise, it will enter the acceptor Voronoi facet of the Voronoi edge again. There is a technical difficulty we need to point out. Unless the acceptor for this Voronoi edge is different from the Voronoi facet the flow came from, we may encounter a cycle. The following lemma saves us from this awkward situation.

Lemma 3 *Let F be a Voronoi facet and let d be its driver. Let e be a Voronoi edge for which F is acceptor and x be any point on e. Also assume the ray from d to x intersects a Voronoi edge e'. If e' is transversal, the acceptor of e' is different from F.*

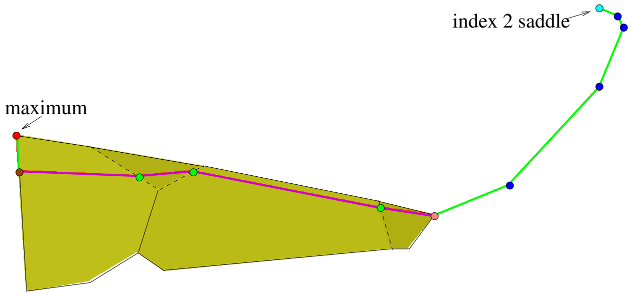

Figure 6: Unstable manifold $U(c)$ of an index 2 saddle point c. c is drawn with a cyan circle. The portion of $U(c)$ which is a collection of Voronoi edges is drawn in green with intermediate Voronoi vertices drawn in blue. The pink circle is a Voronoi vertex on $U(c)$ where the flow enters a Voronoi facet. The portion of $U(c)$ which lies inside the Voronoi facets is drawn in magenta. The transversal Voronoi edges intersected by this portion of $U(c)$ are dashed. $U(c)$ ends at a local maximum which is drawn in red.

Figure 6 shows an example of the unstable manifold of an index 2 saddle point.

Following the above discussion on the structure of $U(c)$ we devise the algorithm to compute the unstable manifold of an index 2 saddle point c. We assume, the saddle point c carries the information about the two neighboring tetrahedra σ_1, σ_2 and additionally we have access to Del$P$ which is used to evaluate the utility routines like acceptor() , terminus() etc. The pseudo-code of the algorithm is given in Figure 7.

4.2 Unstable Manifold of Index-1 Saddle Points

Unstable Manifold of index 1 saddle points are two dimensional. Due to hierarchical structure, they are bounded by the unstable manifold of index 2 saddle points. In this section we first describe the structure of the unstable manifolds and then describe an algorithm that computes an approximation of the unstable manifold of an index 1 saddle point.

Let us consider an index 1 saddle point, c. This point lies at the intersection of a Voronoi facet F and a Delaunay edge. For any point $x \in F \setminus c$, the driver is c. For all such x, if they are allowed to move in the direction of flow, they will move radially outward and hit the Voronoi edges bounding F. Thus F is in $U(c)$. Now we analyze the flow when a point hits a Voronoi edge.

We have characterized the position of the drivers for a Voronoi edge and we have also seen that depending on the driver, one can classify the Voronoi edges into two categories - transversal and non-

```
UM_INDEX_2($c$)
1   $U_1 = cv_1$ and $U_2 = cv_2$
2   $v = v_1$
3   end$(U_1) = v_1$
4   while ($v$ is not a maximum) do
5       if($v$ is not a Voronoi vertex)
6           $e =$ Voronoi edge containing $v$
7           if($e$ is non-transversal)
8               end$(U_1) =$ terminus$(e)$
9               $U_1 = U_1 \cup$ segment$(v, $end$(U_1))$
10              $v =$ terminus$(e)$
11          else
12              $F =$ acceptor$(e)$
13              $d =$ driver$(F) =$ driver$(e)$
14              $x = \overrightarrow{dv} \cap e' \neq \emptyset$, $e'$ is a Voronoi edge of $F$
15              end$(U_1) = x$
16              $U_1 = U_1 \cup$ segment$(v, $end$(U_1))$
17              $v = x$
18      else
19          if($v$ falls under Case 2.1)
20              $e =$ *outgoing* Voredge (v)
21              repeat steps 7-17.
22          else if($v$ falls under Case 2.2)
23              $F =$ acceptor$(v)$
24              repeat steps 13-17.
25  endwhile
26  Similarly compute $U_2$.
27  return $U_1 \cup U_2$.
```

Figure 7: Pseudo-code for computation of unstable manifold of an index 2 saddle point.

transversal. For a non-transversal Voronoi edge, the flow is along the Voronoi edge. Such Voronoi edges lie on the boundary of $U(c)$. On the other hand, $U(c)$ grows via the acceptor facets of transversal Voronoi edges. Depending on the position of the driver, which by Lemma 2 is same for both the edge and the acceptor facet, a *truncated cone* defines the extension of $U(c)$ into the acceptor Voronoi facet. Consider the cone defined by the two rays emanating from the driver and passing through the endpoints of the transversal Voronoi edge. The intersection of the acceptor facet with the cone defines the truncated cone. The truncated cone hits a continuous chain of Voronoi edges in the acceptor facet. Some of them are completely contained in the truncated cone and some of them are intersected by the two rays and hence are partially contained in it. This chain of edges defines the new boundary of $U(c)$ through some of which $U(c)$ can be extended further recursively. Figure 8 shows an example truncated cone in a Voronoi facet F by the driver d and the end Voronoi vertices of the transversal Voronoi edge (green).

To compute $U(c)$ accurately, one therefore needs to compute the intersection of a ray and a line segment in three dimension. Such computations are prone to numerical errors. Therefore, we rely on an approximation algorithm that computes a superset of $U(c)$. The algorithm works as follows.

Starting from the Voronoi facet F containing c, we maintain a list of Voronoi facets which are already in $U(c)$ and a list of active Voronoi edges which are transversal edges and lie on the boundary of the current approximation of $U(c)$. Through these transversal edges we collect their acceptor facets and grow $U(c)$. Instead of computing the new set of active edges by an expensive numerical calculation of ray-segment intersection, we collect all the transversal edges of this new acceptor Voronoi facets. This way we grow $U(c)$ recursively till we have a set of Voronoi facets which are bounded by only a set of transversal Voronoi edges.

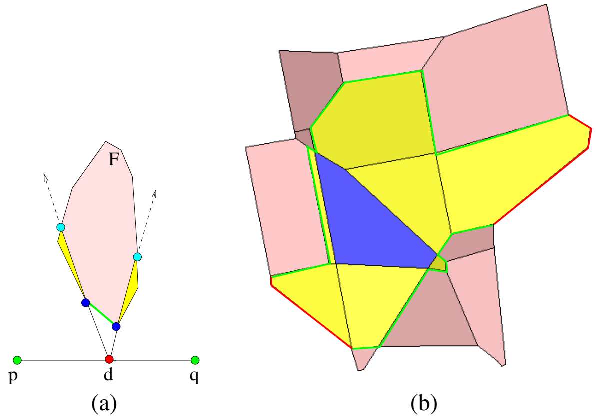

Figure 8: (a) Truncated Cone. Accurate computation selects only the pink region from the yellow Voronoi facet as part of unstable manifold of an index 1 saddle point c (not shown). (b) Snapshot of approximate computation of $U(c)$ at a generic stage.

Figure 8(b) illustrates an intermediate stage of this computation. The index 1 saddle point c is contained in the blue Voronoi facet. The yellow Voronoi facets are already in $U(c)$. The red edges designate the static boundary as they are non-transversal and the green edges designate the active boundary through which the pink facets are included in $U(c)$ in the later stage of the algorithm. Following is the pseudo-code for this algorithm. Given an index 1 saddle point c it computes an approximation of $U(c)$. We assume c also has information about the Voronoi facet F it is contained in.

```
APPROX_UM_INDEX_1($c$)
    $U = F$
    $B =$ Voronoi edges of $F$
    while ($B \neq \emptyset$) do
        $e =$ pop$(B)$
        if ($e$ is transversal)
            $U = U \cup$ acceptor$(e)$
            $B = B \cup$ unvisited edges of acceptor$(e)$
    endwhile
    return $U$.
```

Figure 9: Pseudo-code for approximate computation of unstable manifold of an index 1 saddle point.

4.3 Classification of Medial Axis

In the previous two subsections we have described the structures of the unstable manifolds of an index 1 and index 2 saddle points. We have also given an accurate and an approximate algorithm to compute them. Our goal is to identify the unstable manifolds near the medial axis of Σ. Ultimately these manifolds are mapped back to Σ for the feature annotation. For this we first compute a Voronoi subcomplex that approximates the medial axis M_Σ and then identify different regions of this approximate medial axis as the unstable manifolds computed by the two subroutines UM_INDEX_2 and APPROX_UM_INDEX_1.

Before we describe our approach, we briefly mention a recent result by Dey, Giesen, Ramos and Sadri [Dey et al. 2005] where they proved that under sufficient sampling of Σ by P, the critical points of h_P lie either close to Σ or close to M_Σ. This motivates our approach. Applying the same result, we filter out only the index 1 and index 2 saddle points near M_Σ instead of Σ. Further, we consider only the components of M_Σ which lie in the interior of the solid bounded by Σ. For this purpose we use the TIGHTCOCONE algorithm by Dey and Goswami [Dey and Goswami 2003]. The implementation of this algorithm is freely available in the public domain [Cocone] along with the software for medial axis approximations which is computed as a Voronoi subcomplex according to the algorithm by Dey and Zhao [Dey and Zhao 2004]. For the purpose of reconstruction, any other reconstruction algorithm also could be used [Bernardini et al. 1999; Bajaj et al. 1995]. Applying TIGHTCOCONE followed by medial axis approximation we get the approximate interior medial axis of Σ. We perform the critical point detection only within the Voronoi subcomplex that approximates this medial axis. Let us call this set of index 1 saddle points C_1 and that of index 2 saddle points C_2. We then apply UM_INDEX_2(c) for all $c \in C_2$ and APPROX_UM_INDEX_1(c) for all $c \in C_1$. $U(c \in C_1)$ is two dimensional and $U(c \in C_2)$ is one dimensional. Therefore, by restricting the unstable manifold computation only within M_Σ we obtain two subsets of M_Σ. In the next section, we describe how this classification can be mapped back to Σ for automatic identification of its flat and tubular regions.

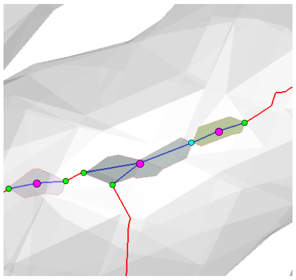

Figure 10: Removal of small patches in the tubular region via starring. Magenta circles indicate the centroids of these patches, green circles are the boundary vertices which connect a patch with a linear portion (red line) and cyan circle indicates where two different patches join at a common vertex. Blue lines are the replacements of these small patches obtained by the starring process.

Because of sampling artifacts, sometimes the interior medial axis in the tubular regions have a few index 1 saddle points. The unstable manifold of these saddle points need to be detected and approximated by lines. We partition the set C_1 based on the connectivity of their unstable manifolds via a common edge and every partition creates a patch which is the union of the unstable manifolds of all the index 1 saddle points falling into that partition. We further assign an *importance* value based on the area of the patch and sort the patches according to their *importance*. One could also employ other attributes like diameter, width etc. to evaluate the importance. The small clusters are then detected either by a user-specified threshold value or by simply selecting the k-smallest clusters where k is also a user-supplied parameter. These insignificant planar regions are then approximated by a set of straight lines emanating from the centroid of the patch to the boundary points which are connected to either a polyline (green circles in Figure 10) or another patch (cyan circle in Figure 10). We call this process *starring*.

The resulting one dimensional and two dimensional subsets of the interior medial axis is shown in Figure 11. Left column shows the approximate medial axis computed by [Dey and Zhao 2004]. The right column shows the subset of medial axis captured by $U(C_2)$ and $U(C_1)$.

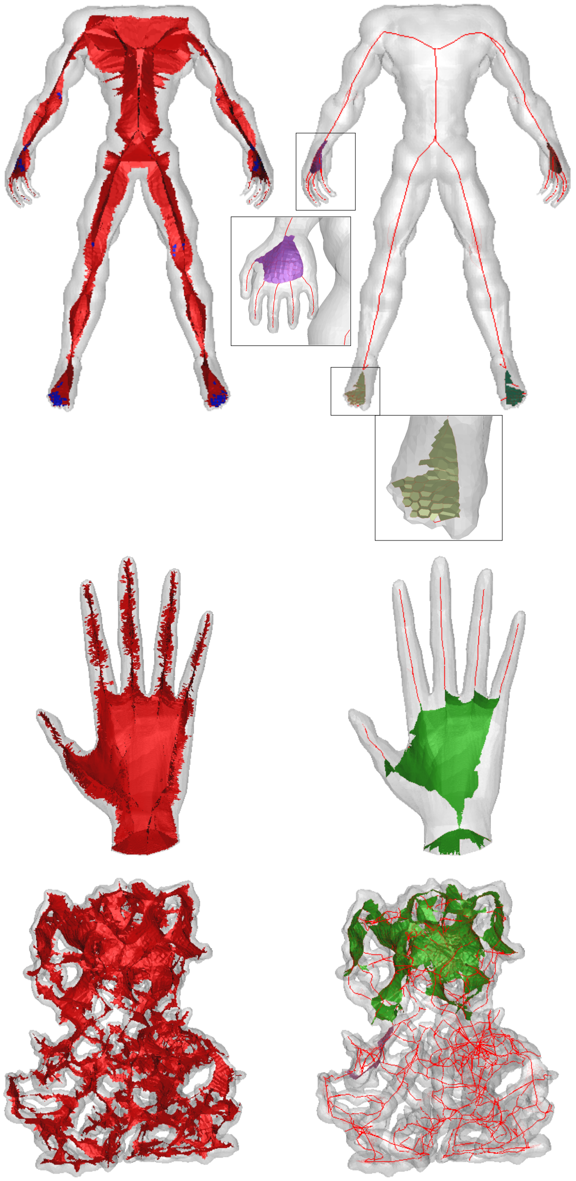

Figure 11: Results of Medial Axis classification. Top row shows the result for HEADLESS MAN. Two closeups have been shown to highlight the planar clusters in the palm of the hand and the feet. The closeup of hand has been rotated for visual clarity. The middle row shows the result on HAND dataset and the bottom row shows the result on a molecule data 1BVP.

5 Feature Annotation Algorithm

5.1 Mapping of Unstable Manifolds to Σ

There is a natural association between the medial axis M_Σ and Σ via the map $\phi : \Sigma \longrightarrow M_\Sigma$ where $\phi(x)$ is the center of the medial ball touching Σ at x. Following this map, any subset $A \subseteq M_\Sigma$ can be associated with $\phi^{-1}(A) \subseteq \Sigma$. Let A_1 and A_2 be the closure of the unstable manifolds of index 2 and index 1 saddles in M_Σ defined by the distance function h_Σ. Recall that, generically, A_1 is one-dimensional and A_2 is two-dimensional. Ideally, we would like to identify $\phi^{-1}(A_1) \subseteq \Sigma$ as tubular and $\phi^{-1}(A_2) \subseteq \Sigma$ as flat. As we have an approximation of h_Σ by h_P, we compute these tubular and flat regions for the unstable manifolds in the approximate medial axis which we denote also as M_Σ for convenience.

We face a difficulty to compute an approximation of the preimage of ϕ from the approximate medial axis M_Σ. We are interested in computing an approximation of the preimage of $M'_\Sigma = A_1 \cup A_2 \subseteq M_\Sigma$ under the map ϕ.

Unfortunately, this requires an expensive computation to cover the entire M'_Σ which often spans a substantial portion of M_Σ. A naive approach is to take only a sample of M'_Σ, namely the Voronoi vertices, and then associate them to P, a sample of Σ, via the Voronoi-Delaunay duality. This also proves useless because M'_Σ does not contain all the Voronoi vertices and therefore many points in P cannot be covered by this Voronoi-Delaunay duality.

It turns out that the distance function h_P again proves to be useful to establish a correspondence between Σ and M'_Σ. Recall that, the stable manifold of a critical point is a collection of points whose orbits terminate at that critical point. Let X and Y be the set of maxima in $A_1 \subseteq M'_\Sigma$ and $A_2 \subseteq M'_\Sigma$ respectively. Consider the stable manifolds of the maxima in X and Y. The points in P that are in the stable manifolds of X are associated with the tubular regions and those in the stable manifolds of Y are associated with the flat regions. If a point belongs to the stable manifolds of maxima in X as well as in Y, we tag it arbitrarily. These points belong to the regions where a tubular part meets a flat part. Subsequently, every triangle of the surface reconstructed by TIGHT COCONE is tagged as flat or tubular if at least two of its vertices are already marked as flat or tubular respectively.

Computation of stable manifold of maxima has been described in [Giesen and John 2003] and its approximation was given in [Dey et al. 2003]. We follow the approximate algorithm to compute the stable manifolds of the local maxima lying on M'_Σ.

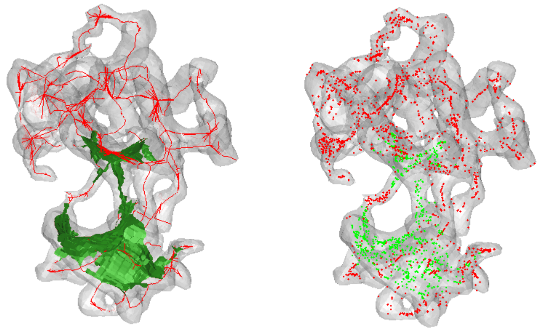

Figure 12: One dimensional subset of the interior medial axis is drawn in red and the two dimensional subset of the medial axis is drawn in green for the molecule data 1IRK. The right subfigure shows the selection of local maxima of the distance function in those two parts, colored accordingly.

Figure 12 shows the set M'_Σ of the molecule data 1IRK, and the set of maxima belonging to that set and identified as linear or planar. The corresponding flat and tubular portions of the surface captured by the mapping via stable manifold of these maxima - colored golden and cyan respectively - are shown in Figure 14. We collected the protein from Protein Data Bank [Berman et al. 2000] and blurred the molecule at a resolution 8 angstrom. Further we took the vertex set of a suitable level set as the input to our program. We verified the result with the existing literature in structural biology and we have seen that the flat regions identified by our algorithm correspond to the β-sheets of the protein molecule.

5.2 Annotation Algorithm

The modules described in the previous sections and subsections can thus be combined to devise an algorithm for automatic feature annotation of Σ. We give the pseudo-code of this annotation algorithm here.

```
IDENTIFY_FLAT_AND_TUBULAR_REGIONS($P$)
Compute Vor $P$ and Del $P$.
Compute the interior Medial Axis $M_\Sigma$
    by TIGHTCOCONE_AND_MA ($P$)
$C_1$ = set of index 1 saddle points lying on $M_\Sigma$ and
$C_2$ = set of index 2 saddle points lying on $M_\Sigma$,
$A_1 = A_2 = \emptyset$
for all $c \in C_2$
    $A_1 = A_1 \cup$ UM_INDEX_2($c$)
for all $c \in C_1$
    $A_2 = A_2 \cup$ APPROX_UM_INDEX_1($c$)
$X$ = maxima in $A_1$
$Y$ = maxima in $A_2$
$\Sigma_{Tubular}$ = MAPPING_VIA_STABLE_MANIFOLD($A_1$)
$\Sigma_{Flat}$ = MAPPING_VIA_STABLE_MANIFOLD($A_2$)
return $\Sigma_{Tubular}$ and $\Sigma_{Flat}$.
```

Figure 13: Pseudo-code of the feature annotation algorithm.

6 Results

6.1 Implementation Issues

The algorithm works on the Voronoi-Delaunay diagram of the set of sample points lying on the surface. To robustly compute the Delaunay triangulation and its dual Voronoi diagram for the input set of points we use the library CGAL [CGAL Consortium] which is freely available.

Even in CGAL-framework, we sometimes face the degenerate case of five or more points being cospherical. This case has to be handled with special care because only one Voronoi vertex is repeated and therefore the flow along the Voronoi edges is not well-defined anymore. To deal with such situations, we modify the algorithm slightly. At the start of the algorithm we collect the sets of tetrahedra which are cospherical. While computing the unstable manifold of index 2 saddle points, if the polyline hits a Voronoi vertex whose dual is a member of one such cospherical cluster, the algorithm automatically advances through the non-degenerate Voronoi edges which are dual to the triangles bounding the cospherical lump. This degeneracy poses a more serious threat to the computation of unstable manifold of index 1 saddle points and at this stage, we do not extend the manifold through any Voronoi edge whose dual Delaunay triangle is shared by two cospherical tetrahedra.

There are some parameters involved in the full feature annotation process. For surface reconstruction and medial axis approximation we used the software [Cocone]. The parameters for these routines are described in [Dey and Goswami 2003], [Dey and Zhao

2004]. For noisy inputs we replace TIGHT COCONE by ROBUST COCOCNE and the parameters for that step are again described in [Dey and Goswami 2004]. The rest of the algorithm requires only one parameter k which is the number of flat regions to be output.

6.2 Performance

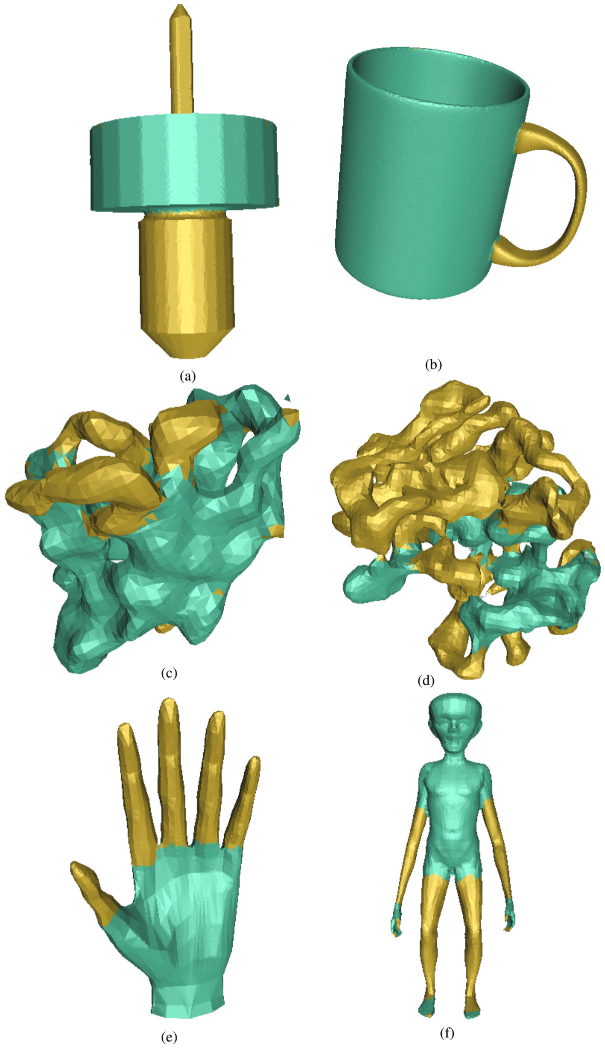

Figure 14: Performance of the feature annotation algorithm. The models are (a) PIN, (b) MUG, (c) molecule 1CID (d) molecule 1IRK, (e) HAND, (f) ALIEN

Figure 14 shows the performance of the annotation algorithm on six datasets. The datasets have been chosen to represent different domains this algorithm can possibly be applied in. PIN is a CAD dataset which has two tubular parts joined in the middle through a flat portion. The algorithm can identify them correctly. Similarly the method can correctly identify the handle as the tubular and the body as the flat region for the MUG dataset. In the second row we show the performance of our method on two protein molecules obtained from Protein Data Bank [Berman et al. 2000]. We took the crystal structure of these two molecules (PDB ID 1CID and 1IRK) and blurred them with Gaussian kernel. We further took a level set which represents a molecular surface and used the vertex set of that isosurface as the input to our algorithm. The flat features identified by our method correspond to the β-sheets of the secondary structure of those two proteins. In the last row we show the result on two free form objects containing both flat and tubular features. As we can see, the palm of the HAND has been detected as flat whereas the fingers have been detected as tubular. Our method can also capture the major flat and tubular features of ALIEN.

We purposefully show the performance of the algorithm on ALIEN as it brings forth the limitations of our algorithm. We see that a portion of the arm has been identified as flat. This is because the initial reconstruction phase could not separate the beginning of the arm from the torso due to lack of sampling. Secondly, one of the feet could not be fully identified as flat by our algorithm. This is because the approximate medial axis, that we started with, is not a close approximation of the true medial axis in that region, again due to lack of sampling. Because of that, our method fails to collect sufficiently many index 1 saddle points leading to incomplete identification of flat features in that region.

Figure 15 shows the performance of our method on noisy dataset HORSE. Instead of applying TIGHT COCONE, we first mark the interior and exterior of the closed surface from its noisy point sample by ROBUST COCONE ([Dey and Goswami 2004]) and then obtain the interior medial axis and proceed further with the unstable manifold computation and feature identification. Originally there were some thin flat regions due to the unstable manifold of some index 1 saddle points near the hind legs which we filtered out by thresholding in order to get a clean skeleton of the HORSE. In the rightmost picture we see some white triangles near the ears. These triangles appear as the mapping via stable manifold misses some points on the surface in that portion.

6.3 Timings

The time and space complexity of the algorithm is dominated by the complexity of Delaunay triangulation. We report the timings of the entire execution into four major steps

1. Step 1: Building the Voronoi-Delaunay diagram of the point set (Line 1 of Figure 13).

2. Step 2: Computation of interior medial axis. (Line 2 of Figure 13).

3. Step 3: Computation of unstable manifold of index 1 and index 2 saddle points lying on the interior medial axis. (Line 3-8 of Figure 13).

4. Step 4: Mapping the maxima in the planar and linear portion of the medial axis back to the surface. (Line 9-13 of Figure 13).

We built the code using CGAL [CGAL Consortium] and gnu C++ libraries. The code is compiled at an optimization level $-O2$. We run the experiments in a machine with INTEL XEON processor with 1GB RAM running at 1GHz cpuspeed. Table 1 reports the time taken in the four steps of the algorithm for a number of datasets. It is clear from the breakup of timing that the first two steps of building the Delaunay triangulation and then computing the interior medial axis are the two most expensive steps. For noisy datasets, additionally ROBUST COCONE is used to obtain an initial in-out marking. This step is comparatively inexpensive. For example, for NOISY HORSE (48,000 points) this step only adds 10 sec to the whole computation time which is approximately 100 sec.

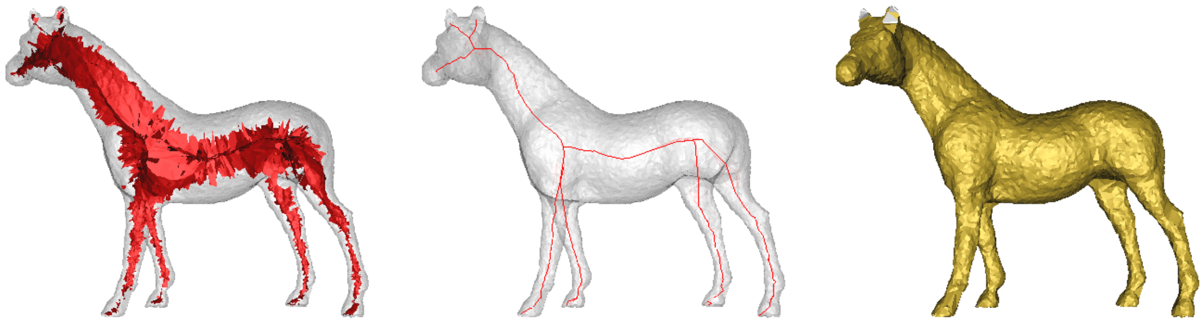

Figure 15: Results on Noisy Data.

object	# points	Step 1 (sec.)	Step 2 (sec.)	Step 3 (sec.)	Step 4 (sec.)
1CID	5170	7.59	15.63	6.69	0.39
1IRK	13940	29.88	43.93	15.63	1
Headless Man	16287	18.63	51.30	16.01	1.26
Pin	15530	15.73	41.4	21.53	0.92
Club	16864	20.54	47.3	19.83	1.24
Mug	27109	37.68	83.28	47.14	2.19
Hand	40573	53.48	120.16	40.67	2.69
P8	48046	33.46	136.59	39.97	3.22
1BVP	53392	148.18	159.52	62.19	3.53
Alien	78053	102.62	242.33	64.11	5.4

Table 1: Timings

7 Conclusions

In this work, we first described the structure of the unstable manifold of index 1 and index 2 saddle points of the distance function induced by a set of points sampled from a surface. We further used this analysis to compute the unstable manifold of an index 2 saddle point exactly and the unstable manifold of an index 1 saddle point approximately. We then used the unstable manifold of index 1 and index 2 saddle points near the medial axis of the surface to automatically detect the flat and tubular features of the shape.

We believe that this work will be useful in many areas of science and engineering. One natural connection to structural biology is the elucidation of secondary structural properties of protein molecules. Secondary structure of a protein molecule is made up of a collection of α-helices and β-sheets. α-helices are tubular and β-sheets are flat. The results of our algorithm on protein molecules have been verified against the true structural information obtained from Protein Data Bank [Berman et al. 2000] and the existing literatures in structural biology. We have seen that our method can identify the secondary structural motifs correctly. Often the applications in structural biology require to elucidate the secondary structural information in the absence of atomic level representation of a protein molecule. This is particularly the case when a protein molecule is present in a larger assembly such as in virus capsid and the input is obtained only at a resolution coarser than 4 angstrom via electron microscopy. This method will prove fruitful in analyzing the secondary structural properties in such situations.

This work has triggered several questions. We have collected the initial set of index 1 and index 2 saddle points only from the interior medial axis computed by [Dey and Zhao 2004]. This is only an approximation of the true medial axis. As a result, the collection thus obtained often misses some critical points which are close to the true medial axis but do not lie on the approximation. One needs to devise an algorithm to collect all the critical points near the true medial axis efficiently, say by the critical point separation algorithm of Dey et al. [Dey et al. 2005]. Most likely this will improve the performance of our algorithm.

The algorithm sometimes fails to collect a flat region completely as can be seen from the feet of Alien in Figure 14. This is partly due to the fact that we map the linear and planar regions of the medial axis via the stable manifold of the maxima lying in those regions. We apply the approximation algorithm of [Dey et al. 2003] to compute the stable manifolds of the maxima. This method computes these stable manifolds approximately as subcomplexes of Del P. To improve the performance of our algorithm, while still maintaining efficiency, we plan to investigate how we can use the exact computation presented in [Giesen and John 2003] only for those local maxima that border a one- and a two-dimensional region in the medial axis.

Acknowledgments

First and third authors are supported in part by NSF grants ITR-EIA-0325550, CNS-0540033 and NIH grants P20 RR020647, R01 GM074258-021 and R01-GM073087. The second author is supported in part by NSF CARGO grant DMS-0310642 and ARO grant DAAD19-02-1-0347. We thank the Jyamiti group at The Ohio State University for providing the Tight Cocone and Medial software. We also thank Zeyun Yu for providing the molecular datasets.

References

Bajaj, C., Bernardini, F., and Xu, G. 1995. Automatic reconstruction of surfaces and scalar fields from 3D scans. In *ACM SIGGRAPH*, 109–118.

Berman, H. M., Westbrook, J., Feng, Z., Gilliland, G., Bhat, T., Weissig, H., Shindyalov, I., and Bourne, P. 2000. The protein data bank. *Nucleic Acids Research*, 235–242.

Bernardini, F., Bajaj, C., Chen, J., and Schikore, D. 1999. Automatic reconstruction of 3d cad models from digital scans. *Int. J. on Comp. Geom. and Appl. 9*, 4-5, 327–369.

CGAL Consortium. CGAL: Computational Geometry Algorithms Library. *http://www.cgal.org*.

Chaine, R. 2003. A geometric convection approach of 3-d reconstruction. In *Proc. Eurographics Sympos. on Geometry Processing*, 218–229.

Chazal, F., and Lieutier, A. 2004. Stability and homotopy of a subset of the medial axis. In *Proc. 9th ACM Sympos. Solid Modeling and Applications*, 243–248.

Cocone. Tight Cocone Software for surface reconstruction and medial axis approximation. *http://www.cse.ohio-state.edu/∼tamaldey/cocone.html*.

Dey, T. K., and Goswami, S. 2003. Tight cocone: A water-tight surface reconstructor. In *Proc. 8th ACM Sympos. Solid Modeling and Applications*, 127–134.

Dey, T. K., and Goswami, S. 2004. Provable surface reconstruction from noisy samples. In *Proc. 20th ACM-SIAM Sympos. Comput. Geom.*, 330–339.

Dey, T. K., and Zhao, W. 2004. Approximating the Medial axis from the Voronoi diagram with convergence guarantee. *Algorithmica 38*, 179–200.

Dey, T. K., Giesen, J., and Goswami, S. 2003. Shape segmentation and matching with flow discretization. In *Proc. Workshop Algorithms Data Strucutres (WADS 03)*, F. Dehne, J.-R. Sack, and M. Smid, Eds., LNCS 2748, 25–36.

Dey, T. K., Giesen, J., Ramos, E., and Sadri, B. 2005. Critical points of the distance to an epsilon-sampling of a surface and flow-complex-based surface reconstruction. In *Proc. 21st ACM-SIAM Sympos. Comput. Geom.*, 218–227.

Edelsbrunner, H. 2002. Surface reconstruction by wrapping finite point sets in space. In *Ricky Pollack and Eli Goodman Festschrift*, B. Aronov, S. Basu, J. Pach, and M. Sharir, Eds. Springer-Verlag, 379–404.

Giesen, J., and John, M. 2003. The flow complex: a data structure for geometric modeling. In *Proc. 14th ACM-SIAM Sympos. Discrete Algorithms*, 285–294.

Katz, S., and Tal, A. 2003. Hierarchical mesh decomposition using fuzzy clustering and cuts. In *Trans. on Graphics*, vol. 3, 954–961.

Mortara, M., Patanè, G., Spagnuolo, M., Falcidieno, B., and Rossignac, J. 2004. Blowing bubbles for the multi-scale analysis and decomposition of triangle meshes. *Algorithmica 38*, 227–248.

Mortara, M., Patanè, G., Spagnuolo, M., Falcidieno, B., and Rossignac, J. 2004. Plumber: a multi-scale decomposition of 3d shapes into tubular primitives and bodies. In *Proc. 9th ACM Sympos. Solid Modeling and Applications*, 139–158.

Pottmann, H., Hofer, M., Odehnal, B., and Wallner, J. 2004. Line geometry for 3d shape understanding and reconstruction. *Computer Vision - ECCV 3021*, 1, 297–309.

Shinagawa, Y., Kunni, T., Belayev, A., and Tsukioka, T. 1996. Shape modeling and shape analysis based on singularities. *Internat. J. Shape Modeling 2*, 85–102.

Siddiqi, K., Shokoufandeh, A., Dickinson, J., and Zucker, S. 1998. Shock graphs and shape matching. *Computer Vision*, 222–229.

Várady, T., Martin, R., and Cox, J. 1997. Reverse engineering of geometric models - an introduction. *Computer Aided Design 29*, 255–268.

Verroust, A., and Lazarus, F. 2000. Extracting skeletal curves from 3d scattered data. *The Visual Computer 16*, 15–25.

Wu, J., and Kobbelt, L. 2005. Structure recovery via hybrid variational surface approximation. *Computer Graphics Forum 24*, 3, 277–284.