Automatic surface reconstruction from point sets in space

Marco Attene and Michela Spagnuolo

Istituto per la Matematica Applicata, Consiglio Nazionale delle Ricerche, Genova, Italy

Abstract

In this paper an algorithm is proposed that takes as input a generic set of unorganized points, sampled on a real object, and returns a closed interpolating surface. Specifically, this method generates a closed 2-manifold surface made of triangular faces, without limitations on the shape or genus of the original solid. The reconstruction method is based on generation of the Delaunay tetrahedralization of the point set, followed by a sculpturing process constrained to particular criteria. The main applications of this tool are in medical analysis and in reverse engineering areas. It is possible, for example, to reconstruct anatomical parts starting from surveys based on TACs or magnetic resonance.

1. Introduction

Automatic techniques for surface reconstruction play an extremely important role in a variety of applications, such as computer vision or reverse engineering. The main difficulties involved are caused by the use of automatic digitizing systems, such as laser-range scanners or digitizing machines, that generate a set of points belonging to the object surface. Digitizing machines usually perform the sampling along predefined directions, sections or profiles, which induce at least a partial order in the data set. Conversely, laser scanning techniques produce data sets without any particular spatial organization among points.

The methods developed have mainly been defined in a case by case manner in order to exploit the partial structure of the data. The use of shape-based reasoning techniques is especially efficient in this context since several structural features of the underlying surface may be easily detected by analyzing the geometric configuration of points along the sampling directions. The reconstruction may therefore be made a posteriori with respect to the recognition of a rough shape skeleton, which has the twofold advantage of guiding the reconstruction itself and allowing the insertion of shape constraints in the surface model [1,2,3].

Methods developed for unorganized points, that is, data sets without any specific spatial distribution, are generally based on the construction of neighboring relationships among points and on local piece-wise approach to surface fitting. Obviously, reconstruction processes may result in different surface models, such as triangulation [4,5,6,7] , B-spline patches [8,9], or implicit surface description [10].

The design of an algorithm for the reconstruction of surfaces is not a simple problem. Some methods have been proposed in the scientific literature, but almost all of them are restricted in terms of applicability spectrum:

- Some algorithms compute an approximation of the original surface by generating a new, denser set of points. This kind of approach is often used in those cases where an already very dense starting point set is available. The disadvantage lies in the necessity of a further step for surface generation, a process that often involves the computation of huge quantities of information.
- Generally, the reconstruction of particularly sharp edges and high curvature areas poses great difficulties for most algorithms.
- Sometimes the set of points must respect conditions like uniform density if the algorithm is to reconstruct the surface correctly [5,10].
- The topology of the objects that can be reconstructed often limits the applicability of the algorithm. For example, many algorithms can only reconstruct objects without holes [5,6,11].
- Some algorithms depend on a global parameter, which often calls on the user to make repeated adjustments in order to obtain a good result [12].
- Many *surface reconstruction* algorithms are limited in usage because they require excessively high computation times [7,13].

The solution presented in this paper generates a closed two-manifold surface made of triangular faces, without limitations on the shape or genus of the original solid. The proposed method is based on constrained sculpturing of the Delaunay tetrahedralization of the data points. It makes use of two geometric structures, the Euclidean minimum spanning tree and the Gabriel graph [6], to transform the Delaunay tetrahedralization of the object in its boundary surface triangulation.

The reminder of the paper is organized as follows. First, the problem of surface reconstruction is posed in formal terms, and basic notions as well as relevant previous work are described. Then the proposed method is described in detail, and some implementation issues are discussed. Examples and execution times are also presented. Finally, some conclusions are drawn and future developments are described.

2. Problem statement and previous work

From a mathematical point of view, a surface in the Euclidean three-dimensional space R^3 is defined as a two-dimensional manifold that is compact, connected and orientable. In other words, we might say that a surface is a "continuous" subset of points in R^3 which is locally two-dimensional. A surface may have a border, when the boundary is not empty, or it may be closed, when the boundary is empty. The problem of surface reconstruction can therefore be formalized as follows: given a set of points $\boldsymbol{S} = \{ \boldsymbol{P_i}=(x_i,y_i,z_i) \ / \ (x_i,y_i,z_i) \in \boldsymbol{M} \subset R^3 , \ i=1\ldots k, \ \boldsymbol{M}$ *surface in* $R^3\}$ find a surface $\boldsymbol{M'}$ which interpolates or approximates $\boldsymbol{M}$, using the data set $\boldsymbol{S}$ and information on the sampling process of $\boldsymbol{M}$.

Obviously, the reliability of the reconstructed surface depends on the *amount* of information available about the surface, whatever method is used to perform the reconstruction. In other words, no algorithm can accurately reconstruct a sub-sampled surface. To understand why this is actually not a limitation but a necessary condition, let us consider the problem of mono-dimensional signal reconstruction, for example sound waves. In this case, the Shannon sampling theorem proves that it is not possible to reconstruct a band-limited signal if its sampling frequency is below a fixed threshold value. In a similar way, it is not possible to reconstruct a surface accurately if its sampling point set is insufficiently dense. While the sampling theorem is intended for uniformly sampled signals, surface sampling may have different spatial distributions. In this sense, the formal framework proposed by Hoppe [13] et al. provides a good formalization of the requirements needed for point density: given a surface $\boldsymbol{M}$, its sampling $\boldsymbol{S_M}=\{ \boldsymbol{P_i}=(x_i,y_i,z_i,) \ / \ i=1\ldots k\}$ is said to be ρ-dense if every sphere with radius ρ and center on $\boldsymbol{M}$ contains at least one point of $\boldsymbol{S_M}$. Hence, it makes no sense to attempt reconstruction of surface shape features whose "dimension" is less than the sampling density associated to $\boldsymbol{S_M}$. This statement simply reflects Shannon's sampling theorem applied to surfaces.

Therefore, making no assumptions about the spatial distribution of the data points, we may say that the information contained in S_M is represented solely by the point position in space. The approaches used to solve this problem can be divided into two main classes: interpolation and approximation methods. Using the interpolation strategy, the reconstructed surface will preserve the original data set, that is, the measured points will also belong to the reconstructed surface. Interpolation methods can be further classified as *global* or *local* according to the degree to which each point is considered to influence the reconstruction of the surface at distant locations: in global methods, all points are used to define the interpolant, while in local methods only nearby points are used to compute a "piece" (or patch) of the whole surface. Global methods are rarely used, as the large volume of data involved generally causes high computational complexity. Piece-wise fitting, instead, is a much more flexible procedure.

The work presented in this paper approaches the reconstruction problem using a *local* strategy for defining a triangular mesh which interpolates the vertices of S_M. The basic idea is to take into account the juxtaposition of points in space; more precisely, it is considered *preferable* to connect two points in the final mesh with an edge if they are spatially *close*.

The concept of point neighborhood has been widely studied in the field of computational geometry, and efficient algorithms exist that compute solutions to a number of known problems [14] (*closest points, all nearest neighbors, Euclidean minimum spanning tree,* etc*)*. The solutions to these problems can be efficiently represented by graphs in which pairs of points are linked by an edge if and only if the pair respect the problem condition. Some of these graphs have properties which are useful for surface reconstruction. In particular, the Euclidean minimum spanning tree has been used by several authors as a first step in the reconstruction process [7,13].

For the sake of clarity, let us describe some of these basic geometric structures by giving the following definitions [6], where $\boldsymbol{P}=\{P_i=(x_i,y_i,z_i) \ / \ P_i \in R^3\}$ denotes the set of data points, and the metric used is the classical Euclidean distance in R^3.

- *Nearest neighbour graph, NNG*
  The nearest neighbor graph of $\boldsymbol{P}$ is the maximal graph NNG($\boldsymbol{P}$)=($\boldsymbol{P},\boldsymbol{E}$) such that $\boldsymbol{E} \subseteq \boldsymbol{P}\times\boldsymbol{P}$ and $\boldsymbol{E}$={ $e_k=(P_i,P_j)$ k=1,…n / P_j is the point of $\boldsymbol{P}$ closest to P_i }

- *Euclidean minimum spanning tree, EMST*
  The Euclidean minimum spanning tree $\boldsymbol{P}$ is the maximal tree EMST($\boldsymbol{P}$)=($\boldsymbol{P},\boldsymbol{E}$) such that $\boldsymbol{E} \subseteq \boldsymbol{P}\times\boldsymbol{P}$ and $\boldsymbol{E}$={$e_k=(P_i,P_j)$ k=1,…n / $\sum l(e_k)$ is minimum, where $l(e_k) = |P_i-P_j|$ }

- *Gabriel graph*
  The Gabriel graph of $\boldsymbol{P}$ is the maximal graph GG($\boldsymbol{P}$)=($\boldsymbol{P},\boldsymbol{E}$) defined by $\boldsymbol{E} \subseteq \boldsymbol{P}\times\boldsymbol{P}$ and $\boldsymbol{E}$={ $e_k= (P_i,P_j)$ k=1,…n / the smallest sphere for P_i and P_j does not contain any other point of $\boldsymbol{P}$ }

- *Delaunay tetrahedralization, DT*
  The Delaunay tetrahedralization of **P**, *DT(***P***)* is the maximal set of tetrahedra $\boldsymbol{T}\subseteq\boldsymbol{P}^4$,$\boldsymbol{T}$={ $t_k=(P_{k1},P_{k2},P_{k3},P_{k4})$ k=1,…n *}* such that*:*
  - ✓ $\forall \ P_i \in \boldsymbol{P}$, P_j is a vertex of some $t_k \in \boldsymbol{T}$**;**
  - ✓ $\forall \ t_k , \ t_n \in \boldsymbol{T}$, either their intersection is empty or they intersect at a common face, edge or vertex;
  - ✓ $\forall \ t_k \in \boldsymbol{T}$, the sphere circumscribing t_k does not contain any other point of $\boldsymbol{P}$.

We note that the boundary faces of DT(***P***) interpolate a subset of the data points and therefore define an initial step for the solution to the surface reconstruction problem. Moreover, if we denote with EDT(***P***) the set of edges of DT(***P***), the following inclusion relation holds:

$$NNG(\boldsymbol{P}) \subseteq EMST(\boldsymbol{P}) \subseteq GG(\boldsymbol{P}) \subseteq EDT(\boldsymbol{P})$$

More precisely, the previous relation is true only if the related graphs are unique for a given data set ***P*** but, for simplicity's sake, this point will not be discussed here.

Several methods use the DT(***P***) structure as initial approximation of the surface shape and iteratively remove inner tetrahedra and portions of the DT(***P***) which are judged to be external to the real object surface. This process, called *sculpturing*, was introduced in 1984 by Boissonat [5]. It consists of an iterative removal of some tetrahedra from the DT until all the vertices lie on the boundary. One very interesting characteristic of this approach is the possibility to maintain a coherent data structure at each step[I]. Moreover, by some simple reasoning, it is possible to implement very efficient sculpturing algorithms, O(**t**log**t**), where **t** is the number of tetrahedra of the starting DT.

It is not difficult to fix some simple rules that each sculpturing algorithm must respect:

- Removal of a tetrahedron with 3 boundary triangles can irreversibly disconnect a vertex from the current boundary, therefore removal of such a tetrahedron is never allowed.
- A tetrahedron with exactly two faces on the boundary with a common edge **e** can be removed only if the edge opposed to **e** is not already on the boundary.
- A tetrahedron with only one face on the boundary can be removed only if the vertex opposed to that face is not already on the boundary.

The order in which tetrahedra are removed influences the final result, therefore it is necessary to determine a convenient sorting criterion. A number of different methods have been proposed in this regard. For example, Boissonnat uses a criterion based on the minimum change of the surface area, while O'Rourke [15] uses a mathematical tool, the *Voronoi Skeleton*, as a sorting criterion. Veltkamp [6,11] defines for each tetrahedron a value called γ-indicator; tetrahedra are eliminated by growing order of this parameter.

Unfortunately, each of the aforementioned methods has its limitations, such as the need for user interaction, the impossibility to reconstruct solids with holes, the creation of unaesthetic surfaces, excessive complexity, and so on.

[I] At each step the boundary is a 2-manifold

The sculpturing method proposed in this paper integrates the standard rules for getting a 2-manifold surface with additional constraints. These constraints are a combination of two different criteria whose advantages are coupled, eliminating the need for user-interaction to locally adjust the resulting surface. The constraints used for the sculpturing process are defined by the EMST and an Extended Gabriel hypergraph, which is defined in the next section.

Some authors take into account only a part of the constraints proposed here. For example, Mencl and Müller [7] developed a reconstruction method that, starting from the EMST, extends this graph to a so-called *surface description graph* using assumptions on the position and shape of edges. This algorithm, moreover, determines some shape characteristics before the surface is completely reconstructed and uses these as a support for completing the process. Unfortunately, the algorithm requires considerable computation time, in the order of hours for a few tens of thousands of points, and even though several assumptions are made, a good result cannot be guaranteed.

Another work relevant to our method is the sculpturing proposed by Veltkamp [6,11], which starts from DT(***P***) and sculptures tetrahedra away in a very efficient fashion. This method generates a 2-manifold surface interpolating all vertices, but may produce *unaesthetic* surfaces with long, thin tetrahedra. This problem is caused by the removal criteria used for tetrahedra, which is based on the concept of γ-indicator. Moreover, the reconstructed surface always has genus 0, therefore objects with through holes cannot be accurately reconstructed.

3. The proposed algorithm

The proposed method stems from analysis of the criteria used by various authors, and can be seen as a hybrid approach based on sculpturing and on the use of some interesting properties of geometric graphs, as defined in the previous section. The EMST is used as a constraint during the sculpturing of the Delaunay tetrahedralization of the data set, and in addition another constraint is used, the so-called *Extended Gabriel Hypergraph* (EGH). Roughly speaking, given ***P*** as the initial point set, the algorithm starts with the generation of DT(***P***); then, tetrahedra are iteratively removed from DT(***P***), until all *vertices* lie on the boundary. The removal process is constrained to the EMST(***P***) and to the EGH(***P***), as explained in the following. The boundary of DT(***P***), simplified as previously described, defines the reconstructed surface.

Many authors use the EMST as a constraining or starting graph for surface reconstruction because its definition guarantees that the resulting edges are the shortest possible. Therefore, close points in the data set are likely to be linked in the graph, which is an important starting point for pursuing our aim. Moreover, since the EMST has a tree structure, there is always a path between the two vertices of

the graph. The Gabriel graph has not been widely used for surface reconstruction and is mainly considered as a self-standing entity. However, this graph gives a kind of indication about the *best interconnection* of points when used for the reconstruction of the boundary of a 2D data set [6,11].

3.1. Extended Gabriel Hypergraph

The concept of extended Gabriel hypergraph has been introduced to locate, inside the Delaunay tetrahedralization, those triangles that have a high probability of being close to the original surface.

Given P, and given the associated GG(P)=(P, E_{GG}), the EGH(P) is defined as EGH(P)=(P,E_{EGH},T) such that E_{EGH}, the edge set, is initially defined by E_{GG} while T, the triangle set, is initially empty. The final sets are constructively defined as follows:

- ✓ $\forall\ e_1, e_2 \in E_{GG}$, $e_1=(v_1, v_2)$ and $e_2=(v_2, v_3)$, if v_1, v_2 and v_3 are not aligned and if the smallest sphere for v_1, v_2 and v_3 does not contain any other points of ***P***, then $E_{EGH}= E_{EGH}\cup\{(v_1, v_3)\}$
- ✓ any cycle of three edges in E_{EGH} defines a new triangle in T.

Theorem: *If the Delaunay tetrahedralization DT(**P**) is unique, then all the triangles of the Extended Gabriel Hypergraph are triangles of the DT(**P**).*

$$EGH(\mathbf{P}) \subseteq DT(\mathbf{P})$$

Proof

Each triangle of EGH(**P**) satisfies one of the following two conditions:

1. It is made of three edges of GG(**P**).
2. The smallest ball touching its vertices does not contain other points of **P**.

In the first case, it is enough to consider the inclusion GG(**P**) $\subseteq$ DT(**P**) to assert that the triangle is in the DT.

In the second case, let's consider that if an empty ball touching the three vertices of the triangle exists, it can be enlarged, maintaining the condition that it touches the above three vertices and is empty, until it reaches the closest fourth vertex of **P**. In this way we have determined a tetrahedron **T** whose four vertices define an empty ball, **T** belongs to the DT(**P**) and the analyzed triangle is a face of **T**, in other words it is a triangle of the DT(**P**).

$\otimes$

The above theorem already represents a first important characteristic of the EGH, the inclusion in the Delaunay tetrahedralization.

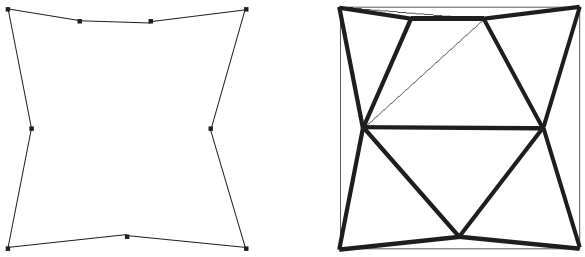

Figure 1: *Example of the boundary of a point set in 2D and corresponding Delaunay triangulation. In the DT the Gabriel graph edges are highlighted.*

The above figure shows how, in 2D, the Gabriel graph gives a rough approximation of the boundary to be reconstructed. The graphical representation of the three-dimensional case is shown in the following figure.

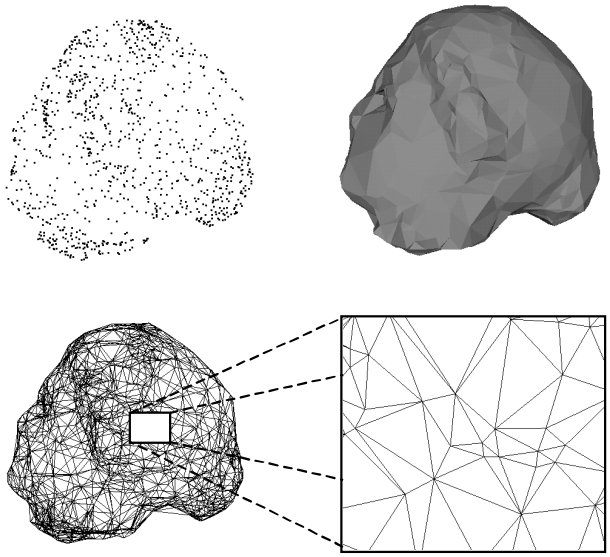

Figure 2: *A set made of 1079 points sampled from a head of Venus, its reconstruction and the Extended Gabriel Hypergraph.*

3.2. Constrained sculpturing

Insertion of constraints in a sculpturing algorithm requires a degree of caution. First of all it is necessary to define the basic structure of the algorithm:

1. Construction of the Delaunay tetrahedralization
2. Construction of a heap containing **removable** tetrahedra sorted with a **criterion**

3. While (∃ one vertex not lying on the boundary and the heap is not empty) {
4. T = root(heap); remove T from heap
5. if (T is **removable**) {
6. remove T from DT
7. insert each new **removable** tetrahedron into the heap

}
}

Chosing the *heap* for storing tetrahedra is justified by the quick insertion and removal of an element in this structure (O(log**n**), **n** = number of stored elements). Although at the beginning the heap only contains removable tetrahedra, the test (5) is necessary because as a result of previous removals a tetrahedron may become irremovable.

Constraints come into play in the definition of *removable* tetrahedron: **T** is classified *removable* if and only if all the following rules are respected:

- basic sculpturing rules (see Section 2)
- if **T** has the only face **t** on the boundary, **t** must not belong to EGH.
- if **T** has two faces on the boundary, these must not belong to EGH, moreover the common edge of the two faces must not belong to the EMST.

In such a way, the sculpturing proceeds without removing those elements that, according to EGH and EMST properties, have a high probability of belonging to the surface. It is possible that the constraint given by the EGH will prevent some vertices reaching the boundary; in this case, at the end of the *while* cycle, the heap is refilled with removable tetrahedra not constrained to the EGH. The above disadvantage is caused by a characteristic behavior of the EGH in the presence of badly sampled vertices (see Fig. 3).

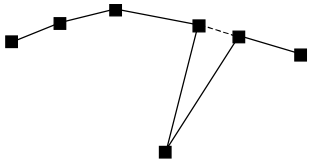

Figure 3: *Typical example of a vertex hidden by the EGH. It is likely that such vertices are originated by errors of the digitizing tool or by insufficient sampling.*

The **criterion** used for sorting tetrahedra into the heap may be any of those explained in the previous section. An additional criterion that is experimentally sound is the following:

Tetrahedra that have the longest edge on the boundary have to be removed first.

This kind of sorting is based on the observation that in the reconstruction of a well sampled object, the linkage of two very distant vertices is less probable than that of two very close vertices.

Until this point the algorithm is able to reconstruct surfaces of genus 0 (solids without holes). To extend the capabilities of the algorithm it is necessary to use a mathematical tool that locates the presence of a hole. Once again the EMST proves to be the best choice; to this end, the following algorithm can be considered, in which the *removable* condition is to be intended as **not constrained**:

1. Consider all the removable tetrahedra whose removal adds one edge of the EMST to the boundary.
2. Remove such tetrahedra.
3. If there exists an edge **e** of the EMST that is not on the boundary and it is possible to create a hole, *create a hole according to* **e,** start again with **constrained** sculpturing and, at the end, go to 1.

Let us consider a tetrahedralization with 2-manifold boundary. A simple analysis shows that the simplest solid which, if removed, produces a hole is a pseudo-prism with triangular bases consisting of 3 tetrahedra (see Figure 4).

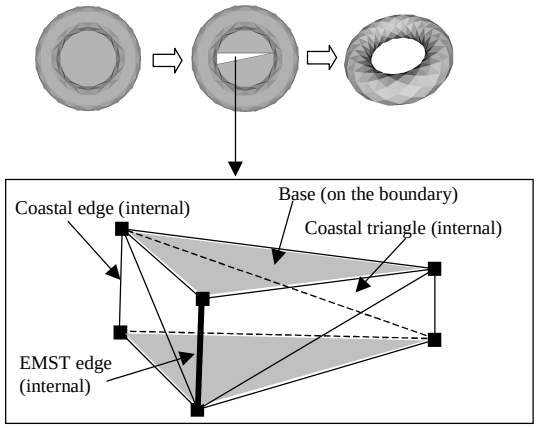

Figure 4: *Pseudo-prism made of 3 tetrahedra removed from a torus to create the hole.*

The simplest way to create a hole, therefore, is to remove a pseudo-prism whose bases, and only those, belong to the boundary. The criterion that determines whether and how to create a hole according to an edge **e** is the following:

Create a hole, if possible, according to ***e***

1. **e** ∉ boundary
2. Determine pseudo-prisms such that:
   - **e** is a coastal edge;
   - the six coastal triangles don't belong to the boundary;
   - the two bases belong to the boundary
3. Remove the pseudo-prism (the three tetrahedra that constitute it) with the longest edge on the boundary.

If the Euler-Poincaré formula is considered [16]:

$$v - e + f = 2(s - h)$$

it is possible to analyze the coherence of the discussed method. The removal of a pseudo-prism that respects the hypothesis mentioned above causes the insertion of six new edges on the boundary, the removal of two faces (the bases) and the insertion of six other faces (coastal triangles):

$$v - (e+6) + (f-2+6) = 2(1 - h')$$
$$2(1 - h) - 6 - 2 + 6 = 2(1 - h')$$
$$2(1 - h) - 2 = 2(1 - h')$$
$$2(1 - h - 1) = 2(1 - h')$$
$$2(1 - (h+1)) = 2(1 - h')$$
$$h' = h + 1$$

Those pseudo-prisms that meet the given hypothesis can be determined quite simply by adjacencies starting from **e**. The figure below shows an example of hole creation.

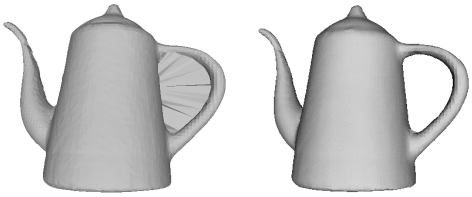

Figure 5: *Example of hole creation.*

The following pseudo-C algorithm summarizes the whole method. The predicate **removable** may or may not be constrained, so it has to be specified each time:

1. Generation of the DT, the EMST and the EGH
2. Construction of a heap; constraints are ON;
3. Fill the heap with all **removable** tetrahedra sorted by the longest boundary edge
4. Nv = number of vertices
5. Nbv = number of vertices on the boundary
6. Ne = number of EMST edges
7. Nbe = number EMST edges on the boundary
8. While (Nbv < Nv && heap ≠ ∅) {
9. T = root(heap); remove T from heap
10. if (T is **removable**) {
11. remove T from the DT
12. insert each new **removable** tetrahedron in heap
}
}
13. constraints are OFF; if (Nbv < Nv) GOTO 3
14. if (Nbe < Ne) {
15. Fill the heap with those **removable** tetrahedra whose removal adds an EMST edge to the boundary
16. while (Nbe < Ne) ..as the previous while
17. if (∃ **e** ∈ EMST: **e** ∉ boundary && it is possible to create a hole according to **e**) {
18. Create a hole according to **e**
19. constraints are ON; GOTO 3
}
}

3.3. Complexity analysis

Referring to the previous algorithm, we can analyze the computational complexity as follows.
Let **n** be the number of vertices and **t** the number of tetrahedra:

- (1) The determination of the DT requires O(**t**log**t**) operations, and in the worst case **t** = O(**n**^{2}) therefore the complexity O(**n**^{2}log**n**) [17, 18]. Given the DT, the EMST can be computed with O(**e**log**n**) operations, number of edges **e** = O(**n**^{2}) and from it O(**n**^{2}log**n**) [14]. The construction of the EGH from the DT can be done trivially in O (**t**log**t**) = O(**n**^{2}log**n**) operations.
- (2-3) Construction and initialization of the heap requires O(**t**log**t**) = O(**n**^{2}log**n**) operations.
- (while 8-12) Removal of the heap's root requires O(log**t**) operations and can produce at most three new tetrahedra to insert; the cost of this last operation is O(log**t**) operations. The whole cycle terminates in a time O(**n**log**t**) = O(**n**log**n**).
- (13) As before.
- (14-16) As before.
- (17-18) Searching **e** requires O(|EMST|) = O(**n**) operations. Computation of the pseudo-prisms analyzes tetrahedra that are incident in the two vertices of **e**, from that O(**n**).
- (19) The jump is done once for each hole created, so, if **h** is the number of holes in the solid, the block (3-19) terminates in a number of operations = O(**hn**^{2}log**n**).

The worst-case complexity is O(**hn**^{2}log**n**). It ought to be considered that the worst condition in which **t** = O(**n**^{2}) hardly ever occurs. Moreover, in practical cases the number of holes in the solid is very low. Analysis of computing timing during experiments, in fact, shows an average case complexity measurable in O(**n**log**n**) (see Table 1, Sec. 4).

4. Implementation

The algorithm has been implemented in C++ with OpenInventor, Motif and ViewKit libraries for visualization and graphical interface. Further optimizations have been introduced.

The main data structure explicitly stores vertices, edges and triangles. Tetrahedra are stored in a temporary list in the form of 4-tuples of vertex indexes; at the end of the reconstruction process this list is dismissed. Each edge is represented by a pair of vertex indexes whose order gives an orientation to the edge; in this way it is possible to compute the ET relation that associates all the triangles incident to

each edge from a single explicitly stored triangle. Indeed, each triangle stores its two incident tetrahedra and, by adjacencies, it is possible to turn the edge around to obtain the ET relation in optimal time.

The algorithm has been tested with different data sets, ranging in size from a few hundreds of points to many tens of thousands. Surfaces have been reconstructed starting from uniformly and non-uniformly distributed points, and from convex surfaces and surfaces of solids with holes, as shown by the examples in Appendix 1.

To compute the Delaunay tetrahedralization, the *Quickhull algorithm* [19,20] has been used. Extraction of the EMST starting from the DT has been implemented according to Prim's [14] method. Computation of the *extended Gabriel Hypergraph* has been implemented by the direct use of the definition and considering all the information given by the DT; specifically, the *empty ball* test only needs to analyze the adjacent vertices as follows:

- The smallest sphere for a triangle **t** is empty if, for each tetrahedron with face **t,** the vertex that doesn't belong to **t** is not in the sphere;
- The smallest sphere for an edge **e** is empty if, for each edge that shares a vertex with **e,** the vertex that doesn't belong to **e** is not in the sphere;

Simple observations have made it possible to implement the computation of the EGH in such a way that it terminates in a time that is proportional to the number of tetrahedra.

The following table shows how the required computing time grows with the growth of the number of input points. Experiments have been done on a PC Pentium III 450 MHz running the Linux 2.2.12 operating system and equipped with 128M RAM memory. Reported timings are expressed in seconds, and the time required to load the point set and save the VRML file is not included.

Number of points	Computing seconds
256	0
512	0
1024	1
2048	3
4096	5
8192	13
16384	23
32768	41
65536	83

Table 1: *Computing seconds required to reconstruct a solid without holes. Input and output times are not included.*

The following images show a comparison of some traditional sculpturing methods and the constrained method proposed in this paper. Implementation of the other methods was quite simple; in fact, all that is required is to modify the function that associates to each tetrahedron the key for sorting the heap and removing the test that imposes the constraints.

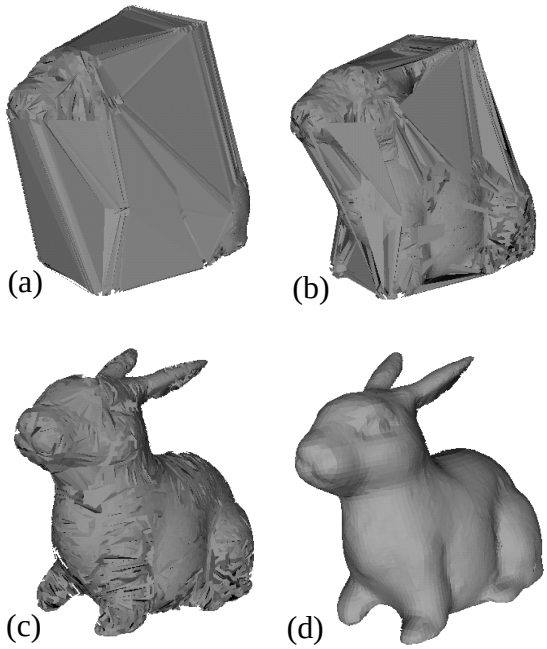

Figure 6. *Example: Minimal Area Change method (a), γ-indicator (b), Maximum Edge Length without constraints (c), EMST and EGH constrained Maximum Edge Length (d).*

5. Conclusions

The proposed algorithm represents a useful tool for surface reconstruction from a sampled point set of which nothing else but the point position is known. When additional information is available to the user, it is usually possible to obtain better results by way of specific methods that take into account that information.

The innovation introduced by this method is the simultaneous use of multiple criteria, which overcomes the limits of each method considered separately and, at the same time, takes advantage of all their potentialities. Specifically, the problem has been approached in a more general way by defining the reconstructed surface not only as a 2-manifold interpolating mesh, but also taking into account some desired shape properties. Consequently, the reconstruction turns out to be aesthetically *pleasant* in most practical cases. Two important sculpturing characteristics have been exploited: efficiency, with regard to required computation time, and the possibility of maintaining a coherent data structure at each step. The *Euclidean Minimum Spanning*

Tree has been used because it provides a good surface description. Moreover, the notion of *Extended Gabriel Hypergraph* has been introduced, which represents a valid *triangle-oriented* description of the surface to reconstruct.

We foresee the main applications of this tool will be developed in medical analysis and Reverse Engineering. A priori knowledge of some characteristics of the object to reconstruct can be used to improve reconstruction in a particular application area. Other future work may lie in method extension to the reconstruction of opened surfaces, simplification of the reconstructed geometric model, and finally the study of parallelization criteria for handling very large point sets.

Acknowledgements

This work has been partially supported by the EC Project Nr. F0069 "THETIS". Thanks are due to Dr. Bianca Falcidieno for her encouragement in writing the paper and to the members of the Computer Graphics group, IMA, for their helpful advice.

References

1. B. Falcidieno, M. Spagnuolo, "A shape abstraction paradigm for modeling geometry and semantics" *Proceedings of the International Conference on Computer Graphics International*, IEEE Computer Society Press, Los Alamitos, CA, June 1998.
2. A. Raviola, M. Spagnuolo, "Shape-based Surface Reconstruction from profiles for rapid/virtual prototyping", *Proceedings of the International Conference Numerization3D*, Paris, May 1999.
3. S. Biasotti, M. Mortara, "Topological Compression and Decompression using Reeb graphs", IMA Technical Report Nr. 18/99, December 1999.
4. M. Attene, "Surface reconstruction from clouds of points", IMA Technical Report Nr. 17/99, December 1999.
5. J.-D. Boissonnat, "Geometric structures for three-dimensional shape representation", *ACM Transactions on Graphics*, 3(4):266-286, October 1984.
6. R.C. Veltkamp, "Closed object boundaries from scattered points", PhD Thesis, Department of Interactive Systems, Center for Mathematics and Computer Science, August 1992.
7. R. Mencl, H. Müller, "Graph based surface reconstruction using structures in scattered point sets", *Proceedings of Computer Graphics International (CGI '98)*, IEEE Computer Society Press, 298-311, 1998.
8. M. Eck, H. Hoppe, "Automatic reconstruction of B-spline surfaces of arbitrary topological type", *Computer Graphics Proceedings*, SIGGRAPH '96, Annual Conference Series.
9. B. Guo, "Surface reconstruction: from points to splines", *Computer Aided Design*, Vol. 29(No. 4):269-277, April 1997.
10. E. Bittar, N. Tsingos, M.-P. Gascuel, "Automatic reconstruction of unstructured data: Combining a medial axis and implicit surfaces", *Computer Graphics Forum*, 14(3):457-468, 1995. Proceedings of EUROGRAPHICS '95.
11. R.C. Veltkamp, "3D computational morphology", *Computer Graphics Forum*, 12(3):115-127, 1993. EUROGRAPHICS '95.
12. H. Edelsbrunner, E. Mücke, "Three dimensional alpha-shapes", *ACM Transactions on Graphics*, 13(1):43-72, 1994.
13. H. Hoppe, T. DeRose, T. Duchamp, J. McDonald, W. Stuetzle, "Surface reconstruction from unorganized points", *Computer Graphics (SIGGRAPH '92 Proceedings)*, 26(2):71-78, July 1992.
14. J. O'Rourke, "Computational Geometry in C", ISBN 0-521-44592-2/Pb, *Cambridge University Press*, 1994.
15. J. O'Rourke, H. Booth, R. Washington, "Connect-the-dots: a new heuristic", *Computer Vision, Graphics, and Image Processing*, 39, 1987, 258-266, 1987.
16. M. Mäntylä, "An introduction to solid modeling", ISSN 0888-2096, Computer Science Press, 1988.
17. K.Q. Brown, "Voronoi diagrams from convex hulls", *Information Processing Letters*, 9(5):223 – 228, 1979.
18. R. Seidel, "Constructing higher-dimensional convex hulls at logarithmic cost per face", *Proceeding of the 18th Annual ACM Symposium of Theory of Computing*, ACM press, 404 – 413, 1986.
19. C.B. Barber, D.P Dobkin, H. Huhdanpaa, "The quickhull algorithm for convex hulls", *ACM Transactions on mathematical software*, 22(4):469-483, December 1996.
20. G. Leach, "Improving Worst-Case Optimal Delaunay Triangulation Algorithms", *Proceedings of the 4th Canadian Conference on Computational Geometry*, 340-346, June 1992.

Appendix 1.

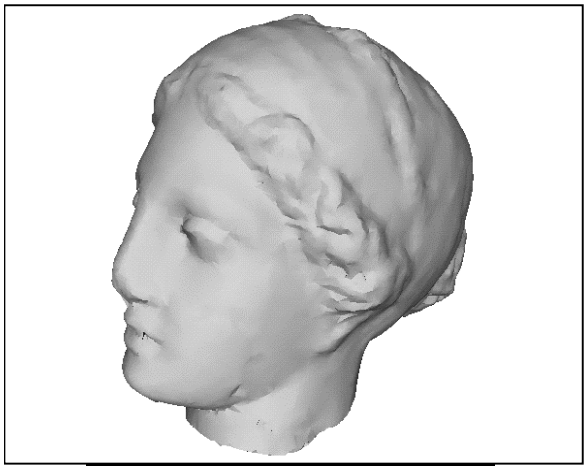

Figure 7: *Venus – 15000 points*

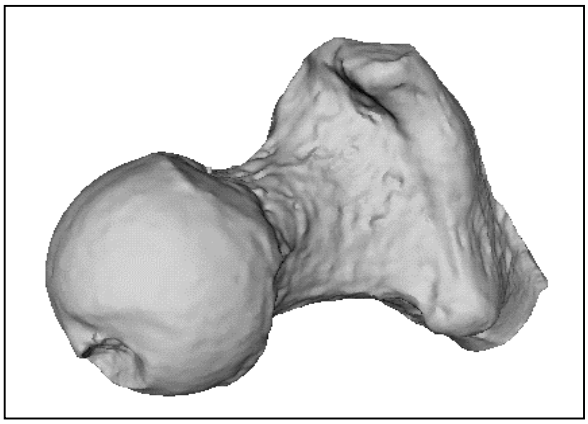

Figure 8: *Ball Joint – 30000 points*

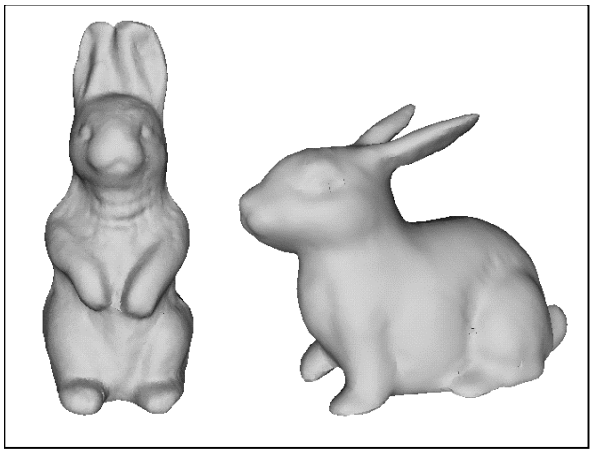

Figure 9: *Bunny – 15000 points, Rabbit – 7600 points*

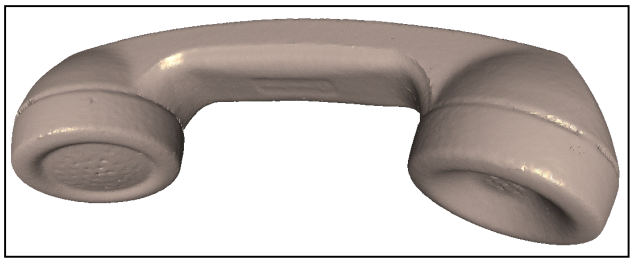

Figure 10: *Handphone – 44000 points*

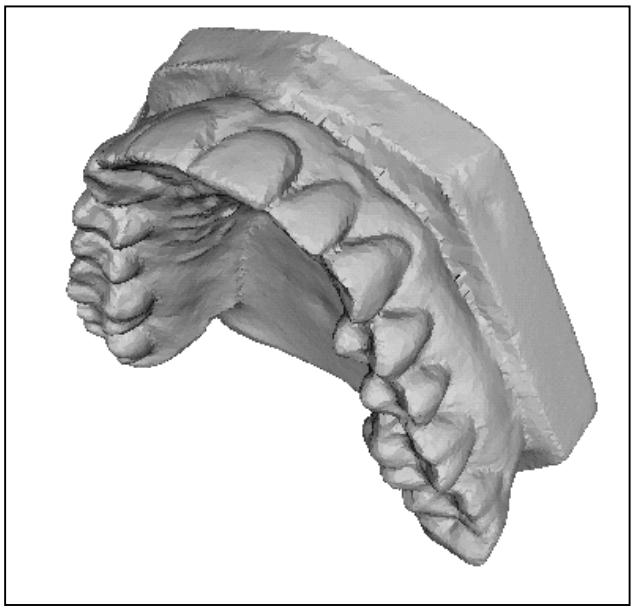

Figure 11: *Teeth – 29500 points*

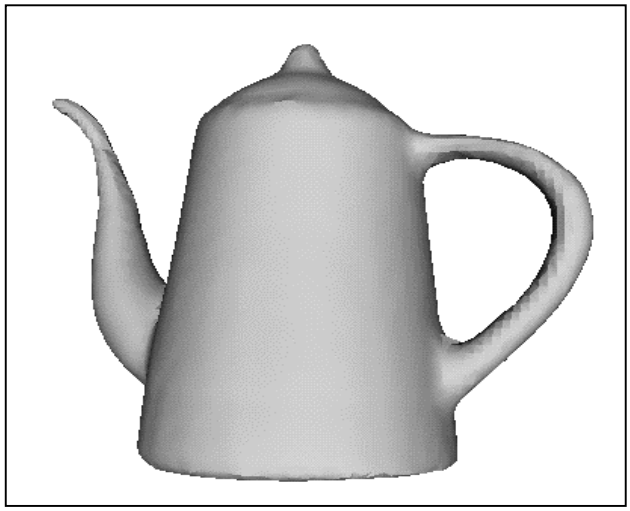

Figure 12: *Tea pot – 7300 points*