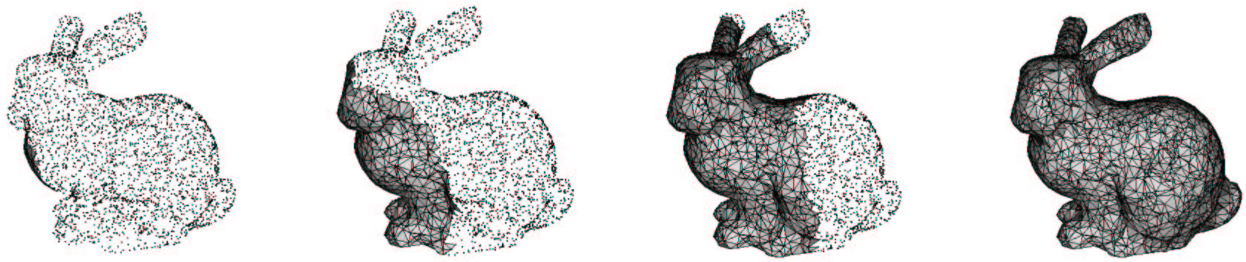
A Topological Framework for Advancing Front Triangulation

ESDRAS MEDEIROS[1], LUIZ VELHO[1], HÉLIO LOPES[2]

[1]IMPA–Instituto de Matemática Pura e Aplicada
{esdras,lvelho}@visgraf.impa.br
[2]PUC-RJ–Pontifícia Universidade Católica do Rio de Janeiro
lopes@mat.puc-rj.br

Abstract. In this paper we study advancing front methods for surface reconstruction. We propose a topological framework based on handlebody theory to implement such methods in a simple and robust way. As an example of the application of this framework we show an implementation of the Ball-Pivoting Algorithm.



Triangulating the Stanford Bunny using the Ball-Pivoting advancing front algorithm.

1 Introduction

Surface reconstruction is an important problem in Geometric Modeling and Computer Graphics and received great attention in recent years. The research in this area has been stimulated mainly by 3D Photography applications that became popular with advances in three dimensional range scanner technology.

Typically, surface reconstruction methods generate a continuous surface by means of a triangle mesh that interpolates a set of points in space. The advantage of the triangle mesh representation lies in its simplicial structure that gives geometrical and topological information about the surface.

Methods for surface reconstruction can be classified into four categories, according to Mencl and Muller [12]: spatial subdivision methods (cf. Boissonant [14]); distance function methods (cf. Hoppe et al [13]); deformation methods (cf. Zhao et al [15]); and incremental methods (cf. Bernardini et al [1]).

The advancing front triangulation algorithm is one of the most powerful among the incremental surface reconstruction methods. It is based on growing a surface by moving its boundary curves until the geometry and topology of the whole object is captured.

The main difference between algorithms in this class concerns the criteria used to advance the front. Boissonat's surface contouring algorithm [14] starts with an edge and iteratively attaches triangles at boundary edges of the emerging surface using a projection technique to generate manifolds without boundaries. Mencl and Muller [16] use graph techniques to complete the surface. Bernardini et al [1] developed the Ball-Pivoting algorithm which grow the surface locally exploiting properties of alpha shapes.

In this paper we propose a topological framework for the analysis of advancing front triangulation. This framework is based on handlebody theory and provides the key concepts to understand the computational principles behind the algorithm, as well as, the basic operators for a roubst implementation.

The handlebody theory has been an important tool for geometric modeling and most recently for mesh compression [6]. When used in conjunction with stellar theory, it forms the basis for atomic operations on manifolds with or without boundary [11].

The rest of the paper is structured as follows: Section 2 gives an intuitive description of the advancing front triangulation algorithm. Section 3 reviews the main concepts of the handlebody theory. Section 4 shows the relation of handlebody theory with the advancing front algorithm. Section 5 shows how handlebody operators can be used in a simple and robust implementation of the Ball-Pivoting advancing front triangulation. Finally, Section 6 concludes the paper with examples of triangulated data-sets generated with the Ball-Pivoting algorithm.

2 Advancing Front Algorithms

In surface triangulation using advancing front methods [2, 3], the mesh is constructed by progressively attaching triangles to the mesh boundary using some geometrical criteria (see Figure 1). The mesh boundary is composed of closed loops of piecewise linear curves. This set of boundary curves forms an advancing front which is the border between meshed and unmeshed regions of the surface. The iteration of the basic step of incorporating triangles to the mesh boundary results in a propagation of the front that terminates when the whole surface is covered by the mesh.

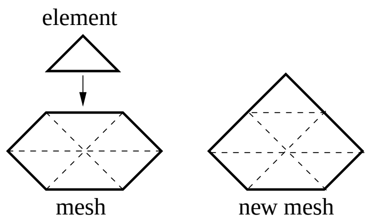
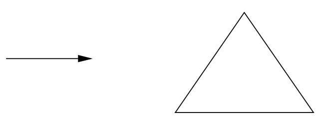


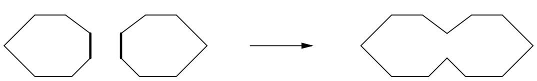
Figure 1: The basic idea of an advancing front method. Dashed lines represent interior edges and solid lines represent boundary edges.

Although the advancing front idea is simple, the algorithmic details of the method are complex. The main difficulty with this method lies in the need of merging different loops in the advancing front. Frequently the edges of a new triangle created in the iteration are "glued" with another edge of the front, changing the topology of the front. What we mean by topological change is the increase or decrease of the number of boundary curves. There are four types of topological changes that can happen:

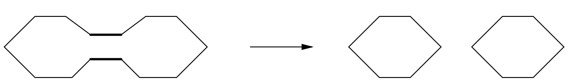
1. One curve is created:



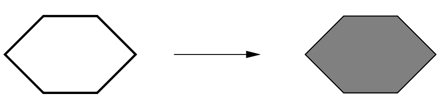
2. Two curves are joined into one curve:



3. One curve is splited into two curves:



4. One curve is closed:



These topological events are the object of study of the handlebody theory.

3 2D-Handlebody Theory

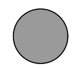
The Handlebody Theory [6] is a mathematical tool which will help us to understand better the topological changes in the mesh construction of a surface. It also provides necessary and sufficient conditions to deal with these topological changes. In the first instance we will introduce some necessary theory.

Let D^i be a disc with $i = 0, 1, 2$ dimension and ∂D^i its boundary.

Definition 1 *$H_\lambda = (A_\lambda, B_\lambda)$ is a handle with index $\lambda = 0, 1, 2$ such that $B_\lambda \subseteq \partial A_\lambda$ where $A_\lambda = D^\lambda \times D^{2-\lambda}$ and $B_\lambda = (\partial D^\lambda) \times D^{2-\lambda}$.*

According to definition above there exists only three types of *handles*:

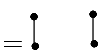
Type-0, $\lambda = 0$:

$A_0 = D^0 \times D^2 =$ 

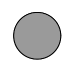
$$B_0 = (\partial D^0) \times D^2 = \emptyset$$

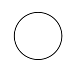
Type-1, $\lambda = 1$:

$A_1 = D^1 \times D^1 =$ 

$B_1 = (\partial D^1) \times D^1 =$ 

Type-2, $\lambda = 2$;

$A_2 = D^2 \times D^0 =$ 

$B_2 = (\partial D^2) \times D^0 =$ 

To attach a handle to a boundary of a 2-manifold S means to identify by homomorphism the set $B_\lambda \subseteq \partial A_\lambda$ with a subset I contained in the boundary.

Theorem 1 *For every manifold S there is a finite sequence of surfaces $S_{i=1\ldots N}$ such that $S_0 = \emptyset$, $S_N = S$ and the manifold S_i is obtained by attaching a handle to the boundary of S_{i-1}. This sequence is called the handlebody decomposition of S.*

For each handle type there is a different topological change in the surface:

- The type-0 *handle* creates a new connected component homeomorphic with a disc and a new boundary curve is created.

- If type-1 *handle* is attached to a suface, two cases may occur:

- It can be attached to two dijoint intervals in the same boundary curve. The curve is splited into two.
- It can be attached to intervals of different boundary curves in the surface. T he curves are joined into one.

- The type-2 case occurs when a boundary curve is closed.

To apply this theory in the construction of a surface we need a discrete representation of the surface and operators to deal with topological changes discussed above. This computational framework will be introduced in the next section.

4 Mesh Representation and Handle Operators

A mesh is defined as $M = (V, E, F, B)$ where E, V, F, B are the sets of vertices, edges, faces and boundary curves respectivelly. To retrieve topological information we can represent each edge with the well known *half edge* data structure [10].

One aspect which deserves attention is the difference between point and vertex. Their role is to represent the mesh geometrically and topologically, respectivelly. This detail is important because all boundary curves should be 1D-monifolds. In Figure 2 we show an example that the distinction between geometry and topology can solve ambiguities: one curve which is geometrically non-manifold but topologically may represent either one curve, or two curves.

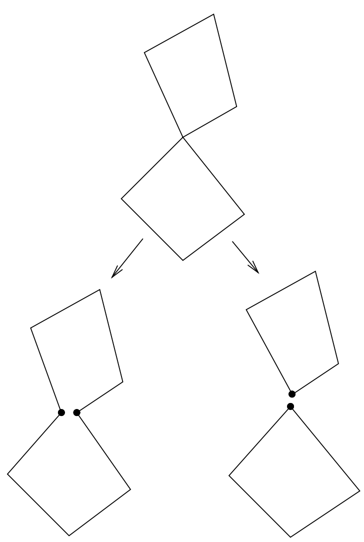


Figure 2: (top) - Geometric representation of one curve; (bottom)- Two topological representation of the same curve.

Observe that one *point* can be assigned to more than one *vertex* and for query purposes each point must keep a reference to one vertex that points to it (see Figure 3).

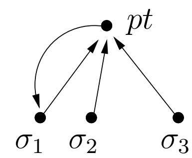


Figure 3: point/ vertex relationship.

In order to build the mesh iteratively for advancing front methods we can now use the handlebody theory and our mesh representation to introduce the handle operators and their API. Our purpose is to create a computational method to "mimic" the surface construction process described in theorem 1.

Let e_{ij} represent an edge, p_i, p_j its end points and σ_i, σ_j its end vertices.

Definition 2 *Two edges e_{ij} and e_{kl} are geometrically (resp. topologically) coincident if they have the same geometry (resp. topology), i.e., $\{p_i, p_j\} = \{p_k, p_l\}$ (resp. $\{\sigma_i, \sigma_j\} = \{\sigma_k, \sigma_l\}$.*

)

Definition 3 *Two edges e_{ij} and e_{kl} are topologically semi-coincident if $\#(\{\sigma_i, \sigma_j\} \cap \{\sigma_k, \sigma_l\}) = 1$.*

Definition 4 *Two edges e_{ij} and e_{kl} are topologically non-coincident if $\#(\{\sigma_i, \sigma_j\} \cap \{\sigma_k, \sigma_l\}) = 0$.*

Note that if two edges are topologically coincident then they are geometrically coincident. The converse is not true.

We can define four types of mesh handlebody operators:

1. The *handle operator* of type-0 creates a new triangle. It always generates a new connected component;

2. The *handle operator* of type-1 identify two geometrically coincident edges in the boundary but topologically non-coincident. The edges may be in the same boundary curve or in different boundary curves. In the first case the curve is splited into two curves. In the second one the curves are joined into one curve.

3. The *handle operator* of type-2 identify two geometrically and topologically coincent edges. This operator closes one curve.

4. The *homeomorphisms* identify two geometrically and semi-coincident edges. It performs a “zip”, i.e., the size of one boundary curve is decreased by two edges. Theres is no topolgy change in the curve.

Now we will define the *create* and *glue* topological operators. They will implement all handle operators described above and they can be used to construct a simplicial mesh in an advancing front algorithm.

The first routine, *create*(p_0, p_1, p_2), receives three points as input and it creates a triangle face, three edges, three vertices and one boundary curve. This API is equivalent to the handle operator of type-0.

The second routine, *glue*(e_{ij}, e_{kl}), receives two geometrically coincident edges and it treats internally the last three handle operators described above. It updates the mesh data structure by merging vertex and edge appropriately. It also maintains the list of boundary curves.

5 Ball-Pivoting: A Case Study

The Ball-Pivoting algorithm (BPA) builds a mesh by creating triangles which are circumscribed to an empty sphere [1] of constant radius r. These triangles are a subset of the alpha-shapes of the sample points [8].

As we said in the introduction, the BPA is an advancing front algorithm for surface reconstruction. For algorithms of this class it is necessary a criteria to choose a new element to be assigned to the mesh. In the case of the BPA, the criteria is a gometric step implemented in the ball_pivoting routine. This routine takes a boundary edge e_{ij} (pivot) and the sphere S of radius r which has e_{ij} as a cord. The ball is turned around e_{ij} until it touches a point p_k. This point will be a candidate to compose a new triangle with p_i and p_j (see Figure 4). To start the mesh construction there is another geometric routine, the find_seed_triangle, which returns three points circumscribed by an sphere.

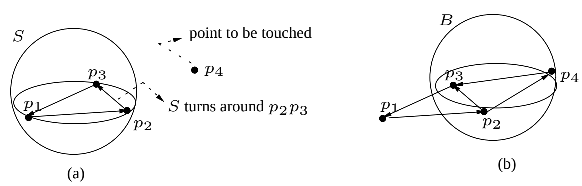


Figure 4: Ball Pivoting intuition. In the begining the front is composed by the polygonal $p_1p_2p_3$ (a). After pivoting the new polygonal of the front is $p_1p_2p_3p_4$ (b).

In this algorithm boundary edges have two classifications. They are classified as *active* or *inactive*. An active edge is one that can be used for pivoting. If it is not possible to pivot the edge is classified as inactive.

To avoid a non-manifold reconstruction, i.e., self intersections in the mesh, the BPA performs some verifications to the candidate point p_k after the pivoting. There are two manifold possibilities: the point is not yet in the mesh or the point is boundary. For these verifications we need the queries not_used() and on_boundary().

The above routines are essentially of geometric nature or queries. All the remaining routines used in the BPA are part of the handlebody API as defined in the last section. They create triangles and assign then to the mesh using the handle operators.

The BPA algorithm is shown below:

Algorithm Ball-Pivoting

while (not done)
  while ($e_{ij} \leftarrow$ get_active_edge(B) $\neq$ NULL)
    $p_k \leftarrow$ ball_pivot(e_{ij})
    if ($p_k \neq \emptyset$) and (not_used(p_k) or on_boundary(p_k)) **then**
      create(p_i, p_j, p_k)
      glue(e_{ij}, e_{ji})
      if $e_{ki} \in B$ **then**
        glue(e_{ik}, e_{ki})
      end if
      if $e_{jk} \in B$ **then**
        glue(e_{kj}, e_{jk})
      end if
    else
      mark-as-inactive(e_{ij})
    end if
  end while
  if (p_i, p_j, p_k)=find_seed_triangle() **then**
    create(p_i, p_j, p_k)
  end if
end while

Observe no algoritmo acima que, exceto quando uma nova componente conexa eh criada pela rotina find_seed_triangle apos a operacao create, a operacao glue, cujo proposito eh ligar arestas geometricamente coincidentes, eh chamada obrigatoriamente uma vez[2] e duas vezes condicionadas a existencia do par geometricamente coincidente nas curvas de bordo.

O caso interessante eh quando acontecem todas as tres chamadas. Nesse caso uma curva de bordo triangular esta sendo fechada i.e. espera-se que algum operador handle do tipo-2 eh tratado internamente em alguma das tres chamadas, mais especificamente na terceira. De fato eh possivel mostrar que o operador handle do tipo-2 acontece se e somente se

[1] i.e. there is no sample points inside the sphere

[2] De fato, a aresta pivoteada ja eh geometricamente coincidente.

tres operacoes glue sao chamadas num mesmo passo de um algoritmo de frente. Veja na figura 5. um interpretacao pictorica deste fato.

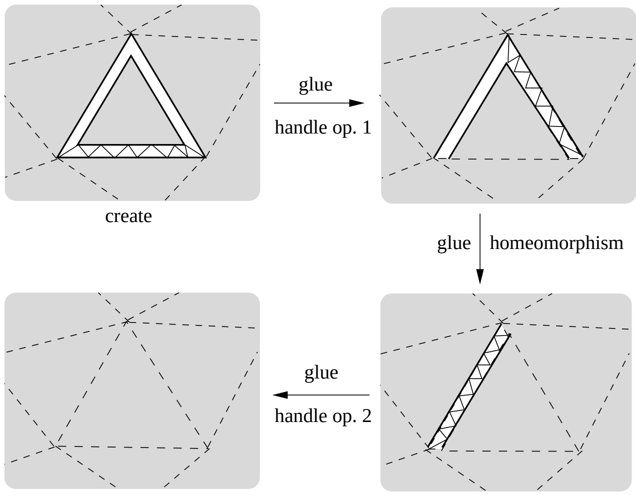


Figure 5: fgafga

6 Discussion

We implemented the BPA using the handle operators as described in this paper. We now show the results of triangulating two point clouds datasets: the Caltech head model and the hand.

The model in Figure 6 is a range scan of a clay head obtained from Caltech consisting of 38000 samples. We show the point cloud dataset and the triangulation generated by the Ball-Pivoting algorithm. The holes are parts of the surface occluded from the scanner.

Figure 7 depicts a dataset of hand bones, containing 65000 sample points and the resulting triangulated surface.

In conclusion, we presented a topological framework based on handlebody theory for the implementation of advancing front triangulation algorithms. Our analysis also provided a comprehensive interpretation of the Ball Pivoting algorithm, more specifically, regarding its topological aspects. This framework can be easily applied to the other advancing front algorithms. The basic difference lies on the geometric criteria step used to add triangles to the mesh.

Acknowledgements

This research has been developed in the VISGRAF Laboratory at IMPA. VISGRAF is sponsored by CNPq, FAPERJ, FINEP and IBM Brasil.

References

[1] F. Bernardini and J. Mittlelman and H. Rushmeir and C. Silva, *The ball-pivoting algorithm for surface reconstruction* ,IEEE Trans. Visual. Comput. Graphics , vol. 5, n. 8, pp. 349-359, (1999).

[2] S.H. Lo, *A New Mesh Generation Scheme for Arbitrary Planar Domains*, nternational Journal for Numerical Methods in Engineering, vol. 21, pp. 1403-1426, (1985).

[3] P.L. George and E. Seveno,*The Advancing Front Mesh Generation Method Revisited*, International Journal for Numerical Methods in Engineering, vol. 37, pp. 3605-3619, (1994).

[4] H. Lopes, J. Rossignac, A. Szymczack, A. Safanova and G. Tavares, *Edgebreaker: A simple algorithm for surfaces with handles*, Proceedings of the seventh ACM SIGGRAPH Symposium on Solid Modeling and Its Applications, pp. 108-117, (2002).

[5] E. Medeiros, *Algoritmo ball-pivoting: contextualização e estado da arte*, Master Thesis at IMPA, (2003).

[6] C. P. Rourke and B. J. Sanderson, *Introduction to piecewise linear topology*, Springer Study Edition, (1982).

[7] M. Levoy, J. Davis, S. R. Marshner, M. Garr *Filling holes in complex surfaces using volumetric diffusion*, 13th International Symposium on 3D Data Processing Visualization and Transmission , June 2002, Padova, Italy.

[8] H. Edelsbrunner and E. P. Mucke, *Three-dimensional alpha shapes*, ACM Trans. Graph.,Jan., vol. 10, n. 1, pp. 43-72, (1994).

[9] F. Bernardini and Holly Rushmeier *The 3D model acquisition pipeline*, Computer Graphics forum, vol. 21, n. 2, pp. 149-172, 2002.

[10] B. Baumgart, *A polyhedron representation for computer vision, AFIPS Conf. Proc.*,pp. 589-596, (2002).

[11] L. Velho, H. Lopes, G. Tavares, *Mesh Ops*, preprint.

[12] E. Mencl and H. Muller, *Interpolation and approximation of surfaces from three-dimensional scattered data points*, State of the Art Reports, ACM Eurographics 98, pp. 63-69.

[13] H. Hoppe and T. DeRose and T. Duchanp and J. McDonald and W. Stuetzle, *Surface reconstruction from unorganized points*, ACM Computer Graphics, vol. 26, pp. 71-78,(1992).

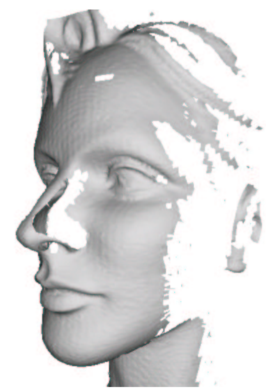
[14] J. D. Boissonant, *Geometric structures for three-dimensional shape representation*, ACM Transactions on Graphics, vol. 3, n. 4, pp. 266-286, (1984).

[15] H. K. Zhao and S. Osher and B Merriman and M. Kang, *Implicit, non-parametric shape recontruction from unorganized points using variational levelset method*, omputer Vision and Image Processing,(2002).

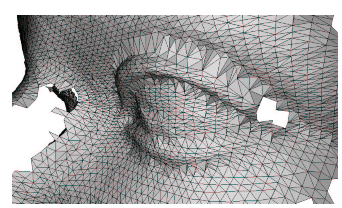
[16] R. Mencl and H. Muller. *Interpolation and approximation surfaces from three-dimensional sccatered data points.*, State of the Art Reports, Eurographics, pp. 51-67, (1998).

[17] E. Hartmann. *A marching method for the triangulation of surfaces.*, The Visual Computer, Volume 14 Issue 3, pp. 95-108, (1998).

(a)

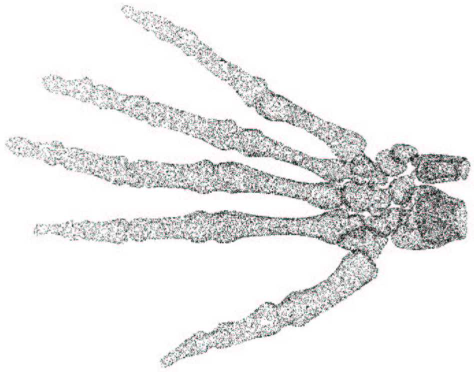


(b)

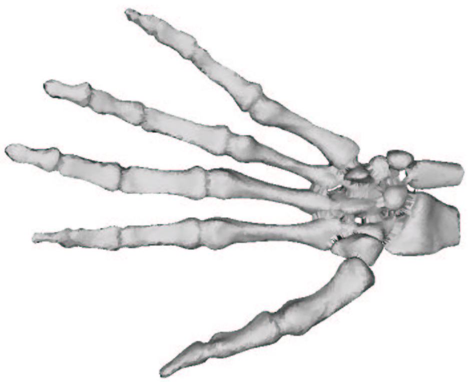


(c)

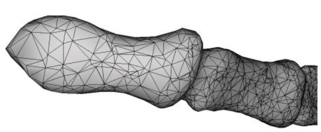
Figure 6: Head with 38000 points: (a) samples; (b) surface; (d) detail of right eye.



(a)



(b)



(c)

Figure 7: Hand with 65000 points: (a) samples; (b) surface; (d) detail of middle finger.