Triangle-Based Surface Models

Leila De Floriani, Paola Magillo
Information and Computer Science Department (DISI)
University of Genova
Via Dodecaneso, 35, 16146 Genova (Italy)
Tel.: +39-10-3538033, Fax: +39-10-3538028
Email: {magillo, deflo} @ disi.unige.it

Silvia Bussi
Elsag Bailey
R & D Department
Via Puccini, 2, 16154 Genova (Italy)
Tel.: +39-10-6582748
Email: sbussi @ eslag.it

Abstract

The problem of reconstructing a digital model of a surface from a finite set of sampled points is fundamental in many different application domains, including computer graphics, geographic data processing, computer vision and computer aided design. A triangulated surface model is often used, because of the possibility of including surface features, and of the simplicity of the topological structure.

The definition of a triangle-based surface model relies on the concept of triangulation. In this paper, we discuss the basic properties of triangulations, Delaunay triangulations, constrained and conforming triangulations. We present a survey of algorithms for building these kinds of triangulations, which represent the first step in the construction of a surface model.

A special attention is given to the surface reconstruction problem in $2\frac{1}{2}$ dimensions, which is connected to digital terrain modeling in geographic information systems. The more general problem of reconstructing the bounding surface of a solid object from three dimensional scattered data is also considered, and a brief survey of the main approaches proposed in the literature is presented.

1 Introduction

Surface modeling plays an important role in many different application domains, including computer graphics, geographic data processing, computer vision and computer aided design.

The general problem can be stated as reconstructing a model of a surface by interpolating a finite set of points in space belonging to it. Defining a convenient relational model for encoding the neighborhood relations among the data points is necessary to build an effective surface representation.

The surface reconstruction problem in $2\frac{1}{2}$ dimensions is strongly related to applications to

digital terrain modeling in geographic information systems. In this case, the problem can be mathematically stated as that of interpolating a bivariate function when its values are given either at a uniformly spaced grid or at a set of irregularily distributed points. The surface to be reconstructed is the graph of a bivariate function, and thus the problem reduces to that of computing a 2D tesselation of the domain of such function.

A 2D tesselation of the function domain, provides a relational model for encoding the neighborhood relations among the projections of the data points on the $x - y$ plane. A triangle-based representation is a very flexible surface model: it approximates the surface by means of a network of planar, non-overlapping and irregularily shaped planar facets. Even when the surface is sampled at regular intervals in both coordinates, using a triangle-based surface model turns out to be advantageous, since such model can adapt to the changes in the surface and include a set of "surface-specific" points and lines, which characterize the surface independently of the data sampling. The construction of a triangle-based approximation reduces to the problem of computing a straight-line plane graph, termed plane triangulation, in which the data projections are joined by straight-line segments intersecting only at their endpoints and the projections of the given straight-line segments are included in a subset of its edges.

An arbitrary triangulation may not represent, in general, an acceptable solution for numerical interpolation because of the elongated shape of its triangles. Intuitively, a "good" triangulation is one in which triangles are "as much equiangular as possible", so as to avoid thin and elongated triangular facets. Delaunay triangulation is optimal with respect to such requirement, and thus has been extensively used as a basis for surface models.

This article describes the basic properties of standard Delaunay triangulations and of Delaunay triangulations constrained by and conforming to a given set of straight-line segments. It also contains a survey of algorithms for building these kinds of triangulations.

When a large number of sampled points is available, a triangulation joining all the data can be highly inefficient in storage and for search and retrieval operations. Approximated models based on triangular grids have been used in the past. Such models are built on the basis of a restricted subset of the data, chosen in such a way to provide a representation of the surface within a certain error tolerance. Approximated surface models are a good data compression mechanism, but they give an approximation at a predefined level of accuracy. Multiresolution surface models, on the contrary, encode a set of increasingly finer surface representations, thereby allowing efficient access and manipulation of surface data at different levels of abstraction. A brief survey on multiresolution surface models is included in this paper.

In general, the problem of reconstructing a boundary model of a solid object from a finite set of data in 3D space finds applications in computer graphics, computer aided design, computer vision and robotics. As in the $2\frac{1}{2}$ case, input data may consist of points or straight-line segments lying on the object boundary. In addition, a set of plane contours, obtained by intersecting the object with a collection of planes, may also be given.

Many existing algorithm face the boundary reconstruction problem in an undirect way, building first a 3D representation of the object volume by using a tetrahedralization of the input data. We briefly discuss Delaunay tetrahedralization, possibly constrained or conforming, which represent an extension to three-dimensions of (constrained or conforming) Delaunay triangulations. More general geometric structures (such as the γ-neighborood graph, which extends the concept of Delaunay triangulation) are also introduced.

This paper is organized as follows. In Section 2, we describe the basic properties of standard Delaunay triangulation and of Delaunay triangulation constrained by and conforming to a given

set of straight-line segments, which are the basis for digital models of two-and-half dimensional surfaces. In Section 3, we provide a survey of algorithms for building the above three kinds of triangulations. In Section 4, we introduce multiresolution models of surfaces in $2\frac{1}{2}$D. In Section 5, we consider the problem of reconstructing the bounding surface of a solid object, and provide a survey of existing approaches to its solution as well as a description of geometric structures used in such algorithms.

2 Two-and-Half Dimensional Surfaces

A two-and-half dimensional surface is the graph of a bivariate real function $z = \phi(x, y)$ defined over a connected subset D of the $x - y$ plane. The surface reconstruction problem, faced in this survey, consists of computing a *Digital Surface Model* (DSM), which is a discrete approximation of a surface, built by interpolation based on a finite set of data points sampled on it. Because of the one-to-one correspondence between points of the surface and points of the two-dimensional domain D, building a DSM reduces to computing a plane subdivision of D into regions, and to establishing an analytic expression for the correspondence between each region and the face defined over it.

More formally, a *Digital Surface Model* (DSM), built on a finite set $\mathcal{S}$ of sampled points, consists of a pair $\mathcal{D} \equiv (\Sigma, \Phi)$, where

(1) Σ is a plane subdivision [Pre85], having the set of points $\{(x, y) \mid (x, y, z) \in \mathcal{S}\}$ as its vertices;

(2) Φ is a family of bivariate continuous functions, such that every function $\phi_i \in \Phi$ is defined on a closed region f_i of Σ, and interpolates the elevations of the data points whose projections are the vertices of f_i;

(3) for every pair of adjacent regions f_i and f_j in Σ, functions ϕ_i and ϕ_j set the same values on the edge shared by f_i and f_j.

Surface patches, defined over the regions of Σ, are called *faces* of the DSM. Condition (3) can be necessary to ensure the continuity of the surface.

Since a plane subdivision with n vertices is composed of $O(n)$ edges and regions, the spatial complexity of a DSM with n vertices is a linear function of n.

Polyhedral Surface Models (PSMs) are DSMs characterized by linear interpolating functions (and, thus, they have plane faces). PSMs used in practice have triangular faces, and are built based on a triangulations of the set of data points. For a polyhedral surface, the continuity is ensured by the interpolation condition in (2), and thus condition (3) is redundant.

Points forming set $\mathcal{S}$, on which a DSM or PSM is built, can be either distributed on a regular grid, or can be irregularly sampled. *Triangulated Irregular Networks* (TINs), which are triangulated PSMs based on irregularly distributed data, can better adapt to surface features, since they have the capability of including special points (minima, maxima, saddle points) and, through the notion of constrained and conforming triangulation, also special lines (ridges, valleys), which characterize the surface. Thus, TINs succeed in representing a surface at a certain level of accuracy using a smaller amount of data.

In many practical applications (for instance, in geographical applications), a large number of sampled values is available. Building a DSM based on the whole set of data points would be

too expensive in terms of storage. Thus, a subset of the original dataset is selected, according to some appropriate mechanism, and an approximate DSM is built based on them. An *approximate* digital surface model built on a data set $\mathcal{S}$ is a digital surface model built on a subset $\mathcal{S}'$ of $\mathcal{S}$. Different norms can be used to evaluate the approximation error of an approximate DSM. The approximation error at a point $P \equiv (x, y, z) \in \mathcal{S} - \mathcal{S}'$ is often evaluated as the infinite norm, i.e., the absolute value of the difference between the interpolated and the measured elevation values. The error on a face is defined as the maximum error of points in $\mathcal{S} - \mathcal{S}'$ whose vertical projections lie inside or on the boundary of the face. A DSM is said to approximate a surface *at level ε of accuracy* if, for every face, the approximation error is less or equal to ε.

Even if the points of the original dataset $\mathcal{S}$ were distributed on a regular grid, the selected subset $\mathcal{S}'$, on which an approximate DSM is built, can have an irregular distribution: often, data points are incrementally inserted into $\mathcal{S}'$ until the approximation error becomes less than a predefined tolerance value, thus adding several points in rough areas, and fewer in flat ones, to reflect the different complexity of the surface. Thus, TINs are used to deal with such irregularly distributed data.

Building a TIN reduces to the problem of computing a triangulation of the domain with vertices at the projections of the data points on the $x - y$ plane. Often, a Delaunay triangulation [Pre85] is used as domain subdivision for a TIN, because of its good behaviour in numerical interpolation (intuitively, a Delaunay triangulation avoids long and thin triangles). As shown in [Rip90], among all possible triangulations of a fixed data set, the Delaunay triangulation is the one that minimizes the roughness of the approximating surface.

3 Triangulations

This Section deals with triangulations, Delaunay triangulations, constrained and conforming Delaunay triangulations. A survey of related construction algorithms is also provided.

Given a finite set V of points in the plane, a *triangulation* of V is a maximal straight-line plane graph having V as its set of vertices. Thus, in a triangulation, every region, except for the external region, is a triangle.

The problem of finding an optimal triangulation of a given set of points has been considered for many different applications. In surface approximation problems, a criterion related to the size of the angles of triangles is used. When a triangulation is taken as the basis for a digital surface model, the approximated elevation of a point P, internal to a triangle, is obtained as a function of the elevations of the vertices of that triangle. A better approximation is obtained when the three vertices of the triangle lie as close as possible to P. Intuitively, a *Delaunay triangulation* of a set V of points, is, among all the possible triangulations of V, the one in which triangles are as much equiangular as possible (see Figure 1).

The Delaunay triangulation of a set V of points in the plane is usually defined in terms of another geometric structure, the *Voronoi diagram*, which describes the proximity relationship among the points of V. The Voronoi diagram of a set V of n points is a subdivision of the plane into n convex polygonal regions, called *Voronoi regions*, each associated with a point P_i of V. The Voronoi region of P_i is the set of points of the plane which lie closer to P_i than to any other point in V. Two points P_i and P_j of V are said to be *Voronoi neighbors* when the corresponding Voronoi regions are adjacent.

The *geometric dual graph* of the Voronoi diagram is a plane graph $T \equiv (V, E)$, called the

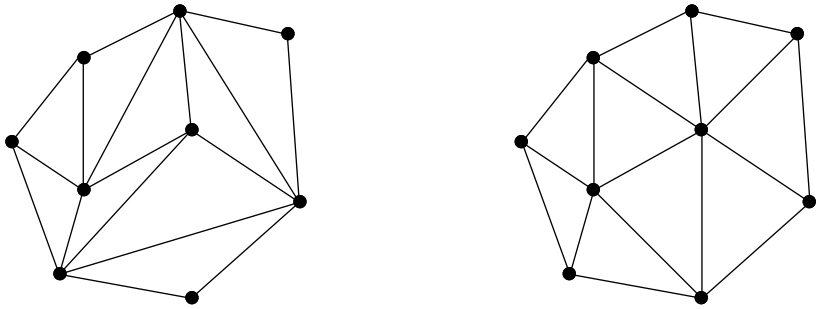


Figure 1: (a) An arbitrary triangulation of a point set, and (b) a Delaunay triangulation of the same set.

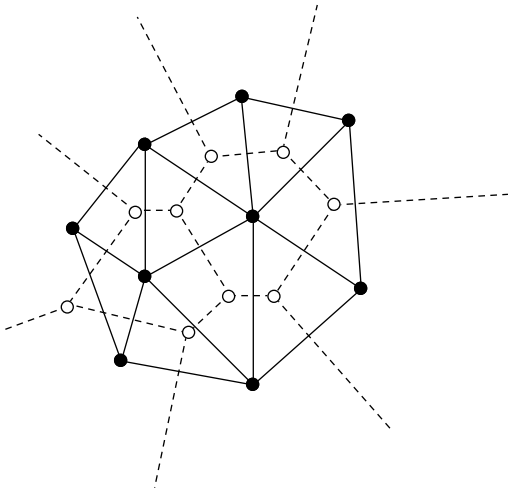
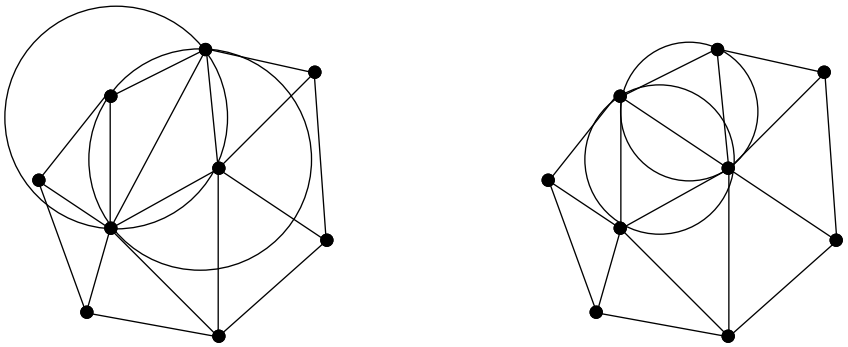
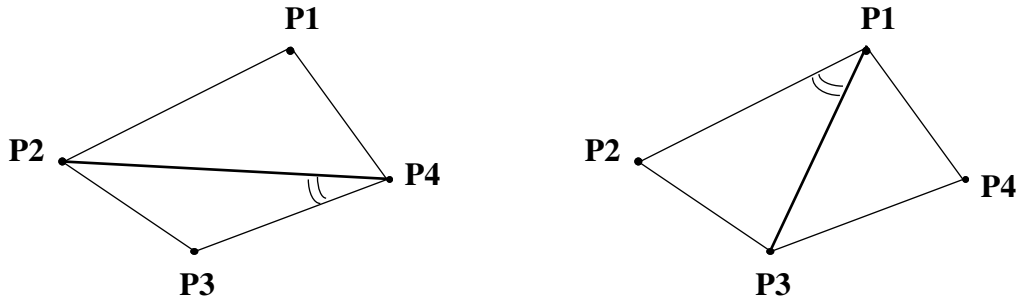


Figure 2: The Voronoi diagram of a point set (dashed lines), and its dual graph, the Delaunay triangulation (plain lines).



(a) (b)

Figure 3: The empty circle property: (a) is not a Delaunay triangulation, (b) is a Delaunay triangulation.



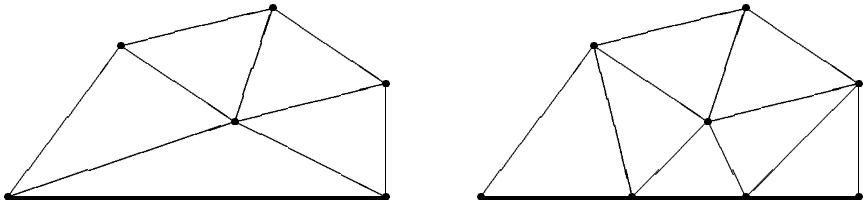
(a) (b)

Figure 4: The max-min angle property: the thick edge in (a) is not locally optimal, the one in (b) is locally optimal.

Delaunay graph of V, whose edges join pair of points P_i, P_j $(i \neq j)$ of V, such that P_i and P_j are Voronoi neighbors (see Figure 2). The Delaunay graph explicitly represents the Voronoi neighborhood relation induced by the Voronoi diagram over set V. If every face of T is a triangle, then T is the same as the Delaunay triangulation of V. Otherwise, T can always be completed to a Delaunay triangulation by decomposing each of its non-triangular faces. Since non-triangular faces can be decomposed in different ways, there are many possible Delaunay triangulations generated by the same Delaunay graph. The Delaunay triangulation of a set V is unique (i.e., the Delaunay graph is a triangulation) if and only if, for any four points of V, such points do not belong to the same circle.

An alternative characterization of the Delaunay triangulation is given by the so-called *empty circle property*. Let τ be a triangulation of a set V of points. A triangle t of τ is said to satisfy the *empty circle property* if and only if the circle circumscribing t does not contain any point of V in its interior. A triangulation τ of V is a Delaunay triangulation if and only if every triangle of τ satisfies the empty circle property (see Figure 3).

A Delaunay triangulation satisfies also the *max-min angle property*, which is used operatively by several construction algorithms. Let τ be a triangulation of V, let e be an edge of τ, and Q be the quadrilateral formed by the two triangles of τ adjacent to e. Edge e is said to satify the *max-min angle property* if and only if either Q is not strictly convex, or replacing e with the opposite diagonal of Q does not increase the minimum of the six internal angles of the resulting triangulation of Q. An edge e, which satisfies the max-min angle property, is also called a *locally*



(a) (b)

Figure 5: (a) A constrained Delaunay triangulation, and (b) a conforming one (there is only one constraint segment, which is drawn in thick lines).

optimal edge (see Figure 4). A triangulation τ of V is a Delaunay triangulation if and only if every edge of τ is locally optimal.

The equivalence of the max-min angle property and the empty circle property has been shown by Lawson [Law77] by considering the triangulation of a strictly convex quadrilateral. It is important to point out that a locally optimal edge does not necessarily belong to a Delaunay triangulation. If the repeated application of the max-min angle criterion to the edges of a triangulation τ does not cause any diagonal flipping, then τ is a Delaunay triangulation.

Another characterization of the Delaunay triangulation of a set V of points in d dimensions is provided by its relation with the convex hull of a transformed set V' of points in $d+1$ dimensions. When $d = 2$, then V' is obtained from V by trasforming every point $P_i \equiv (x_i, y_i) \in V$ into the point $P' \equiv (x_i, y_i, x_i^2 + y_i^2)$ on the paraboloid $z = x^2 + y^2$. It can be shown that the Delaunay graph of V is the vertical projection of the lower portion of the convex hull of V'.

3.1 Constrained and Conforming Delaunay Triangulation

In practice, a triangulation must sometimes be forced to include a given set of segments among its edges. The need for including a constraining set of segments in a triangulation arises, for instance, when reconstructing an object from a set of three-dimensional stereo segments lying on the object boundary and obtained from several viewpoints [Boi90]. In terrain modeling, the possibility of including some a-priori knowns lineal features of the terrain, such as ridges, rivers, country boundaries [Peu75], has a fundamental importance in order to obtain a representation that adapts to the natural characteristics of the surface. Moreover, it is important to combine the possibility of including special lines with the advantages of a Delaunay triangulation (i.e., a good behaviour in numerical approximations).

The problem of including segments in a Delaunay triangulation has been faced in the literature by two different perspectives, leading to the definitions of *constrained* and *conforming Delaunay triangulation*. The constrained Delaunay triangulation is the best approximation of the Delaunay triangulation containing the set of given segments among its edges [Lee86]. The conforming Delaunay triangulation is a proper Delaunay triangulation of an augmented set of vertices, such that each constraint segment is the union of edges of the triangulation [Ede93] (see Figure 5).

Given a set V of points in the plane and a set L of straight-line segments, having their endpoints in V, and such that they do not intersect each other except at their endpoints, the pair $G \equiv (V, E)$ defines a *constraint graph* (the segments in L are called *constraints*). A *triangulation of V constrained by L* is a triangulation of V containing the constraint graph as a subgraph.

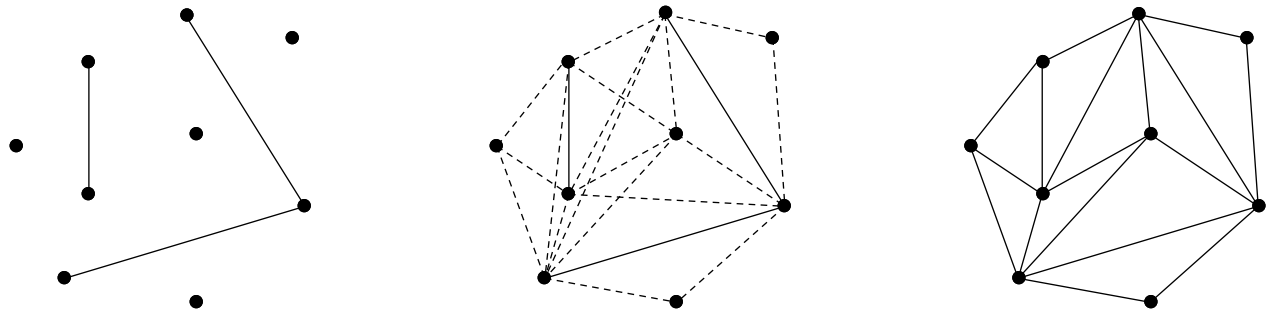


Figure 6: (a) A constraint graph; (b) the corresponding visibility graph, and (c) a constrained triangulation.

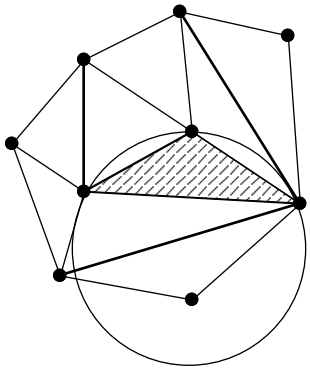


Figure 7: A constrained Delaunay triangulation (constraint segments are drawn in thick lines). Note that the dashed triangle satisfies the constrained version of the empty circle criterion.

An alternative characterization of constrained triangulation can be given through the notion of visibility. Two points P_i and P_j in V are called mutually *visible* with respect to L if and only if they can be joined by a straight-line segment without intersecting the interior of any constraint segment in L. We call *visibility graph* the graph $G_v \equiv (V, E_v)$, where edges in E_v join pairs of mutually visible points in V. Note that the constraint graph is a subgraph of the visibility graph. A *triangulation of V constrained by L* is a maximal straight-line plane subgraph of the visibility graph, containing the constraint graph as a subgraph (see Figure 6).

A *Constrained Delaunay Triangulation* (CDT) of a point set V with respect to a set L of constraints is a constrained triangulation τ of V with respect to L, satisying the following constrained vertion of the *empty circle property*. The circumcircle of each triangle of τ does not contain in its interior any point of V which is visible from all the three vertices of the triangle (see Figure 7). Note that, if the set L of constraints is empty, then we have a standard Delaunay triangulation.

The properties of a standard Delaunay triangulation can be extended to a CDT with some modifications due to the visibility constraints defined by the segments in L. The *max-min angle property* can be reformulated for a CDT by simply considering only those convex quadrilaterals formed by two adjacent triangles which do not share a constraint segment.

One problem with using a constrained Delaunay triangulation in surface approximation is that the forced inclusion of constraint edges may produce thin triangles. An alternative approach is provided by the notion of conforming Delaunay triangulation. Intuitively, the idea is splitting the constraint segments into shorter segments by adding new vertices (the elevations of such new points are inferred by linear intepolation from the ones of the two constraint endpoints) in such a

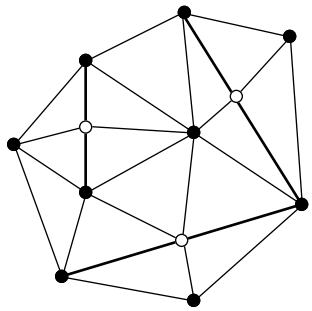


Figure 8: A conforming Delaunay triangulation.

way that the CDT of the resulting set of short segments will be a Delaunay triangulation of the augmented vertex set (see Figure 5 again).

Let $G \equiv (V, E)$ be a constraint graph, and let $W \supseteq V$ be a set of points. A triangulation τ of W *conforms* to G if every constraint in E is the union of edges of τ.

A *Conforming Delaunay triangulation* is a Delaunay triangulation of W which conforms to G (see Figure 8). Let us call the closed portion of an edge of G between two contiguous points of W on this edge an *interval*. A Delaunay triangulation τ of W conforms to G iff every interval defined by G and W satisfies the empty circle property with respect to W. For every constraint graph $G \equiv (V, E)$, there exists a set $W \supseteq V$ such that the Delaunay triangulation of W conforms to G. In particular, it has been proven [Saa91] that there alway exists a set W, whose Delaunay triangulation conforms to G, such that all the extra vertices in $W - V$ lie on the edges of G (most existing approaches to the construction of a conforming Delaunay triangulation are based on this latter result).

3.2 Algorithms for Computing a Delaunay Triangulation

Existing algorithms for building a Delaunay triangulation or, equivalently, its dual graph (the Voronoi diagram) can be classified into the following five categories:

- *two-step algorithms*, which first compute an arbitrary triangulation, and then optimize it to a Delaunay triangulation by iteratively applying either the empty circle or the max-min angle criteria.
- *incremental algorithms* [Gui85, Wat81, Knu92], which construct a Delaunay triangulation by stepwise insertion of the data points, while maintaining a Delaunay triangulation at each step.
- *divide-and-conquer algorithms* [Lee80], which compute a Delaunay triangulation by recursively splitting the point set into two halves, and merging the computed partial solutions.
- *sweep-line methods* [For87], which compute the Voronoi diagram of a set of points by first transforming it in such a way that the Voronoi region of a point P_i is considered only when P_i is intersected by the sweep-line.
- *Three-dimensional algorithms*, which compute the convex hull in 3D, and then project the lower portion on the $x - y$ plane.

Three-dimensional algorithms will not be considered here. See [Ede87] for a treatment of the topic.

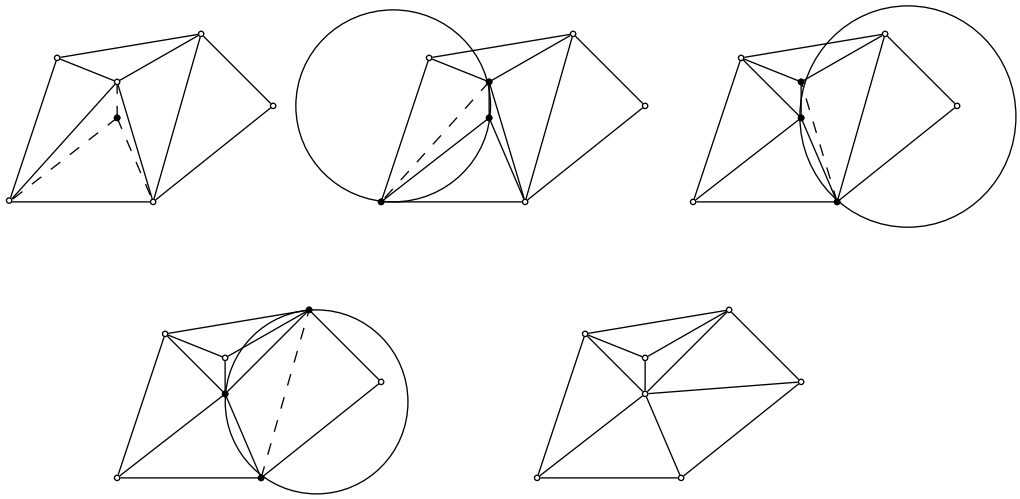


Figure 9: A working example of optimization process in Guibas and Stolfi's algorithm.

The first Delaunay triangulation algorithms were based on a *two-step strategy*. An arbitrary triangulation of the given set V of points can be obtained through the following three steps:

- sort the points of V by increasing x-coordinate;
- form a triangle with the first three non-collinear points in the sorted sequence;
- iteratively add the next point P_i by connecting P_i to all the vertices of the existing triangulation which are visible from P_i (i.e., they can be connected to P_i without intersecting existing edges).

The optimization step iteratively applies the max-min angle (or the empty circle) criterion to any internal edge of the current triangulation, such that its two adjacent triangles form a strictly convex quadrilateral, until no more edge swapping occurs. Constructing the initial arbitrary triangulation requires $O(n \log n)$ operations in the worst case. Lawson [Law77] shows that the optimization process terminates and has a $O(n^2)$ worst-case time complexity.

Incremenlal algorithms can be further classified into *static* and *on-line* algoritms. *Static algorithms* usually start by sorting all the points according to their euclidean distance from a fixed origin and then build the triangulation in such a way that each created triangle belongs to the final tesselation [Law77, McL76].

On-line algorithms are based on the incremental insertion of the internal points in an initial Delaunay triangulation of the domain. The initial triangulation of the domain can be obtained, for instance, by creating a triangle enclosing all the data points, which will be removed together with all the edges incident in its vertices at the end of the process.

The update of the current Delaunay triangulation at the insertion of a new internal point P_i can be performed in two different ways. One approach [Gui85] builds first an arbitrary triangulation by connecting P_i to the three vertices of the triangle t of the existing triangulation, which contains P_i. The triangulation is then optimized by iteratively applying the max-min angle criterion until no more edge swapping occurs (see Figure 9).

Another approach [Wat81], locates first the triangle t of the current triangulation which contains point P_i. Starting from t, it deletes all the triangles of the current triangulation, whose

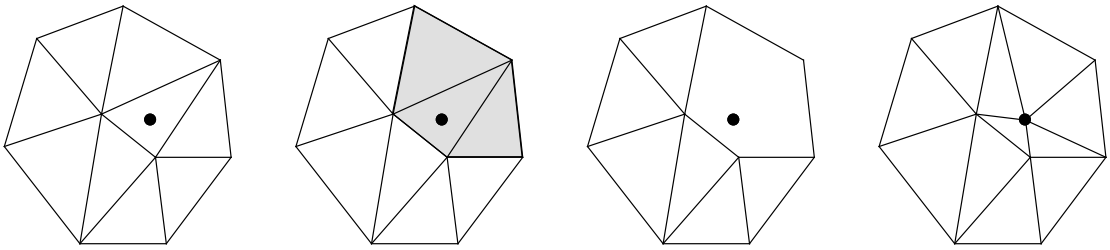


Figure 10: Illustration of the steps of Watson's algoritm.

circumcircle contains P_i. Such triangles form a star-shaped polygon, called the *influence polygon* of P_i, containing P_i in its kernel. The new Delaunay triangulation is simply obtained by connecting P_i with all the vertices of the influence polygon (see Figure 10).

In both algorithms, the update of the triangulation when a new point is inserted requires $O(n)$ steps in the worst case, and thus these algorithm have an $O(n^2)$ time complexity. On the contrary, the insertion of a new point requires only a constant number of steps in the average case [Sib78].

In summary, both two-steps and incremental techniques are sub-optimal, since they have an $O(n^2)$ time complexity in the worst case, but they have a good expected time behaviour: a randomized version of Guibas and Stolfi's algorithm runs in expected $O(n \log n)$ time [Knu92]. On-line algorithms are especially appropriate when not all the data points are known in advance. Such methods are used for the construction of models approximating a surface at a certain level of accuracy, though repeated insertion of data points until the required precision degree is met. They are also used for building multiresolution models, which are collections of approximate models of a surface based on increasingly larger sets of data (see Section 4).

Compared with incremental techniques, *divide-and-conquer algorithms* are more complicated and require more storage space, but are in general computationally more efficient. An asyntotically optimal Delaunay triangulation algorithm, working in $O(n \log n)$ time, is due to Lee and Schacter [Lee80].

Similar to the Voronoi diagram computation algorithm proposed by Shamos [Pre85], the method of Lee and Schacter is based on a recursive splitting of the set of data points into two almost equally sized subsets, and on the pairwise merging of the Delaunay triangulations separately computed. The algorithm performs the following four steps:

- the points of V are preliminarily sorted from left to right (if two points have the same x-coordinate, then the y-coordinate is considered);
- set V is split into two subsets V_L and V_R, where V_L contains the leftmost half of the points of V, and V_R, the rightmost half;
- the Delaunay triangulations of V_L and V_R are recursively constructed and then merged together to form the Delaunay triangulation of V.

The merging step of the triangulations of V_L and V_R starts with the computation of the convex hull of $V = V_L \cup V_R$, which is the domain of the Delaunay triangulation of V. This reduces to determining the lower and upper common tangent of the convex hulls of V_L and V_R (domains of the corresponding triangulations). Then, we move from the lower tangent to the upper one, by deleting the edges that are not in the final Delaunay triangulation of V, and adding the new edges (see Figure 11). It can be shown that the common tangents can be found in $O(n)$ time, and the

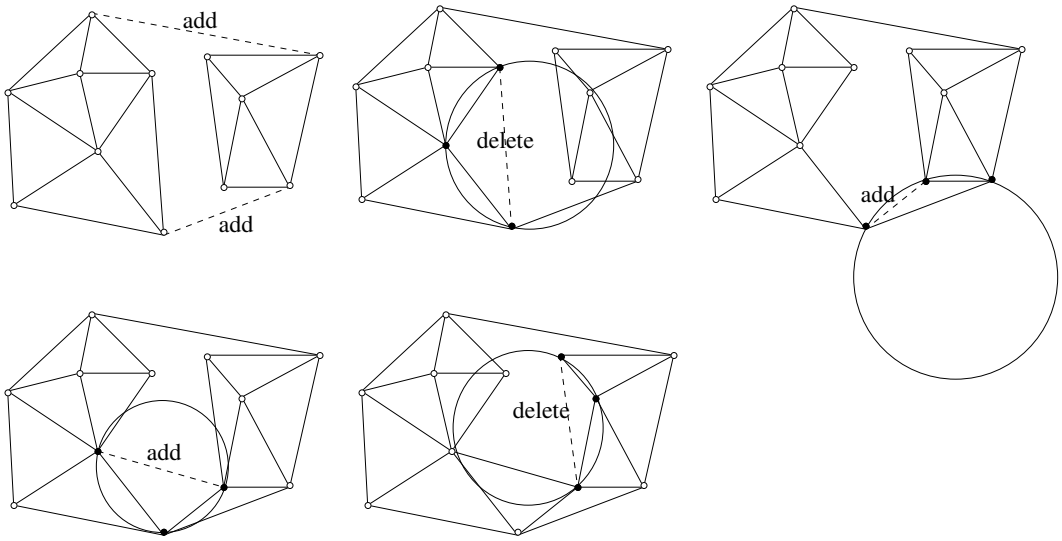


Figure 11: Merge step in the divide-and-conquer algorithm.

whole merging phase has a linear time complexity as well. By inserting a preliminar sorting of the points of V from left to right, the split of V into V_L and V_R requires only a linear time at each iteration. Hence, the worst-case time complexity of this algorithm is globally equal to $O(n \log n)$ time, which is worst-case optimal.

The *sweep-line algorithm* proposed by Fortune [For87] for Voronoi diagram computation is competitive in simplicity with incremental algorithms. Since it avoids the merging step, it is simpler than divide-and-conquer ones. The sweep-line technique conceptually sweeps a horizontal line across the plane, noting the regions intersected by the line as the line moves. Computing the Voronoi diagram directly with a sweep-line technique is difficult, because the Voronoi region of a point may be intersected by the sweep-line before the point itself is intersected. Thus, the algorithm computes a geometric transformation of the Voronoi diagram which has the property that the lowest point of the transformed region of a point appears at the point itself, and, thus, the Voronoi region of a point is considered only when the point itself is intersected by the sweep line.

3.3 Algorithms for Building a Constrained Delaunay Triangulation

The problem of building a constrained triangulation has been first considered for the special case of an arbitrary triangulation of a simple polygon, in connection with applications to polygon decomposition. Since this case is not interesting for applications to surface modeling, here we focus on algorithms for the general case of arbitrary constraints.

Algorithms for solving the general constrained triangulation problem can be classified into two major categories:

- algorithms, that we call *direct algorithms*, which compute the CDT by starting from the whole set of points and constraint segments [Lee86, Boi90, Che89];
- algorithms, that we call *difference algorithms*, which modify a standard Delaunay triangulation (or a Voronoi diagram) of the set of data points by insertion of the constraint segments [Sch87].

The algorithm by Lee and Lin [Lee86] is based on the preliminary computation of the visibility graph, and on the successive iterative elimination of those edges which are not in the CDT. In a CDT, an edge $\overline{P_iP_j}$ is a Delaunay edge if and only if P_i and P_j are mutually visible and there exists a circle passing through P_i and P_j which does not contain any vertex visible from both P_i and P_j. For every vertex P_i of the visibility graph, the algorithm scans the list of vertices visible from P_i in counterclockwise order. At the beginning, the vertex P_j in such list, corresponding to the shortest edge around P_i, is found. Because of the previous result, $\overline{P_iP_j}$ must be a Delaunay edge. For every three consecutive vertices P_1, P_2, P_3 around P_i, the quadrilateral formed by P_i and P_1, P_2, P_3 is considered to decide whether P_iP_2 is a Delaunay edge. The visibility graph can be computed in $O(n^2)$ steps (where n is the number of data points in V), by applying the method described in [Asa86], whereas the detection of Delaunay edges requires $O(n^2)$ steps in the worst case.

The application of the divide-and-conquer paradigm to the design of an efficient algorithm for constrained Delaunay triangulation seems to be difficult because of the problem related to the linear separability of the set of constraint segments. The algorithms by Chew [Che89] and by Joe and Wang [Joe89] extend the divide-and-conquer algorithm of Lee and Schacter for computing a standard Delaunay triangulation. The worst-case time complexity is $O(n \log n)$ for both algorithms. In [Che89], the constraint graph $G \equiv (V, L)$ is assumed to be contained into a rectangle, which is subdivided into vertical strips in such a way that there is exactly one vertex in each strip. According to the divide-and-conquer strategy, the CDT is computed for each strip and adjacent strips are merged together to form new strips containing twice as many vertices. The recursive partition of the problem can be represented by a tree of depth $O(\log n)$. The main difficulty in this process is due to cross edges, i.e., edges of the constraint graph that cross a strip without having an endpoint in it. It can be easily seen that $O(l)$ constraint segments (where l is the cardinality of L) can cross a strip, giving a total of $O(n^2)$ cross edges for all the strips. Chew's algorithm only stores those cross edges which are visible from at least one vertex in the strip, which restricts the total number of stored edges to $O(l)$. Thus, the complexity of all the subproblems at the same level of the tree is $O(n+l)$, where n is the number of points in V and l the number of constraints in L. Chew shows that two adjacent strips can be merged in linear time. Thus, the algorithm builds the resulting CDT in $O(n \log n)$ time.

An on-line incremental algorithm, based on the modification of an existing CDT by iteratively inserting new points and constraint segments has been proposed by De Floriani and Puppo [DeF92a]. This algorithm is an extension of Watson's algorithm for computing a standard Delaunay triangulation, with the difference that, at each step, either a new point or a new constraint segment can be inserted.

The insertion of a new point P_i is performed as in Watson's algorithm, with the difference that now the "constrained" version of the empty circle criterion is adopted to build the influence polygon Q_P of P. A new constraint segment $s \equiv \overline{P_1P_2}$ can be inserted only when the two endpoints P_1 and P_2 of s are already vertices of the current triangulation. Segment insertion also involves the determination and retriangulation of an influence polygon. The *influence polygon* Q_s of s is formed by the union of all the triangles of the old triangulation which are intersected by s. Q_s is a simple polygon, and has segment s as a diagonal. It has been shown that the final updated CDT can be obtained by removing all the triangle edges which lie inside the influence polygon Q_s, splitting Q_s into two subpolygons, separated by diagonal s, and computing a Delaunay triangulation inside the two polygons separately (see Figure 12). Intersecting segment s with the existing triangulation can be done in time linear in the number of intersected triangles, i.e., $O(n)$ in the worst case. The CDT of a polygon can be computed in $O(k \log k)$ time, where k is the size of the polygon.

Since the algorithm is based on the incremental modification of a CDT by the insertion of a

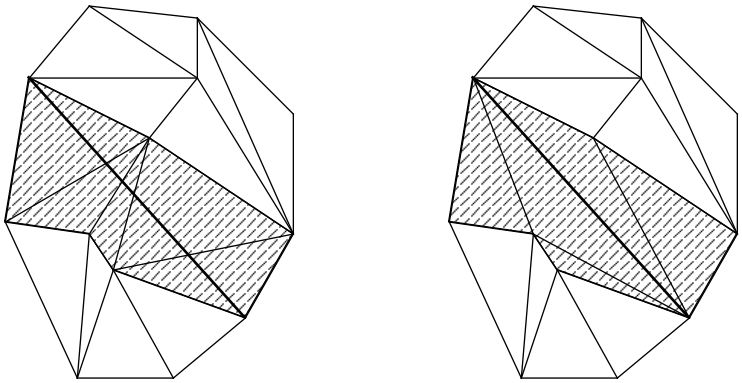


Figure 12: Insertion of a constraint segment s: (a) the influence polygon of s, (b) the updated CDT obtained by re-triangulating the polygon.

single point or segment at a time, it necessarily exhibits a higher worst-case time complexity than non-incremental ones. On the other hand, the approach is particulary interesting since it makes possible to add data points and constraint segments at run time. The insertion of a point has an $O(n)$ complexity, while inserting a segment may require $O(n \log n)$ operations.

In [Kao91], algorithms are described for incremental construction and dynamic maintenance of a constrained Delaunay triangulation, which perform point insertion, point deletion, segment insertion and segment deletion. In particular, randomized algorithms are proposed which compute a CDT of n points and l constrainf segments in $O(n \log n \log l)$ expected time and perform l segment deletions in $O(n \log l)$ expected time.

3.4 Algorithms for Building a Conforming Delaunay Triangulation

Constructing a conforming Delaunay triangulation with respect to a conflict graph $G \equiv (V, E)$ basically reduces to the problem of finding a set V' of points such that the Delaunay triangulation of $W \equiv V \cup V'$ conforms to G. This is generally a much harder problem than computing a constrained triangulation of G, since a large number of extra points may be necessary in order to achieve conformity. The lower bound for the size of V' is equal to $\Omega(nl)$, where n and l are, respectively, the number of vertices and of constraints in G. An $O(l^2 n)$ upper bound for the number of points to be added has been proven by Edelsbrunner and Tan [Ede93].

Proposed approaches [Boi90, Nac91, Olo91, Saa91] are based on the idea of placing sufficiently many points on the edges of the constraint graph in such a way that the Delaunay triangulation of the augmented point set is guaranteed conform to G. The problem is that there is generally no function $f(n)$ that can bound the number of inserted points. In particular, the number of points added grows as the constraint segments move closer to each other.

The algorithm by Boissonnat et al. [Boi90], is based on the remark that a subsegment e of a constraint segment is a Delaunay edge if the circle having e as diameter does not intersect any other constraint segment. Thus, if the circle associated with a constraint segment e intersects some other segment, then e is split into a finite number of subsegments such that none of their circles intersects any constraint. When two constraint segments intersect at an endpoint, one new point is inserted on both segments, in such a way that the circumcircle of the triangle defined by the common endpoint and by the two new points does not intersect any constraint segment.

The method proposed by Saalfeld [Saa91] is based on an iterative process, which, at each step computes an ordinary Delaunay triangulation of the current vertex set (initially, set V): if the triangulation does not conform to G, then the intersection points between the edges of the triangulation and the constraint segments are added to the set and the process is iterated.

It should be remarked that the approaches illustrated above present a time complexity which is not necessarily polynomial, since there is no upper bound to the number of inserted points.

The algorithm by Edelsbrunner and Tan [Ede93] has an $O(l^2n+n^2)$ worst-case complexity, and is optimal with respect to the number of extra points it inserts (at most $O(l^2n)$). This algorithm is based on a different approach, which inserts new points lying not necessarily on constraint segments. The algorithm consists of two phases. The first phase constructs $O(n)$ circles, with disjoint interiors (they are allowed to be tangent) such that their union is connected and contains V. The intersections between these circles and the edges of G are added, together with the possible tangent points between pairs of circles. At the end of this phase, every constraint segment has been split into *protected* intervals (i.e., intervals contained into a circle), and *unprotected* intervals). Unprotected intervals will be further subdivided during the second phase.

4 Multiresolution Surface Models

Organizing surface representations at different levels of detail is a recent research issue. *Multiresolution surface models* provide a data compression mechanism as well as a variable resolution method for representing a surface different levels of abstraction. A multilevel organization allows an easy implementation of searching and other geometric operations, such as finding surface intersection, or zooming when visualizing the surface. Moreover, it makes real-time simulation and visualization possible for those applications in which describing less important areas with fewer details is a relevant issue.

Multiresolution models consist of collections of DSMs built based on increasingly large subsets of the given data set S. The structure in which such individual DSMs are connected induces a classification of multiresolution surface models into *hierarchical* and *pyramidal* models (see [DeF94] for a survey).

4.1 Hierarchical Models

The concept of hierarchical surface model is based on that of hierarchical subdivision of the domain. Intuitively, given a subdivision Σ, a region f of Σ can be seen as an individual entity and refined into a subdivision Σ_f, whose domain covers f. The refinement of f is performed by adding new vertices either inside f or on its sides. Recursive application of the refinement process leads to a hierarchy of subdivisions.

More formally, a hierarchical subdivision is defined based on an ordered collection $\{\Sigma_0, \ldots, \Sigma_m\}$ of subdivisions such that, for every $j > 0$, Σ_j has more than one region, and there exists exactly one $i < j$ such that the domain of Σ_j is a region of Σ_i. A *hierarchical subdivision* $\mathcal{H}$ is a tree with labelled arcs, where the nodes are the subdivisions $\Sigma_0, \ldots, \Sigma_m$. Σ_0 is the root. Every subdivision Σ_j is linked as a child to the unique subdivision Σ_i (with $j < i$) containing the region f which is the domain of Σ_j. Arc (Σ_i, Σ_j) is labeled with region f. An example of a hierarchical subdivision is shown in Figure 13.

A region which is refined in the hierarchy (i.e., which appears as the label of some edge)

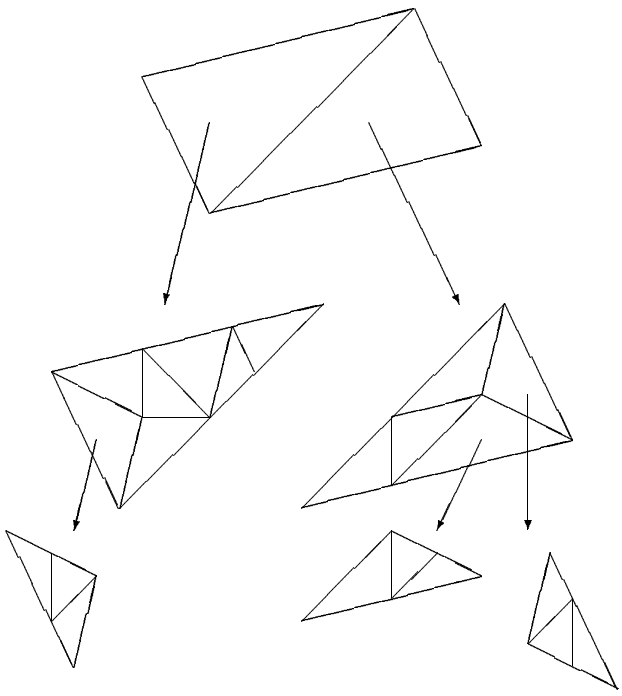


Figure 13: A hierarchical subdivision.

is called a *macroregion*. A region is called *simple* otherwise. The total number of regions in a hierarchical subdivision is linear in the number of simple regions, which, in turn, linearly depends on the total number of inserted data points [DeF92].

A *Hierarchical Surface Model* (HSM) is built on a hierarchical subdivision by associating with each subdivision Σ_i in the hierarchy a DSM $\mathcal{D}_i \equiv (\Sigma_i, \Phi_i)$. Any vertex $P_i \equiv (x_i, y_i)$ which is common to more than one subdivision Σ_j has the same elevation value z_i in each corresponding surface model $\mathcal{D}_j$.

In a hierarchical subdivision $\mathcal{H}$, the refinement of a macroregion can split one of its edges into a chain of edges through insertion of new vertices. Inconsistent refinement of an internal edge e may have undesirable effects on the surface model supported by $\mathcal{H}$, because the continuity of the surfaces on adjacent patches could not be guaranteed. In order to avoid vertical discontinuities in the overlying surface, a special *matching rule* must be satisfied in $\mathcal{H}$. This rule requires that, for any two adjacent regions f_1 and f_2 in a subdivision Σ_i in the hierarchy, the same set of vertices is inserted on the common edge of f_1 and f_2 in the subdivisions refining f_1 and f_2 at the next level.

According to the refinement criterion, existing HSMs (see Figure 14) can be classified into *quadtree-based models* and *hierarchical triangulated models*. Quadtree-based models, such as *quadtrees* [Che86, Sam92, Von87] and *quaternary triangulations* [Bar84, Gom79], are HSMs built on gridded data, in which the refinement process follows a regular geometrical pattern, with rectangles and equilateral triangles, respectively, as basic elements.

Triangle-based HSMs, built on irregularily distributed data, provide a more flexible surface description, since they can better adapt to the roughness of the surface and include surface-specific points and lines, which characterize the surface independently from the data sampling. Such models are based on a non-geometrically fixed hierarchical decomposition, in which faces are arbitrary triangles. *Ternary triangulations* [DeF92, Pon87] represent a first attempt in this

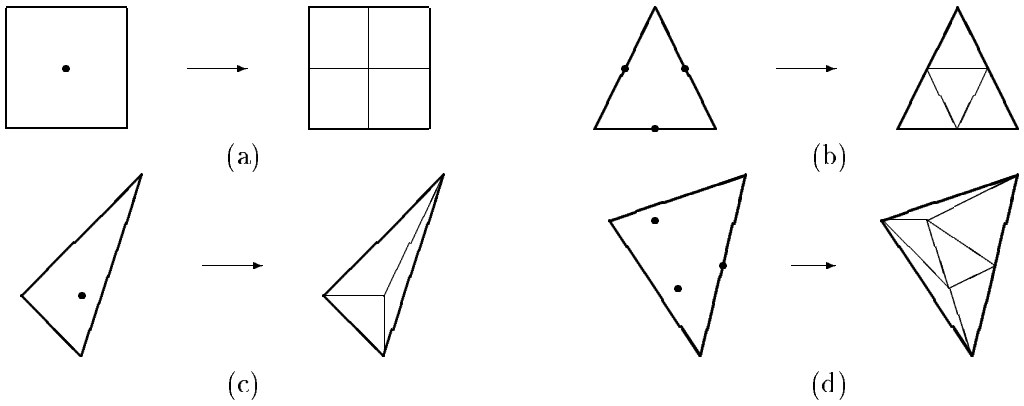


Figure 14: Different refinement rules in existing HSMs: (a) quadtree, (b) quaternary triangulation, (c) ternary triangulation, (d) hierarchical TIN.

direction, using a decomposition pattern with fixed topology and variable geometry, but they have the disadvantage of producing long and thin triangles, and thus a less efficient surface approximation. *Hierarchical Triangulated Irregular Networks* (HTINs) are hierarchical models based on a hierarchy of triangulations, in which the recursive refinement process is driven by an accuracy-based criterion. Given a sequence $\varepsilon_0, \ldots, \varepsilon_m$ of decreasing tolerance values, each level in the hierarchy represents the surface at one of the predefined values ε_i. The top level corresponds to the coarsest description, satisfying the largest tolerance ε_0. Passing from a level i to level $i+1$, every triangle t is expanded in a finer triangulation by iteratively inserting points in its interior or on its sides, until the accuracy ε_{i+1} is reached. The number of points in the refinement of a triangle is not predefined.

The rule used for selecting and inserting points in a triangle characterizes the different types of HTINs (in all cases, the criterion is based on an error evaluation, and ensures the matching rule to be satisfied). In *Adaptive Hierarchical Triangulations* (AHTs) [Sca90], a triangle t is refined by iteratively applying a *splitting rule*. Four maximun-error points, related to the interior and to the three edges of t, are computed, and, among them, the ones corresponding to an error greater than ε_{i+1} are selected for insertion. A predefined decomposition pattern is then applied, depending on the configuration of selected points. This process is repeated until no more points are selected for insertion.

In *Hierarchical Delaunay Triangulations* (HDTs) [DeF92, DeF93], are based on a hierarchy of where each element is a Delaunay triangulation. Notice that, while the subdivision inside every macrotriangle is locally a Delaunay triangulation, the global expanded subdivision of the whole domain generally is not. The refinement of a triangle t is performed by an iterative application of the *Delaunay selector*, which, at each step, updates the current Delaunay triangulation by inserting the point having the maximum error. The basis of the construction algorithm for an HDT must be an on-line method which incrementally builds a Delaunay triangulation through iterative insertion of points, as those described in Section 3.2.

4.2 Pyramidal Models

A strictly hierarchical structure cannot be imposed on a Delaunay triangulation, since the insertion of a new point might cause a modification that can involve the whole tesselation. Even in HDTs, any description of the surface, corresponding to an intermediate level in the tree describing the hierarchical model, does not correspond to a Delaunay triangulation of the domain. Pyramidal

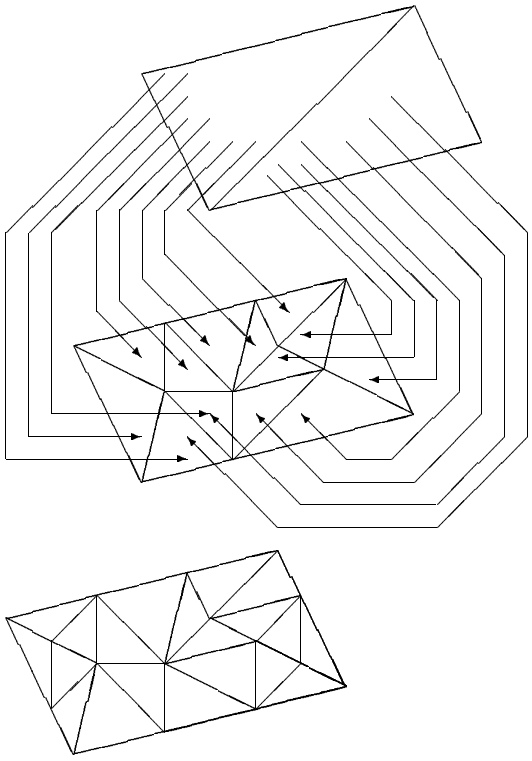


Figure 15: A pyramidal subdivision (only links between the first two levels are shown).

surface models have been developed as alternative multiresolution methods to hierarchical ones; such models allow global properties (such the Delaunayhood of a triangulation) to be satisfied with respect to the whole domain. Again, the concept of pyramidal surface model relies on that of pyramidal subdivision.

A pyramidal subdivision is defined on the basis of a sequence $\{\Sigma_0, ..., \Sigma_m\}$ of plane subdivisions of the domain D, each representing an increasingly finer description of D. Two regions belonging to different subdivisions may overlap. The structure of a pyramidal subdivision is stratified rather than hierarchical. A *pyramidal subdivision* $\mathcal{P}$ is represented by a labelled multigraph (i.e., a graph with parallel arcs), having $\{\Sigma_0, ..., \Sigma_m\}$ as its set of nodes. There is an arc (Σ_i, Σ_{i+1}), joining two consecutive subdivisions in $\mathcal{P}$, for every pair of faces $f_i \in \Sigma_i$ and $f_j \in \Sigma_j$, such that f_i and f_j have a non-empty intersections, and $f_i \neq f_j$. Such arc is labeled with the pair (f_i, f_j). An example of a pyramidal subdivision is shown in Figure 15. Several arcs (Σ_i, Σ_{i+1}), labeled with different pairs of faces (f_k, f_l), can join the same two subdivisions Σ_i and Σ_{i+1}. In the worst case, every region at level i can be linked to every region of the subdivision at level $i+1$, thus leading to an $O(n^2)$ worst-case space complexity of the structure (where n denotes the number of vertices in the most refined level of the pyramid).

The *Delaunay Pyramid* [DeF89] is a multiresolution surface model composed of a sequence of Delaunay triangulations, which represent the surface at increasingly finer levels of detail over the whole domain. Every triangulation is obtained from the previous one by iteratively inserting the data point corresponding to the maximum error, until the current Delaunay triangulation satisfies the required accuracy. Unlike a hierarchical Delaunay triangulation, the Delaunay pyramid guarantees the equiangularity property to be globally satisfied. This advantage is paid in terms of an increased space complexity. Moreover, a pyramidal structure does not allow easy local refinements in areas of interest.

In many applications, the different surface approximations are defined by increasingly larger sets of both points and segments (corresponding to lineal features which must be included in the representation). The definition of *constrained Delaunay pyramid* is analogous to the one of standard Delaunay pyramid, with the difference that every level in the structure consists of a constrained Delaunay triangulation. The construction of a constrained Delaunay pyramid is based on the on-line method for incrementally computing a CDT, described in Section 3.3.

5 Surfaces in Three Dimensions

The problem of reconstructing the bounding surface of a three-dimensional object can be formally stated as follows: given a set of entities in space, known to lie on the boundary of the object, find a polyhedral surface that contains all input data, thus approximating the object boundary.

Input data may consist either of a set of points, of a set of straight line segments, or of a set of polygonal cross sections lying on parallel planes intersecting the object. The format of the input depends on the type of data source. If it is not known how data have been obtained or if a single reconstruction method is to be used for data from various source types, then no structural relation between the input points can be assumed, except that all of them lie on the boundary of an object. Input data consisting of straight-line segments lying on the boundary of an object are typically acquired by means of a stereo process, and mainly occur in robotics applications. Surface reconstruction from a set of parallel cross sections is an important problem in clinical medicine, anatomic research and computer graphics.

The simplest boundary approximation is the one consisting of planar triangles, thus producing a three-dimensional simple closed polyhedron with triangular faces, i.e., a triangulation of the surface.

In this Section, we briefly review some of the existing methods proposed in the literature for triangulating the boundary of $3D$ objects. The interest reader is referred to the various papers mentioned here for a more throughout treatment of this problem. The different approaches can be classified, according to the specific dataset they are suited for, into:

- methods suitable for a set of irregularily distributed points,
- methods suitable for a set of straight-line segments, and
- methods dealing with a set of polygonal contours.

Variable resolution representations of $3D$ surfaces have also been developed, mainly for applications in computer vision and robotics: in [Pon87], a hierarchical model, the *prismtree*, for describing the boundary of a $3D$ object is described together with algorithms for surface intersection and neighbour finding based on such representation.

5.1 Reconstruction from Scattered Points

The reconstruction problem from scattered data points can be stated as follows: given a set V of vertices in $3D$, find a simple closed polyhedron passing through all vertices of V.

The above problem can be addressed in two ways:

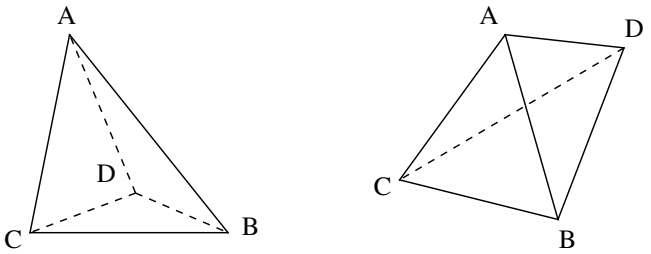


Figure 16: Tetrahedron $ABCD$ (left) can be eliminated if triangle ABC is on the external boundary and vertex D is in the interior; Tetrahedron $ABCD$ (right) can be eliminated if triangles ABC and ABD are on the external boundary and edge CD is in the interior.

- through the *direct* construction of a triangulation of the surface defined by the data [ORo81, Boi82].
- through the intermediate construction of some 3D structure, obtained by filling the interior of the object with tetrahedra, and, then, deriving a triangulated representation of the boundary of the object from such auxiliary structure [Boi84, Vel93].

Direct methods have been proposed in [ORo81, Boi82]. O'Rourke [ORo81] suggests the use of minimal surface area polyhedra: the convex hull is used as a starting point of the reconstruction algorithm, which modifies it systematically in order to include all data points internal to the hull in the boundary . In [Boi82], Boissonnat reduces the problem of building a triangular-faced polyhedron to that of finding several convex hulls on the Gaussian sphere. The important property of such a spherical surface representation is that the description of sufficiently small regions with no change in the sign of the curvature is one-to-one, thus ensuring that a triangulation on the unit sphere corresponds to a triangulation on the surface.

Undirect approaches have been proposed by Boissonnat [Boi84] and Veltkamp [Vel93], based on two different three-dimensional structures. The approach proposed by Boissonnat [Boi84] uses a tetrahedralization of the input data as an auxiliary structure, and extracts from it a representation of the object boundary as the collection of external faces.

A Delaunay tetrahedralization generealizes the concept of Delaunay triangulation, introduced in Section 3, to three dimensions. The *Voronoi diagram* of a set $V = \{P_1, ..., P_n\}$ of n points in the 3D space is defined as a collection of n convex polyhedra, each associated with a point in V, whose union covers $\mathbb{R}^3$. The polyhedron associated to P_i is the locus of points of the space which lie closer to P_i than to any other point in V, and is called the *Voronoi polyedron* of P_i. The geometrical dual of the Voronoi diagram, obtained by linking pairs of points of V whose Voronoi polyhedra are face-adjacent, is called the *Delaunay graph* associated with V. If no five points are cospherical, the $3D$ cells of the Delaunay graph are tetrahedra; otherwise, they can still be decomposed into tetrahedra. The resulting tetrahedralization is called a *Delaunay tetrahedralization* of set P.

The Delaunay tetrahedralization fills the interior of the convex hull of the points in V with tetrahedra. Thus, if all the points of V lie on the convex hull, the Delaunay tetrahedralization is a volumetric polyhedral representation of the object. Otherwise, some tetrahedra are eliminated in such a way that all points of V are on the boundary of the resulting polyhedral shape. This "sculpture" of the convex hull is done by sequentially eliminating one tetrahedron at a time, according to some heuristic criteria, until all points of V are on the boundary.

In order to ensure that, at each step, the boundary of the current three dimensional shape is

a polyhedron, topological constraints, summarized in the following three rules, must be respected (see Figure 16):

1. any tetrahedron with exactly three vertices on the current boundary can be eliminated,
2. any tetrahedron with exactly five edges on the current boundary can be eliminated, and
3. any tetrahedron not satisfying the previous rules cannot be eliminated.

The heuristic criteria try to minimize the variation in the curvature of the boundary, by eliminating less regular tetrahedra first. At the end of the sculpturing process, the set of remaining tetrahedra provides a volumetric representation of the object, and the triangular faces of these tetrahedra, which lie on the external boundary, correspond to a polyhedral approximation of the object surface.

Boissonnat suggests an implementation of the method with a worst case complexity equal to $O(n^2 \log n)$, where n is the number of input data points. A drawback of this algorithm is that the sculpturing process can get locked before all the innermost vertices have been included into the boundary, even if a closed polyedral boundary through all vertices exists. Moreover, it is not known if every Delaunay tetrahedralization contains a polyhedron through all its vertices.

The approach presented in [Vel93] iis based on a different geometric structure, called the γ-*neighborhood graph*. The γ-neighborhood graph is a parametric graph that unifies a number of graphs, such as the convex hull and the Delaunay graph, into a continuos spectrum of graphs that ranges from the Delaunay tetrahedralization to the complete graph.

Let V be the set of data points, and t be a triangle having three points $P_1, P_2, P_3 \in V$ as its vertices. Let r be the radius of the smallest possible sphere touching the three vertices, i.e., the one such that P_1, P_2, P_3 lie on its diametral circle. The radius of any sphere touching P_1, P_2, P_3 can be expressed as $r/(1-c)$, where c is a parameter in the range $[0, 1]$. Note that the case of a sphere with an infinite radius (i.e., a halfspace) is included. A negative sign may be attributed to parameter c, depending on whether the centers of the two spheres lie on the same or on opposite sides with respect to t; thus, the radius must be expressed as $r/(1-|c|)$, where $0 \leq c \leq 1$. The γ-graph $\gamma([-1,1],[d,1])$, for some parameter $d \in$ [-1,0], includes all triangles t for which there exist two values $c_0 \in$ [-1,1] and $c_1 \in [d,0]$ such that two spheres touching the three vertices of t, with radii $r/(1 - |c_0|)$ and $r/(1 - |c_1|)$, respectively, do not contain any point of V in their intersection. A smaller value of d corresponds to a smaller volume of the intersection, and, thus, more triangles are included in the γ-graph (see Figure 17 for some two-dimensional examples of a γ-graph). If $d = -1$, then both spheres are allowed to be two halfspaces, and the γ-graph contains all possible triangles defined by ternes of vertices in V. If $d = 0$, then the γ-graph is the same as the Delaunay tretrahedralization of V, since the Delaunay tetrahedralization can be characterized as the collection of all triangles t having vertices at three points $P_1, P_2, P_3 \in V$, such that there exists an empty sphere touching P_1, P_2 and P_3.

For any $d \in [-1, 0]$, graph $\gamma([-1,1],[d,1])$ has the convex hull of set V as its boundary. Triangles lying on the boundary of the γ-graph are thus called *hull triangles*. The polyhedron corresponding to the boundary of the given object is obtained by constricting the graph γ([-1, 1],[d, 1]) through iterative deletion of hull triangles. A γ-graph, from which triangles have been deleted, is called a *pruned* γ-graph. The constriction process is continued until all data points lie on the pruned graph hull.

Four triangles of the graph may implicitly form a tetrahedron; a tetrahedron formed by at least one hull triangle is called a *boundary tetrahedron*. The selection of the triangle $t \equiv P_1, P_2, P_3$

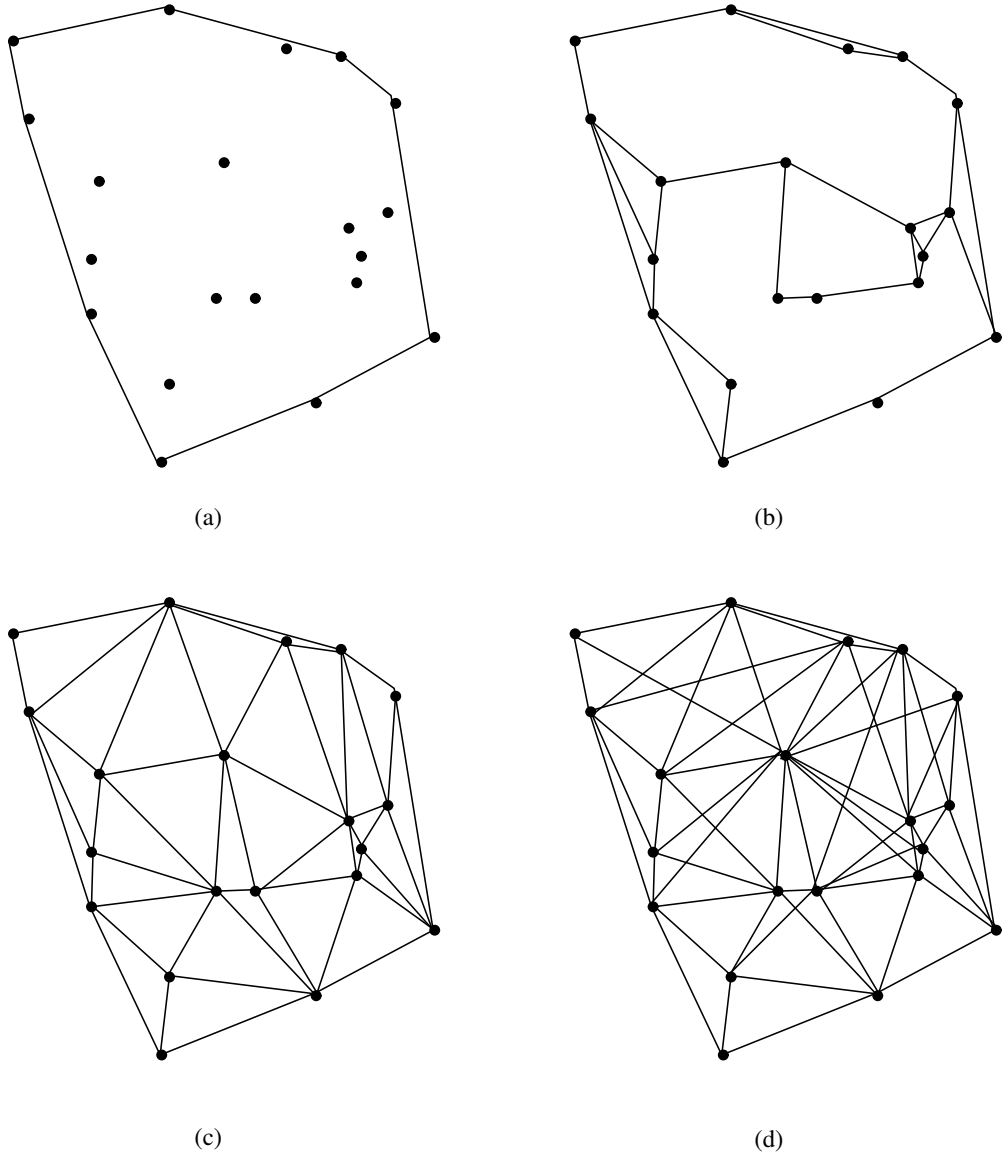


Figure 17: Two-dimensional examples of γ-graphs, containing a increasingly larger sets of edges: (a) $\gamma([-1,1],[1,1]$ (convex hull); (b) $\gamma([-1,1],[\frac{1}{4},1]$; (c) $\gamma([-1,1],[0,1]$ (Delaunay triangulation); (d) $\gamma([-1,1],[-\frac{1}{4},1]$;

to be deleted at a generic step is based on the observation that the opposite vertex P_4 of the tetrahedron P_1, P_2, P_3, P_4 that has the largest solid angle φ has the largest probability to be sensed from outside the boundary. Intuitively, the solid angle φ at P_4 depends on how close P_4 lies to t relative to the size of the tetrahedron, and on the shape of t. This notion is formalized by introducing a γ-indicator, associated with t with respect to P_4. If r is the radius of the smallest sphere touching P_1, P_2, P_3, and R is the radius of the sphere touching P_1, P_2, P_3, P_4, then the γ-indicator is defined as the value of c for which $R = r/(1-|c|)$, where the sign of c discriminates on which side the center of the sphere lies with respect to t.

Deleting a hull triangle $[P_1, P_2, P_3]$, due to the γ-indicator with respect to a vertex P_4, intuitively means that tetrahedron $[P_1, P_2, P_3, P_4]$ is merged into the "empty space". Note that the interior of a tetrahedron formed by triangles of the γ-graph can be intersected by other triangles of the graph. In order to leave the hull of the graph properly defined, any triangle crossing P_1, P_2, P_3, P_4 must also removed. In addition, the boundary polyhedron must remain connected. This is achieved by introduncing topological constraints, similar to the ones mentioned for the algorithm by Boissonnat, which forbid merging a tetrahedron into the free space in some cases.

Any $\gamma([-1,1],[d,1])$ can be used for constriction, but the existence of a solution can be guaranteed only for d "small enough". Unfortunately, the threeshold value is apriori unknown. On the other hand, a small value of d implies a larger graph and, thus, a higher computational complexity. It is possible, though, to start the constriction process from the graph $\gamma([-1,1],[0,1])$, (i.e., the Delaunay tetrahedralization), and adaptively add triangles of a $\gamma([-1,1],[d,1])$, $-1 \leq d < 0$, when necessary, so that the process never gets stuck.

The three dimensional $\gamma([-1,1],[d,1])$ graph can be constructed in $\Theta(n^2)$ time for $d = 0$, and $O(n^3)$ time for $d < 0$, where n is the number of points. Once the γ-neighborhood graph has been built, the boundary constriction algorithm takes $\Theta(t \log t)$ time, with t the number of triangles in the γ-graph.

5.2 Reconstruction from Scattered Segments

Reconstructing the boundary of an object from a set $\mathcal{S}$ of straight-line segments belonging to such boundary implies finding a simple closed polyhedron including the segments of $\mathcal{S}$ as a subset of its edges. Such edges are obtained through a stereo process applied from several viewpoints. The approach used by existing algorithms consists of constructing first a Delaunay tetrahedralization of the input including the given set of segments. Then, a suitable subset of tetrahedra is eliminated in such a way that all given segments lie on the boundary of the resulting shape. As in the two-dimensional case, the problem of including a given set of segments into a tetrahedralization can be faced from two perspectives, thus leading to the concepts of *constrained* and of *conforming Delaunay tetrahedralization.*

The problem of computing a constrained Delaunay tetrahedralization in $3D$ it has been shown to be NP-complete [Rup89], and, thus, not suitable for developing algorithms based on it. A reconstruction algorithm based on the use of a conforming Delaunay tetrahedralization has been proposed by Boissonnat et al. [Boi90]. Their algorithm for building a conforming Delaunay tetrahedralization, already described in Section 3.4 for the two-dimensional case, replaces each data segment with a set of points (which generates a set of points on the object boundary); then, a Delaunay tetrahedralization of the set of segments endpoints augmented with such additional set is computed. Unfortunately, there is no bound to the number of points which must be added to guarantee that all given segments are included in the resulting tetrahedralization.

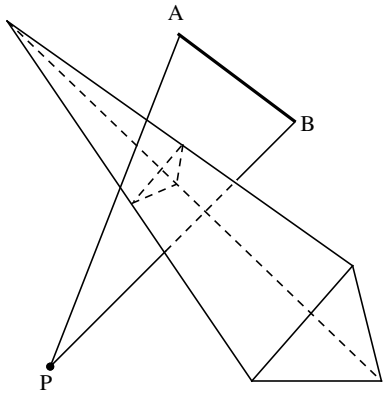


Figure 18: A tetrahedron which cannot be part of the object since it is intersected by a triangle connecting a stereo segment $\overline{AB}$ to the corresponding camera center P.

The conforming Delaunay tetrahedralization, obtained through the above mentioned method, is used as a starting point for reconstructing the boundary of the object. The basic idea is that triangles linking stereo segments to the viewpoints represent optical rays and, thus, cannot intersect the object (see Figure 18). Thus, for each triangle t having an edge in one of the segments and the vertex opposite to the edge in one of the camera optical center, all tetrahedra intersected by t are marked as empty. In general, a better result can be obtained by keeping, for each tetrahedra, a count of the number of times it has been crossed by a triangle, and marking only tetrahedra which have been intersected by a certain number of triangles.

Eliminating tetrahedra marked to be part of free space, is not enough to obtain a valid representation of the object. In fact, some tetrahedra that belong to the free space may not be intersected by any triangle joining a camera center with a measured segment, and thus remain in the final tetrahedralization. This can create singularities on the object surface. A valid representation of the object is obtained by removing tetrahedra which create such singularities.

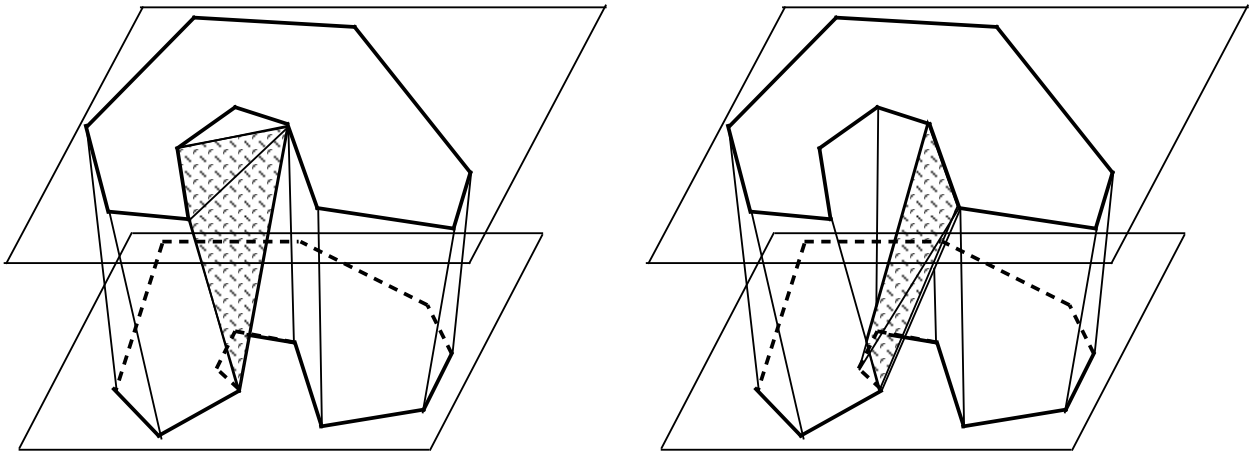
5.3 Reconstruction from Planar Cross Sections

The problem of reconstructing a surface from a sequence of planar cross-sections can be formulated as follows. Let $C_1, ..., C_n$ be a set of contours obtained by intersecting a three-dimensional object by n cutting planes $\Pi_1, ..., \Pi_n$. Each contour C_i consists of a collection of simple polygons (some possibly lying inside others). The aim is finding a simple closed polyhedron such that, for every $i = 1, \ldots, n$ the intersection of the polyhedron with plane Π_i is the same as contour C_i.

In existing approaches, the problem of constructing an object over the n cross section is reduced to the problem of constructing a sequence of $n-1$ partial shapes, each of them connecting two cross sections C_i and C_{i+1}, lying on adjacent planes. Methods proposed in the literature are based on two main techniques:

- reducing the problem to a path searching in a toroidal graph [Kep75, Fuc77];
- using Delaunay triangulations [Boi88].

By assuming that the resulting polyhedron has vertices on the given cross sections and consists of triangular faces only, it has been shown that the remaining problem of finding the edges of the polyhedron which spans the space between two consecutive contours reduces to that of finding a



(a) (b)

Figure 19: (a) A tetrahedron which must be eliminated since it is composed by edges not belonging to the cross section. (b) A tetrahedron which must be eliminated since it is connected to the remainder of the structure only by two edges.

minimum cost cycle in a toroidal graph [Fuc77]. Keppel [Kep75] was the first to reduce the problem of connecting vertices to a search in a toroidal graph. He used a maximum volume heuristic to select a path. Fuchs, Kedem and Uselton [Fuc77] presented the first efficient algorithm. Their method generalizes the results of Keppel and can be used with various optimization criteria. They proposed a minimal surface heuristic, but other optimization criteria may be used.

Boissonnat [Boi88] fills the slice of the 3D object lying between two adjacent planes Π_i and Π_{I+1} by starting from a Delaunay tetrahedralization, and successively deleting empty tetrahedra from it. Let M_j be the set of vertices of contour C_j, for a generic $j \in [0, n]$. The method computes first the Delaunay triangulations of the sets of points M_i and M_{i+1} of C_i and C_{i+1}. The three-dimensional Delaunay tetrahedralization of the points M_i and M_{i+1} is constructed from the two-dimensional ones. Note that both slice $(i-1, i)$ and slice $(i, i+1)$ intersect the cutting plane Π_i along the Delaunay triangulation of M_i, thus ensuring the coherence between between the two slices. Finally, the algorithm eliminates those tetrahedra having an edge in Π_i or in Π_{i+1} outside the contours, and those contributing to non-solid connections (see Figure 19). This procedure is repeated for all the pairs of adjacent cross-sections.

The complexity of the reconstruction algorithm is $O(t)$ if t is the maximum number of tetrahedra in the Delaunay triangulation between two adjacent planes Π_i and Π_{i+1}, containing $m = m_i + m_{i+1}$ points. In the worst case, t is $O(m^2)$, but when the contours in two adjacent cross sections are not too different, it is expected to be $O(m)$.

The triangle-based approach to surface reconstruction from cross-sections is not the only possible one. Sloan and Hrechanyc [Slo81] address the surface reconstruction problem as that of generating a sweeping rule for a generalized cylinder representation giving rise to the contours. This approach is expecially well-suited to deal with sparse data, and provides acceptable reconstructions from cross sections which are too sparse for the smoothness assumption common to the above mentioned triangulation algorithms.

6 Conclusions

We have presented a survey of methods for triangle-based description of surfaces defined by sets of points and straight-line segments in 3D space.

We have focused our attention on functional surfaces, i.e., surfaces defined by a function $z \equiv \phi(x, y)$ of two independent variables. The first problem in defining a triangle-based surface representation is the choice of an appropriate topological structure to connect the projections of the data points and of the constraint segments. Delaunay triangulation is the most common choice for this purposes. We have presented a classification of the different approaches to computing a Delaunay triangulation in both the standard and the constrained and conforming cases.

The reconstruction of the boundary of a three-dimensional object from scattered data has also been addressed. For this problem, input data may consist of a set of points, of a set of segments and of a sequence of parallel cross-sections. An appropriate topological structure connecting the data belonging to the surface to be reconstructed can be derived from a tetrahedralization of the 3D volume of the object. Again, a Delaunay tetrahedralization is usually chosen for such purposes. We have presented an overview of the major known approaches in literature and described geometric structures (Delaunay triangulations, possibly constrained or conforming, and extensions) used by such algorithms.

Acknowledgement

This work has been supported by the Stategic Project "Knowledge through Images: on Application to Cultural Heritage" of the Italian National Research Council through contract N. 94.04221.ST74.

References

[Asa86] Asano, T., Asano, T., Guibas, L., Hershberger, J. Imei, H., Visibility of disjoint polygons, *Algorithmica*, 1, 1986, pp. 45-63.

[Bar84] Barrera, R., Vaquez, A.M., A hierarchical method for representing relief, *Proceedings Pecora IX Symposium on Spatial Information Technologies for Remote Sensing Today end Tomorrow*, Sioux Falls, South Dakota, October 1984, pp. 87-92.

[Boi82] Boissonnat, J.D., Representation of objects by triangulating points in 3-D space, *Proceedings 6th International Conference on Pattern Recognition*, Munich, 1982, pp. 830-832.

[Boi84] Boissonnat, J.D., Geometric structures for three-dimensional shape representation, *ACM Transactions on Graphics*, 3(4), 1984, pp. 266-286.

[Boi88] Boissonnat, J. D, Shape Reconstruction from Planar Cross-Sections", *Computer Vision, Graphics and Image Processing*, 44, pp. 1-29, 1988.

[Boi90] Boissonnat, J.D., Faugeras, O.D., Le Bras-Mehlman, E., Representing Stereo Data with the Delaunay Triangulation, *Artificial Intelligence*, 44, pp. 41-89, 1990.

[Che86] Chen, Z.T., Tobler, W.R., Quadtree representation of digital terrain, *Proceedings Autocarto 86*, London, 1986, pp. 475-484.

[Che89] Chew, L.P., Constrained Delaunay triangulations, *Algorithmica*, 4, 1989, pp. 97-108.

[DeF89] De Floriani, L., A pyramidal data structure for triangle-based surface description, *IEEE Computer Graphics and Applications*, 8, March 1989, pp. 67-78.

[DeF92a] De Floriani, L., Puppo, E., An on-line algorithm for constrained Delaunay triangulation, *CVGIP, Graphical Models and Image Processing*, 54 (3), July 1992, pp. 290-300, Academic Press, Orlando, FL.

[DeF92] De Floriani, L., Puppo, E., A Hierarchical Triangle-based Model for Terrain Description, *Theories and Methods of Spatio-Temporal Reasoning in Geographic Space*, 1992, pp. 236-251.

[DeF93] De Floriani, Mirra, D., Puppo, E., Extracting Contour Lines from a Hierarchical Surface Model *Computer Graphics Forum (Proceedings Eurographics 93)*, 12(3), 1993, pp. 249-260.

[DeF94] De Floriani, L., Marzano, P., Puppo, E. (1994). Hierarchical Terrain Models: Survey and Formalization, *Proceedings ACM Symposium on Applied Computing* (SAC'94), Phoenix, Arizona, March 1994.

[Dil87] Dillencourt, M.B., A non-Hamiltonian, nondegenerate Delaunay triangulation", *Information Processing Letters*, 25(3), pp. 149-151, 1987.

[Ede87] Edelsbrunner, H., *Algorithms in Combinatorial Geometry*, Springer Verlag, 1987.

[Ede93] Edelsbrunner, H., Tan, T.S., An Upper Bound for Conforming Delaunay Triangulations, *Discrete Computational Geometry*, 10, 1993, pp. 197-213.

[For87] Fortune, S., A sweep-line algorithm for Voronoi diagrams, *Algorithmica*, 2(2), 1987, pp. 153-174.

[Fuc77] Fuchs, H., Usenton, S.P., Zedem, Z., Optimal surface reconstruction from planar contours, *Communications ACM*, 20(10), 1977, pp. 693-702.

[Gom79] Gomez, D., Guzman, A., Digital model for three-dimensional surface representation, *Geo-processing*, 1, 1979, pp. 53-70.

[Gui85] Guibas, L.J., Stolfi, J., Primitives for the manipulation of general subdivisions and the computation of Voronoi diagrams, *ACM Transactions on Graphics*, 4, 1985, pp. 74-123.

[Joe89] Joe, B., Wang, C.A., Duality and construction of constrained Voronoi diagrams and Delaunay triangulations, *Technical Report*, Dept. of Computer Science, University of Alberta.

[Kao91] Kao, T.C., Mount, D.M., Incremental construction and dynamic maintenance of constrained Delaunay triangulations, *Proceedings 3rd Canadian Conference on Computational Geometry*, 1991, pp. 170-175.

[Kep75] Keppel, E., Approximating complex surfaces by triangulation of contour lines, *IBM Journal of Research and Development*, 19(1), 1975, pp. 2-11.

[Knu92] Knuth, D.E., Guibas, L.J., Sharir, M., Randomized incremental construction of Delaunay and Voronoi diagrams, *Algorithmica*, 7 (4), 1992, pp. 381-413.

[Lee86] Lee, D.T., Lin A.K., Generalized Delaunay triangulation for planar graphs, *Discrete and Computational geometry*, 1, 1986, pp. 201-217.

[Lee80] Lee, D.T., Schacter, B.J., Two algorithms for constructing a Delaunay triangulation, *International Journal of Computer and Information Sciences*, 9 (3), 1980, pp. 219-242

[Law77] Lawson, C.L., Software for C^1 surface interpolation, *Mathematical Software III*, edited by J.R. Rice, Academic Press Inc., 1977, pp. 161-164.

[Man79] Manacher, G.K., Zobrist, A.L., Neither the greedy nor the Delaunay triangulation of a point set approximates the optimal triangulation, *Information Processing Letters*, 9, 1979, pp. 31-34.

[McL76] Mc Lain, D.H., Two-dimensional interpolation from random data, *The Computer Journal*, 19(2), 1976, pp. 178-181.

[Nac91] Nackman, L.R., Srinivasan, V., Point placement for Delaunay triangulation of polygonal domains, *Proceedings 3rd Canadian Conference on Computational Geometry*, 1991, pp. 37-40.

[Olo91] Oloufa, A.A., Triangulation applications in volume calculation, *Journal Comput. Civil Engrg.*, 5, 1991, pp. 103-119.

[ORo81] O'Rourke, J., Triangulation of minimal area as 3D object models, *Proceedings 7th International Conference on Artificial Intelligence*, 1981, pp. 664-666.

[Peu75] Peuker, T.K., Douglas, D.H., Detection of surface-specific points by local parallel processing of discrete terrain elevation data, *Computer Graphics and Image Processing*, 4, 1975, pp. 375-387.

[Pon87] Ponce, J., Faugeras, O., An object centered hierarchical representation for 3D objects: the prism tree, *Computer Vision, Graphics and Image Processing*, 38(1), 1987, pp. 1-28.

[Pre85] Preparata, F.P, Shamos, M.I., *Computational Geometry: an Introduction*, Springer verlag, 1985.

[Rip90] Rippa, S., Minimal roughness property of the Delaunay triangulation, *Computer Aided Geometric Design*, 7, 7, 1990, pp. 489-497.

[Rup89] Ruppert, J,, Seidel, R., On the difficulty of tetrahedralizing 3-dimensional non-convex polyhedra, *Proceedings 5th ACM Symposium on Computational Geometry*, 1989, pp. 380-393.

[Saa91] Saalfeld, A., Delaunay edge refinements, *Proceedings 3rd Canadian Conference on Computational Geometry*, 1991, 33-36.

[Sam92] Samet, H., Sivan, R., Algorithms for constructing quadtree surface maps. *Proceedings 5th International Symposium on Spatial Data Handling*, Charleston, 1992, pp. 361-370.

[Sca90] Scarlatos, L.L., Pavlidis, T., Hierarchical triangulation using terrain features, *Proceedings IEEE Conference on Visualization*, San Francisco, CA, October 1990.

[Sch87] Schubert, L. Wang, C.A., An optimal algorithm for constructing the Delaunay triangulation of a set of line segments, *Proceedings 3rd ACM Symposium on Computational Geometry*, Waterloo, June 1987, pp. 223-232.

[Sib78] Sibson, R., Locally equiangular triangulations, *The Computer Journal*, 21, 1878, pp. 243-245.

[Slo81] Sloan, K.R., Hrechanyk, L.D., Surface reconstruction from sparse data, *Proceedings Conference on Pattern Recognition and Image Processing*, 1981, pp. 45-48.

[Vel93] Veltkamp, R.C., 3D Computational Morphology, *EUROGRAPHICS '93*, pp. 115-127, 1993.

[Von87] Von Herzen, B., Barr, A.H., Accurate triangulations of deformed, intersecting surfaces, *Computer Graphics (Proceedings SIGGRAPH 87)*, 21(4), July 1987, pp. 103-110.

[Wat81] Watson, D.F., Computing the n-dimensional Delaunay tessellation with application to Voronoi polytopes, *The Computer Journal*, 24, 1981, pp. 728-746.