Tetrahedralisations and 3D Voronoi diagrams

Lesson 3.1

The Delaunay triangulation (DT) and the Voronoi diagram (VD) are fundamental data structures when dealing with spatial datasets, many computer scientists and mathematicians consider the VD as being the most fundamental spatial structure (or spatial model) because it is very simple, and yet is so powerful that it helps in solving many theoretical problems, as well as many real-world applications.

The DT and the VD are most often presented, described, and used, in two dimensions, but their concepts can be generalised to higher dimensions. We describe in this chapter the concepts in $\mathbb{R}^3$, and also discuss the n-dimensional cases when appropriate.

three-dimensional Euclidean space

We also discuss how the constrained and conforming DT can be generalised to $\mathbb{R}^3$.

To read or to watch.

The reader is advised to first read the Chapter *Triangulations & Voronoi diagram* in the book *Computational modelling of terrains* (Ledoux et al., 2020), where the 2D concepts are introduced.

1.1 The three-dimensional Voronoi Diagram

Let S be a set of points in $\mathbb{R}^d$. The Voronoi cell of a point $p \in S$, defined $\mathcal{V}_p$, is the set of points $x \in \mathbb{R}^d$ that are closer to p than to any other point in S; that is:

$$\mathcal{V}_p = \{x \in \mathbb{R}^d \mid \|x - p\| \leq \|x - q\|, \forall q \in S\} \tag{1.1}$$

The union of the Voronoi cells of all generating points $p \in S$ form the Voronoi diagram of S, defined VD(S). If S contains only two points p and q, then VD(S) is formed by a single hyperplane defined by all the points $x \in \mathbb{R}^d$ that are equidistant from p and q. This hyperplane is the perpendicular bisector of the line segment from p to q, and splits the space into two (open) half-spaces. $\mathcal{V}_p$ is formed by the half-space containing p, and $\mathcal{V}_q$ by the one containing q.

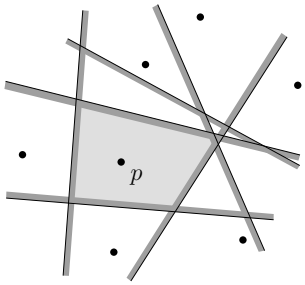


Figure 1.1: The Voronoi cell $\mathcal{V}_p$ is formed by the intersection of all the half-planes between p and the other points.

As shown in Figure 1.1, when S contains more than two points (let us say it contains n points), the Voronoi cell of a given point $p \in S$ is obtained by the intersection of $n - 1$ half-spaces defined by p and the other points $q \in S$. That means that $\mathcal{V}_p$ is always convex, in any dimensions. Notice also that every point $x \in \mathbb{R}^d$ has at least one nearest point in S, which means that VD(S) covers the entire space.

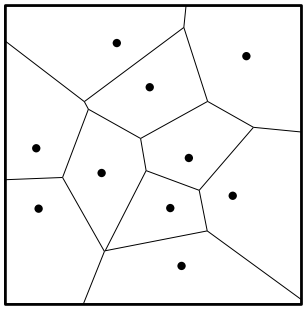


Figure 1.2: The VD for a set S of points in the plane (the black points).

As shown in Figures 1.2, the VD of a set S of points in $\mathbb{R}^2$ is a planar graph, but it can also be seen as a two-dimensional cell complex where each 2-cell is a (convex) polygon. Two Voronoi cells, $\mathcal{V}_p$ and $\mathcal{V}_q$, lie on the

(*last update*: *February* 11, 2021)

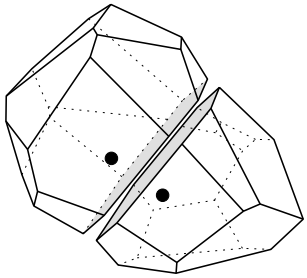


Figure 1.3: Two Voronoi cells adjacent to each other in $\mathbb{R}^3$, they share the grey face.

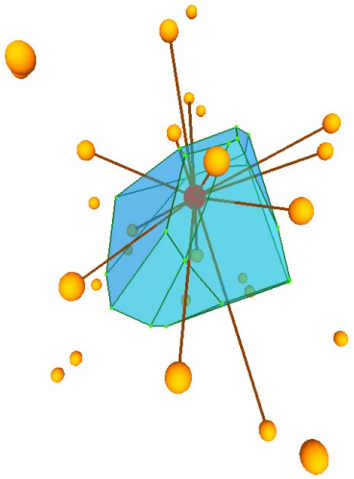


Figure 1.4: The Voronoi cell for the red vertex, the red edges are the Delaunay edges that are dual to the Voronoi facets.

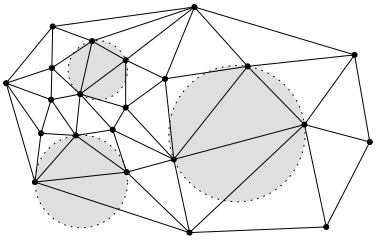


Figure 1.5: The DT of a set of points in the plane.

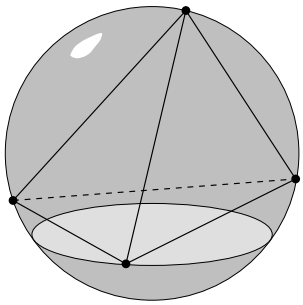


Figure 1.6: A Delaunay tetrahedron has an empty circumsphere.

opposite sides of the perpendicular bisector separating the points p and q.

In $\mathbb{R}^3$, VD(S) is a three-dimensional cell complex. The Voronoi cell of a point p is formed by the intersection of all the half-spaces (three-dimensional planes) between p and the other points in S. Drawing a picture of the three-dimensional case is not easy, thus Figure 1.3 shows two adjacent Voronoi cells (which are convex polyhedra), and Figure 1.4 one cell with its incident Delaunay edges.

The VD has many properties, and most of them are valid in any dimensions. Note that most of these properties are valid only when the set S of points is in *general position*, that is when for example in three dimensions no five points are cospherical, and no four points are collinear. Details concerning the possible degeneracies are given in Section 1.2.6. What follows is a list of the most relevant properties:

Size: if S has n points, then VD(S) has exactly n Voronoi cells since there is a one-to-one mapping between the points and the cells.

Voronoi vertices: in $\mathbb{R}^d$, a Voronoi vertex is equidistant from $(d+1)$ points. In $\mathbb{R}^3$, a Voronoi vertex is at the centre of a sphere defined by 4 points in S.

Voronoi edges: in $\mathbb{R}^d$, a Voronoi edge is equidistant from d points.

Voronoi faces: in $\mathbb{R}^d$, a Voronoi face is equidistant from $(d-1)$ points. Hence, in $\mathbb{R}^3$, it is the bisector plane perpendicular to the line segment joining two points.

Convex hull: let S be a set of points in $\mathbb{R}^d$, and p one of its points. $\mathcal{V}_p$ is unbounded if p bounds conv(S). Otherwise, $\mathcal{V}_p$ is the convex hull of its Voronoi vertices.

1.2 The Delaunay tetrahedralisation

The Delaunay triangulation of a set S of points in $\mathbb{R}^d$ is a simplicial complex where each d-simplex σ, formed by $d+1$ vertices in S, has an empty *circumball* (a ball is said to be *empty* when no points are in its interior). For $\mathbb{R}^3$, it is called the *Delaunay tetrahedralisation*: the space is tessellation into non-overlapping tetrahedra having an empty *circumsphere* (as shown in Figure 1.6).

1.2.1 Duality between the DT and the VD

The VD and the DT are dual to each other, and that in any dimensions. This means they represent the same thing but from a different point-of-view, and one structure can always be extracted from the other. Consider a graph embedded in $\mathbb{R}^d$ as a d-dimensional cell complex. The mappings between the elements of a cell complex in $\mathbb{R}^d$ are as follows: let C be a k-cell, the dual cell of C in $\mathbb{R}^d$ is denoted by $C^\star$ and is a $(d-k)$-cell.

The duality between the VD and the DT in $\mathbb{R}^3$ are thus as follows:

- a Delaunay vertex p becomes a Voronoi cell (Figure 1.7a);
- a Delaunay edge α becomes a Voronoi face (Figure 1.7b);
- a Delaunay triangular face κ becomes a Voronoi edge (Figure 1.7c);
- a Delaunay tetrahedron τ becomes a Voronoi vertex (Figure 1.7d).

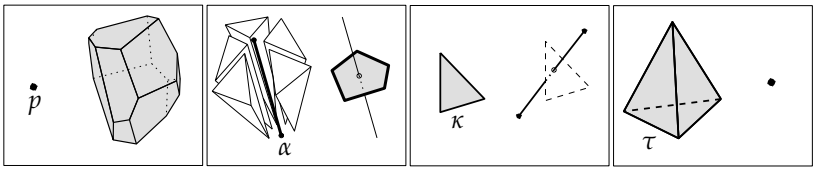


Figure 1.7: Duality in $\mathbb{R}^3$ between the elements of the VD and the DT.

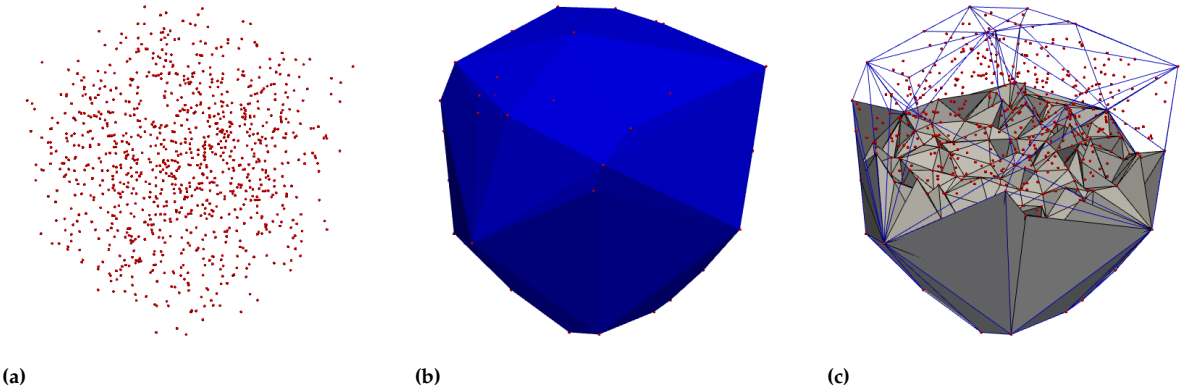


Figure 1.8: (a) A set of 1000 points randomly distributed in a cube. **(b)** Its convex hull. **(c)** The Delaunay tetrahedralisation of the points, 'sliced' in the middle and the upper tetrahedra removed (to be able to visualise the interior).

A Voronoi vertex is located at the centre of the sphere circumscribed to its dual tetrahedron, and two vertices in S have a Delaunay edge connecting them if and only if their two respective dual Voronoi cells are adjacent.

1.2.2 Convex Hull

In any dimensions, the DT of set S of points subdivides completely $\text{conv}(S)$, ie the union of all the simplices in $\text{DT}(S)$ is $\text{conv}(S)$. The boundary of a convex hull in 3D is formed of a set of triangles. Figure 1.8b shows an example.

1.2.3 Local Optimality

facet

Let $\mathcal{T}$ be a triangulation of S in $\mathbb{R}^d$. A facet σ (a $(d-1)$-simplex) is said to be *locally* Delaunay if it either:

(i) belongs to only one d-simplex, and thus bounds $\text{conv}(S)$, or
(ii) belongs to two d-simplices σ_a and σ_b, formed by the vertices of σ and respectively the vertices a and b, and b is outside of the circumball of σ_a.

The second case is illustrated in two dimensions in Figure 1.9a. In an arbitrary triangulation, not every facet that is locally Delaunay is necessarily a facet of $\text{DT}(S)$, but local optimality implies globally optimality in the case of the DT:

locally Delaunay

> Let $\mathcal{T}$ be a triangulation of a point set S in $\mathbb{R}^d$. If every facet of $\mathcal{T}$ is locally Delaunay, then $\mathcal{T}$ is the Delaunay triangulation of S.

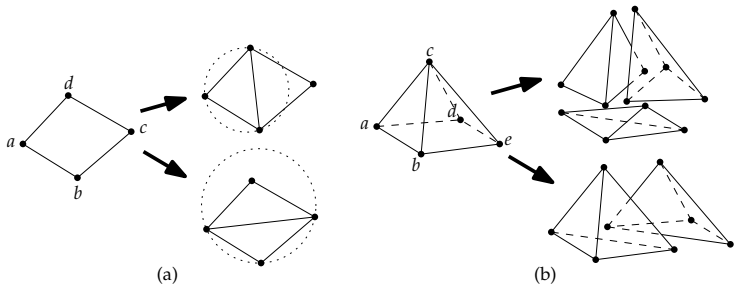


Figure 1.9: (a) A four-sided convex polygon $abcd$ can be triangulated in two different ways, but the empty circumcircle criterion guarantees that the triangles are as equilateral as possible. Notice that the edge ac is not locally Delaunay, but bd is. **(b)** In three dimensions, five vertices can be triangulated with either two or three tetrahedra. Although the tetrahedralisation at the bottom has two nicely shaped tetrahedra, they are not Delaunay (the point d is inside the sphere $abce$, which also implies that b is inside the sphere $acde$). The tetrahedralisation at the top respects the Delaunay criterion, but contains one very thin tetrahedron spanned by the points a, b, d and e.

This has serious implications as the DT—and its dual—are locally modifiable, ie we can theoretically insert, delete or move a points in S without recomputing DT(S) from scratch.

1.2.4 Angle Optimality

max-min angle optimality

The DT in two dimensions has a very important property that is useful in applications such as finite element meshing or interpolation: the *max-min angle optimality*. Among all the possible triangulations of a set S of points in $\mathbb{R}^2$, DT(S) maximises the minimum angle (max-min property), and also minimises the maximum circumradii. In other words, it creates triangles that are as equilateral as possible.

slivers

Finding 'good' tetrahedra, ie nicely shaped, is however more difficult than finding good triangles because the max-min property of Delaunay triangles does not generalise to three dimensions. A DT in $\mathbb{R}^3$ can indeed contain some tetrahedra, called *slivers*, whose four vertices are almost coplanar (see Figure 1.9b); these tetrahedra are Delaunay. Note that such slivers do not have two-dimensional counterparts.

For many applications where the Delaunay tetrahedralisation is used, eg in the finite element method in engineering or when the tetrahedra are used to perform interpolation directly, these tetrahedra are bad and must be removed. Why use the DT in three dimensions then? First, it should be said that in most cases Delaunay tetrahedra have in general a more desirable shape than arbitrary tetrahedra, they tend to favour 'round' tetrahedra. Second, the VD is not affected by them: Voronoi cells in three dimensions will still be 'relatively spherical' even if the DT has many slivers. Third, if the VD is used for interpolation, then the VD is necessary because many GIS operations use the properties of the VD (see Section 1.4.2), and if only one tetrahedron does not have an empty circumsphere, then the VD is corrupted.

1.2.5 Lifting on the paraboloid

There exists a close relationship between DTs in $\mathbb{R}^d$ and convex polytopes in $\mathbb{R}^{d+1}$.

Let S be a set of points in $\mathbb{R}^d$, and let $x_1, x_2, \ldots, x_d$ be the coordinates axes. The parabolic lifting map projects each vertex $v(v_{x1}, v_{x2}, \ldots, v_{xd})$ to a vertex $v^+(v_{x1}, v_{x2}, \ldots, v_{xd}, v_{x1}^2 + v_{x2}^2 + \cdots + v_{xd}^2)$ on the paraboloid of revolution in $\mathbb{R}^{d+1}$. The set of points thus obtained is denoted S^+. Observe that, for the two-dimensional case, the paraboloid in three dimensions defines a surface whose vertical cross sections are parabolas, and whose horizontal cross sections are circles; the same ideas are valid in higher dimensions.

The relationship is the following: every facet (a d-dimensional simplex) of the lower envelope of conv(S^+) projects to a d-simplex of the Delaunay triangulation of S. This is illustrated in Figure 1.10 for the construction of the DT in $\mathbb{R}^2$.

In short, the construction of the d-dimensional DT can be transformed into the construction of the convex hull of the lifted set of points in $(d + 1)$ dimensions. In practice, since it is easier to construct convex hulls (especially in higher dimensions, ie 4+), the DT is often constructed with this method.

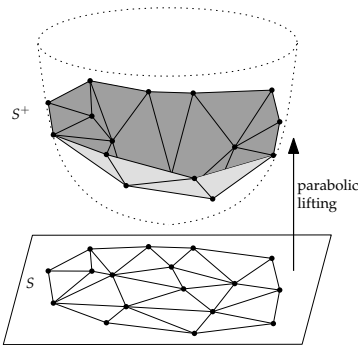


Figure 1.10: The parabolic lifting map for a set S of points $\mathbb{R}^2$.

1.2.6 Degeneracies

The previous definitions of the VD and the DT assumed that the set S of points is in general position, ie the distribution of points does not create any ambiguity in the two structures. For the VD/DT in $\mathbb{R}^d$, the degeneracies, or special cases, occur when $d + 1$ points lie on the same hyperplane and/or when $d + 2$ points lie on the same ball. For example, in three dimensions, when five or more points in S are cospherical there is an ambiguity in the definition of DT(S). This implies that DT(S) is not unique; VD(S) is still unique, but it has different properties.

1.3 Construction of the 3D DT/VD

As is the case in 2D, there exist several algorithms to construct either the DT or the VD from a set of points in 3D.

Mainly three paradigms of computational geometry can be used for computing a Delaunay triangulation in two and three dimensions: divide-and-conquer, sweep plane, and incremental insertion. In two dimensions, each one of these paradigms yields an optimal algorithm. In three dimensions, things are a bit more complicated. Divide-and-conquer algorithms have a worst time complexity of $\mathcal{O}(n^3)$, although in practice they are subquadratic. Only incremental insertion algorithms have a complexity that is worst-case optimal, ie $\mathcal{O}(n^2)$ since the complexity of the DT in $\mathbb{R}^3$ is quadratic. That is, there are configurations of n points that yield a DT with $\mathcal{O}(n^2)$ tetrahedra.

And as is the case in 2D, it is often simpler to reconstruct and store the DT (because they have only 4 vertices and 4 neighbours) and to extract the VD on-the-fly when needed.

The details of the algorithms are out of scope for this course. We provide in the following a general idea of how the reconstruction of the DT is performed in 3D by generalising the algorithm described in GEO1015.

Algorithm 1: Algorithm to insert one point in a DT

```
Input: A DT(S) 𝒯 in ℝ³, and a new point p to insert
Output: 𝒯^p = 𝒯 ∪ {p}
find tetrahedron τ containing p
insert p in τ by splitting it in to 4 new tetrahedra (flip14)
push 4 new tetrahedra on a stack
while stack is non-empty do
    τ = {p, a, b, c} ← pop from stack
    τ_a = {a, b, c, d} ← get adjacent tetrahedron of τ having the edge
     abc as a face
    if d is inside circumsphere of τ then
        if configuration of τ and τ_a allows it then
            flip the tetrahedra τ and τ_a (flip23 or flip32)
            push 2 or 3 new tetrahedra on stack
        else
            Do nothing
```

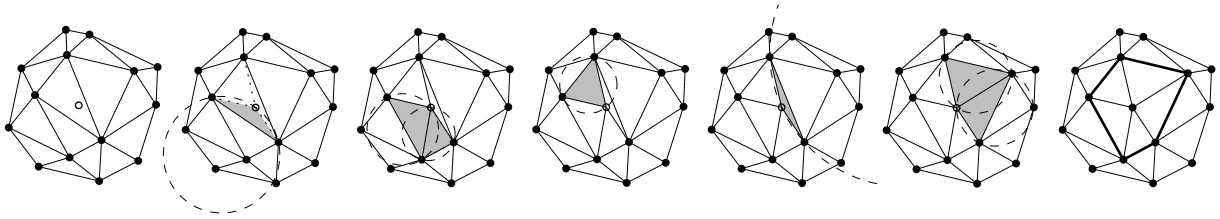


Figure 1.11: Step-by-step insertion, with flips, of a single point in a DT in two dimensions.

How does it work in practice?

Even more than in 2D, the duality between the convex hull in $d+1$-dimension and the DT in d-dimension is in practice exploited. Indeed, one can construct the convex hull of a set of points projected to 4D to obtain the DT in 3D. One popular and widely used implementation is Qhull (`http://www.qhull.org/`).

1.3.1 Generalisation of the flip-based incremental insertion algorithm

The algorithm described in Algorithm 1 is a generalisation to 3D of the flip-based incremental insertion algorithm used for 2D DT.

Most steps can be generalised in a direct way. Figure 1.11 shows the steps from the 2D algorithm, which are conceptually the same for the 3D generalisation of the algorithm (and it is more difficult to draw these steps in 3D).

flips

As is the case with the two-dimensional algorithm, the point p is first inserted in $\mathcal{T}$ with a flip (*flip14* in the case here), and the new tetrahedra created must be tested to make sure they are Delaunay. The sequence of flips needed is controlled by a stack containing all the tetrahedra that have not been tested yet. The stack starts with the four resulting tetrahedra of the *flip14*, and each time a flip is performed, the new tetrahedra created

are added to the stack. The algorithms stops when all the tetrahedra incident to p are Delaunay, which also means that the stack is empty.

Initialisation: the big tetrahedron. A DT is initialised with a tetrahedron several times larger than the spatial extent of S. The points in S are therefore always added inside an existing tetrahedron.

Walk/Point location. To find the tetrahedron containing the newly inserted point p, the adjacency relationships between the tetrahedra can be used. With a series of Orient tests one can navigate from one tetrahedron to the other.

Flips. A flip is a local (topological) operation that modifies the configuration of some adjacent tetrahedra. In 2D, for 4 points, a flip (called *flip22*), modifies the configuration of 2 adjacent triangles by flipping the diagonal of the quadrilateral. In 3D, there are 2 kinds of flips: *flip23* and *flip32*. Consider the set $S = \{a, b, c, d, e\}$ of points in general position in $\mathbb{R}^3$ and its convex hull conv(S). There exist two possible configurations, as shown in Figure 1.12:

1. the five points of S lie on the boundary of conv(S); see Figure 1.12a. There are exactly two ways to tetrahedralise such a polyhedron: either with two or three tetrahedra. In the first case, the two tetrahedra share a triangular face bcd, and in the latter case the three tetrahedra all have a common edge ae.
2. one point e of S does not lie on the boundary of conv(S), thus conv(S) forms a tetrahedron; see Figure 1.12b. The only way to tetrahedralise S is with four tetrahedra all incident to e.

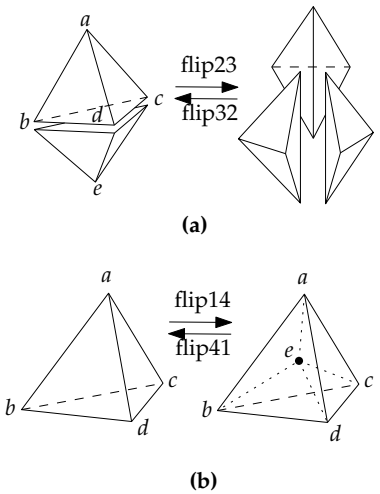


Figure 1.12: The 4 different kinds of flips in 3D.

Based on these two configurations, four types of flips in $\mathbb{R}^3$ can be described: *flip23*, *flip32*, *flip14* and *flip41* (the numbers refer to the number of tetrahedra before and after the flip). When S is in the first configuration, two types of flips are possible: a *flip23* is the operation that transforms one tetrahedralisation of two tetrahedra into another one with three tetrahedra; and a *flip32* is the inverse operation. If S is tetrahedralised with two tetrahedra and the triangular face bcd is not locally Delaunay, then a *flip23* will create three tetrahedra whose faces are locally Delaunay.

A *flip14* refers to the operation of inserting a vertex inside a tetrahedron, and splitting it into four tetrahedra; and a *flip41* is the inverse operation that deletes a vertex.

Flips can not always be applied during an insertion, it depends on the local configuration. For example, in Figure 1.12a, a *flip23* is possible on the two adjacent tetrahedra $abcd$ and $bcde$ if and only if the line ae passes through the triangular face bcd (which also means that the union of $abcd$ and $bcde$ is a convex polyhedron). If not, then a *flip32* is possible if and only if there exists in the tetrahedralisation a third tetrahedron adjacent to both $abcd$ and $bcde$.

1.3.2 Predicates

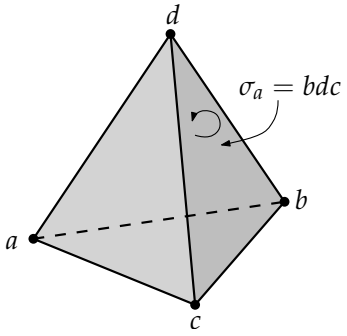


Figure 1.13: The tetrahedron $abcd$ is correctly oriented since ORIENT (a, b, c, d) returns a positive result. The arrow indicates the correct orientation for the face σ_a, so that ORIENT (σ_a, a) returns a positive result.

The 'orientation' of points in three dimensions is somewhat tricky because, unlike in two dimensions, we can not simply rely on the counter-clockwise orientation. In three dimensions, the orientation is always relative to another point of reference, ie given three points we cannot say if a fourth one is left of right, this depends on the orientation of the three points. When dealing with a single tetrahedron τ formed by the four vertices a, b, c and d (as in Figure 1.13), we say that τ is correctly oriented if ORIENT (a, b, c, d) returns a positive value. Notice that if two vertices are swapped in the order, then the result is the opposite (ie ORIENT (a, c, b, d) returns a negative value).

Vertices forming a face in a tetrahedron τ can also be ordered. As shown in Figure 1.13, a face σ_a, formed by the vertices b, c and d, is correctly oriented if ORIENT (σ_a, a) gives a positive result—in the case here, ORIENT (b, c, d, a) gives a negative result, therefore the correct orientation of σ_a is cbd. Observe that the face bcd is called σ_a because it is 'mapped' to the vertex a that is opposite; each of the four faces of a tetrahedron can be referred to in this way.

ORIENT determines if a point p is over, under or lies on a plane defined by three points a, b and c. It returns a positive value when the point p is above the plane defined by a, b and c; a negative value if p is under the plane; and exactly 0 if p is directly on the plane. ORIENT is consistent with the left-hand rule: when the ordering of a, b and c follows the direction of rotation of the curled fingers of the left hand, then the thumb points towards the positive side (the above side of the plane). In other words, if the three points defining a plane are viewed clockwise from a viewpoint, then this viewpoint defines the positive side the plane.

ORIENT can be implemented as the determinant of a matrix:

$$\text{ORIENT}(a, b, c, p) = \begin{vmatrix} a_x & a_y & a_z & 1 \\ b_x & b_y & b_z & 1 \\ c_x & c_y & c_z & 1 \\ p_x & p_y & p_z & 1 \end{vmatrix} \tag{1.2}$$

The predicate INSPHERE follows the same idea: a positive value is returned if p is inside the sphere; a negative if p is outside; and exactly 0 if p is directly on the sphere. Observe that to obtain these results, the points a, b, c and d in INSPHERE must be ordered such that ORIENT (a, b, c, d) returns a positive value.

It should be noticed that INSPHERE is derived from the parabolic lifting map (see Section 1.2.5). It is simply transformed into a four-dimensional ORIENT test: p is inside (outside) the sphere $abcd$ if and only if p^+ lies under (above) the hyperplane $a^+b^+c^+d^+$, and directly on the sphere if p^+ lies on the hyperplane $a^+b^+c^+d^+$.

$$\text{INSPHERE}(a, b, c, d, p) = \begin{vmatrix} a_x & a_y & a_z & a_x^2 + a_y^2 + a_z^2 & 1 \\ b_x & b_y & b_z & b_x^2 + b_y^2 + b_z^2 & 1 \\ c_x & c_y & c_z & c_x^2 + c_y^2 + c_z^2 & 1 \\ d_x & d_y & d_z & d_x^2 + d_y^2 + d_z^2 & 1 \\ p_x & p_y & p_z & p_x^2 + p_y^2 + p_z^2 & 1 \end{vmatrix} \tag{1.3}$$

1.3.3 Data structure

Instead of storing triangles as the atom, tetrahedra are used, they have 4 pointers to their 4 vertices, and 4 pointers to their 4 adjacent tetrahedra. All of them must be oriented correctly (as is the case in 2D where they are all counter-clockwise), as defined above.

1.3.4 Extracting the VD from the DT

Let $\mathcal{T}$ be the DT of a set S of points in $\mathbb{R}^3$. The simplices of the dual $\mathcal{D}$ of $\mathcal{T}$ can be computed as follows (all the examples refer to Figure 1.7):

- **Vertex:** a single Voronoi vertex is easily extracted—it is located at the centre of the sphere passing through the four vertices of its dual tetrahedron τ.
- **Edge:** a Voronoi edge, which is dual to a triangular face κ, is formed by the two Voronoi vertices dual to the two tetrahedra sharing κ.
- **Face:** a Voronoi face, which is dual to a Delaunay edge α, is formed by all the vertices that are dual to the Delaunay tetrahedra incident to α. The idea is simply to 'turn' around a Delaunay edge and extract all the Voronoi vertices. These are guaranteed to be coplanar, and the face is guaranteed to be convex.
- **Polyhedron:** the construction of one Voronoi cell $\mathcal{V}_p$, dual to a vertex p, is similar: it is formed by all the Voronoi vertices dual to the tetrahedra incident to p. Since a Voronoi cell is convex by definition, it is possible to collect all the Voronoi vertices and then compute the convex hull; the retrieval of all the tetrahedra incident to p can be done by performing a breadth-first search-like algorithm on the graph dual to the tetrahedra. A simpler method consists of first identifying all the edges incident to p, and then extracting the dual face of each edge.

Given $\mathcal{T}$, we must obviously visit all its 3-simplices to be able to extract $\mathcal{D}$. This means that computing $\mathcal{D}$ from $\mathcal{T}$ has a complexity of $\Theta(n)$ when S contains n points.

1.4 Applications of the DT and the VD

1.4.1 Modelling continuous 3D fields (as an alternative to voxels)

The objects studied in geoscience are often not man-made objects, but rather the spatial distribution of three-dimensional continuous geographical phenomena such as the salinity of a body of water, the humidity of the air, or the percentage of gold in the rock. These are referred to as fields, and raster structures (voxels or octrees) are the most popular solutions for modelling them. However, using regular structures has shortcomings and therefore the VD is a viable alternative.

One advantage is that the VD will adapt to the anisotropic distribution of the samples collected to study a field, these samples are three-dimensional points (x, y, z) to which an attribute is attached (eg the percentage of a

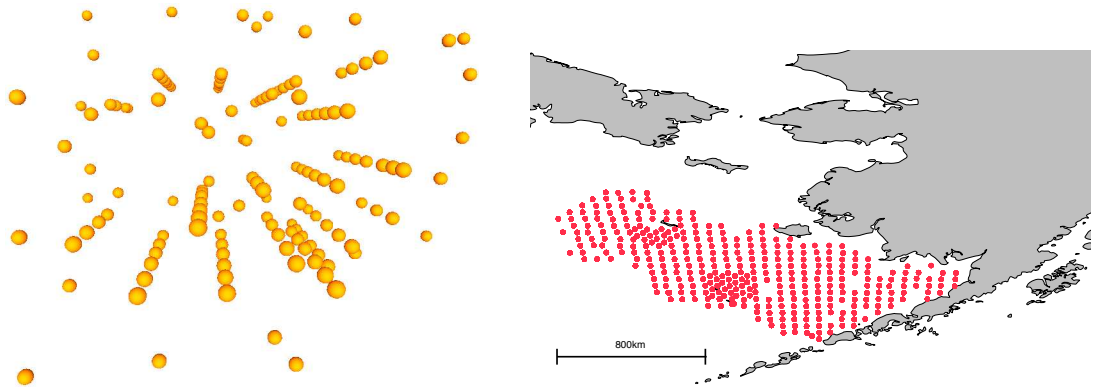


Figure 1.14: (a) Example of a dataset in geology, where samples were collected by drilling a hole in the ground. Each sample has a location in 3D space ($x - y - z$ coordinates) and one or more attributes attached to it. **(b)** An oceanographic dataset in the Bering Sea in which samples are distributed along water columns. Each red point represents a (vertical) water column, where samples are collected every 2m, but water columns are about 35km from each other.

certain mineral in a body of water). In practice, the samples can be very hard and expensive to collect because of the difficulties encountered and the technologies involved. To collect samples in the ground we must dig holes or use other devices (eg ultrasound penetrating the ground); underwater samples are collected by instruments moved vertically under a boat, or by automated vehicles; and samples of the atmosphere must be collected by devices attached to balloons or aircraft. As shown in Figure1.14, samples are often abundant vertically but very sparse horizontally.

Another advantage is that the VD can be efficiently and robustly reconstructed, and that based on it the samples can be interpolated to obtain an estimation of the attribute at any location, see below for details.

Finally, the tessellations of the VD (and the DT) make possible, and even optimise, several spatial analysis and visualisation operations.

1.4.2 Spatial interpolation

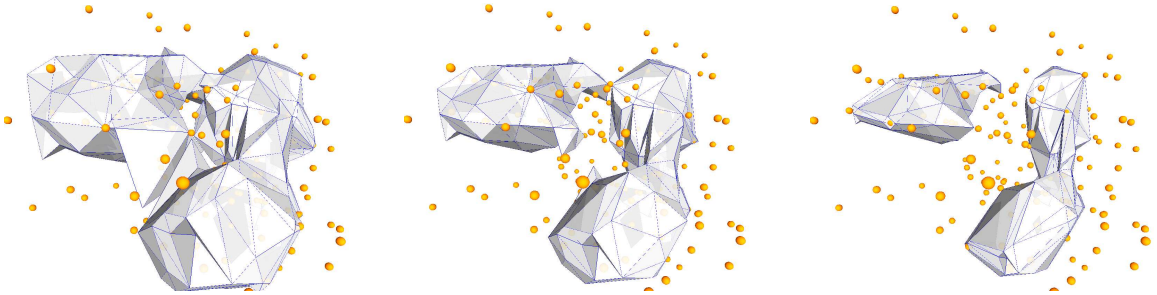
Given a set of samples, embedded in three-dimension, to which an attribute a is attached, spatial interpolation permits us to reconstruct the field that was sampled.

As is the case in 2D, the properties of both the 3DVD and the 3DDT can be used to estimate the value of an attribute.

Chapter 12 presents in details how to extend to three dimensions the usual interpolation methods used in GIS, and discusses whether they preserve their properties or are appropriate for geoscientific datasets.

1.4.3 Iso-surfaces

Given a set of samples from a trivariate field $f(x, y, z) = a$, an isosurface is the set of points in space where $f(x, y, z) = a_0$, where a_0 is a constant. Isosurfaces, also called *level sets*, are the three-dimensional analogous



(a) (b) (c)

Figure 1.15: An example of an oceanographic dataset where each point has the temperature of the water, and three isosurface extracted (for a value of respectively 2.0, 2.5 and 3.5) from this dataset.

concept to isolines (also called contour lines), which have been traditionally used to represent the elevation in topographic maps. Figure 1.15 shows one concrete example.

As explained in Chapter 12, isosurfaces can be extracted automatically from the DT.

1.5 Constrained tetrahedralisations

As is the case in 2D, given as input a set of points, straight-line segments, and faces embedded in $\mathbb{R}^3$, two different Delaunay tetrahedralisations are possible:

- conforming Delaunay tetrahedralisation (ConfDT)
- constrained Delaunay tetrahedralisation (ConsDT)

Both tetrahedralisations covers the convex hull of $\mathcal{P}$, respect every polygon (which can be represented by one or more triangles), and include every segment (which can be one of more edges in the tetrahedralisation) and vertex.

The typical input of a Delaunay tetrahedralisation program (or algorithm) is a called *piecewise linear complex* (PLC). A PLC $\mathcal{P}$ is a set of linear d-cells (where $0 \leq d \leq 3$), that satisfy the following properties:

1. the boundary of a d-cell in $\mathcal{P}$ is a union of cells in $\mathcal{P}$
2. if two distinct cell $f, g \in \mathcal{P}$ intersect, their intersection is a union of cells in $\mathcal{P}$.

Figure 1.16 shows one example. As shown in Figure 1.17, in practice this means that polygons cannot intersect other polygons (there needs to be a vertex and/or edges), but there are otherwise no restriction on the shapes that can be represented. Observe also that a PLC is flexible and allows unconnected (ie 'floating') vertices, edges, and faces (an edge can for instance be inside a polygon). Dangling edges, such as the one in Figure 1.16, are also allowed. The domain represented by a PLC does not have to represent a volume, it can be simply a set of points and surfaces that act as constraints for the tetrahedralisation.

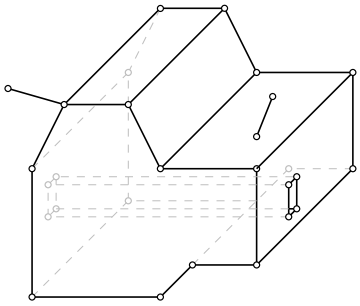


Figure 1.16: A PLC representing a solid (with a genus of 1) and having one dangling left; notice also that one extra edge is on a polygon. **Right:** These two polygons do not form a valid PLC because their intersection is not formed of vertices and edges in the PLC.

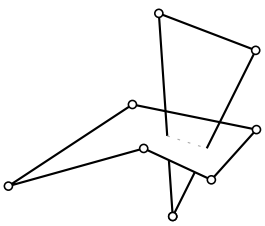


Figure 1.17: These two polygons do not form a valid PLC because their intersection is not formed of vertices and edges in the PLC.

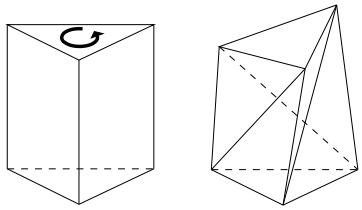


Figure 1.18: The Schönhardt polyhedron is impossible to tetrahedralise without adding extra vertices inside.

Tetrahedralisation of a polyhedron. While any polygon in two dimensions can be triangulated, some arbitrary polyhedra cannot be tetrahedralised without the addition of extra vertices, the so-called Steiner points. Figure 1.18 shows a simple example, called the Schönhardt polyhedron after the mathematician who first described the case. This polyhedron is formed by twisting the top face of a triangular prism to form a 6-vertex polyhedron having eight triangular faces (each one of the three quadrilateral faces adjacent to the top face will fold into two triangles). It is impossible to select four vertices of the polyhedron such that a tetrahedron is totally contained inside the polyhedron, as none of the vertices of the bottom face can directly 'see' the three vertices of the top triangular face.

Conforming DT (ConfDT). A ConfDT is a tetrahedralisation where every tetrahedron has an empty circumsphere, it is thus a 'real' Delaunay tetrahedralisation. This is achieved by adding new extra points to the input PLC $\mathcal{P}$ to ensure that the input constraints are present in the ConfDT. The extra points are called, as is the case in 2D, *Steiner* points.

Steiner points

It is known that every n-vertex PLC has a Steiner tetrahedralisation with at most $\mathcal{O}(n^2)$ vertices; notice here that this tetrahedralisation is *not* necessarily Delaunay.

Obtaining a ConfDT might require inserting significantly more than this, when for instance two or more polygons form a very small angle.

In fact, there do not exist any algorithm that guarantees to insert a polynomial number of vertices.

Most implementations will insert several new vertices, which are often unnecessary. Because of this, ConfDT are less used in practice.

Constrained DT (ConsDT). Given a PLC $\mathcal{P}$, the ConsDT is similar to the Delaunay tetrahedralisation, but the tetrahedra in ConsDT are not necessarily Delaunay (ie their circumsphere might contain other points from $\mathcal{P}$). The empty circumsphere for a ConsDT is less strict: a tetrahedron is Delaunay if its circumsphere contains no other points in $\mathcal{P}$ that are *visible* from the tetrahedron; the constraints polygons in $\mathcal{P}$ act a visibility blockers.

Thus, the ConsDT aims at keeping the Delaunay properties, but relaxes them to be able to respect the constraints (edges and polygons in the PLC).

However, unlike in 2D where it is known that for a set S of points and straight-line segments there is always a ConsDT possible, in 3D this is not the case. As explained above, this is linking to the fact that simple PLC cannot be tetrahedralised at all. As a consequence, the ConsDT of a PLC in 3D allows extra Steiner vertices to be inserted. The existing algorithms (and their implementations) that will insert far fewer vertices in a ConsDT than in a ConfDT. The details of the algorithms are beyond what is covered in this course.

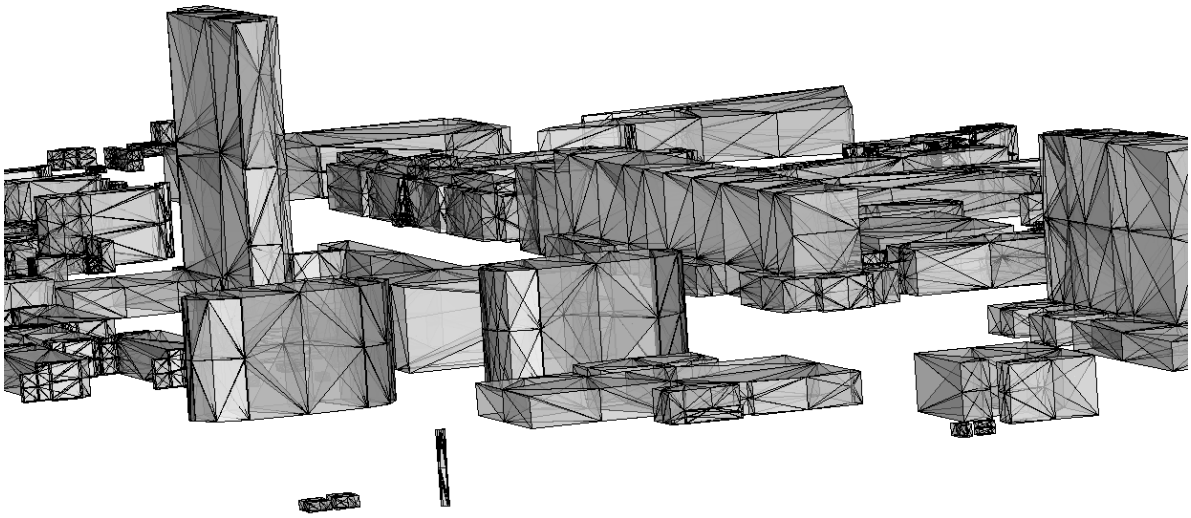


Figure 1.19: The LoD1 3D model of the TU Delft where each building is represented with the ConsDT of its PLC.

How does it work in practice?

Implementing a ConsDT that is robust against all input is difficult, and there exists few reliable libraries. Perhaps the "best" and easiest to use is TetGen, which is open-source; it is available at `http://www.tetgen.org/`. Beware: it expects a *perfect* input PLC, which is often not available for 3D geographical datasets that are made available by municipalities and governments; see Lesson 5.2.

1.6 Notes and comments

Rajan (1991) shows that the smallest sphere containing a Delaunay tetrahedron is smaller than the one of any other tetrahedron, ie the Delaunay criterion favours 'round' tetrahedra.

Cignoni et al. (1998) developed an algorithm, called `DeWall` and based on the divide-and-conquer paradigm, for constructing the DT in any dimensions. Although the worst-time complexity of this algorithm is $\mathcal{O}(n^3)$ in three dimensions, they affirm that the speed of their implementation is comparable to the implementation of known incremental algorithms, and is sub-quadratic.

The Schönhardt polyhedron was first described in Schönhardt (1928).

The algorithm to construct the 3D DT is adapted from Joe, 1991, and is conceptually the same as Edelsbrunner and Shah, 1996. See Ledoux (2007) for an easy explanation of the steps to construct the 3D DT/VD for a set of points, including the handling of the degeneracies.

Ledoux and Gold (2008) presents an overview of why the VD is a better alternatives to grids for the modelling of geoscientific fields.

1.7 Exercises

1. A DT contains 32 tetrahedra and we insert a new point p that falls inside one of the tetrahedra. If we insert and update the tetrahedralisation (for the Delaunay criterion), what is the number of tetrahedra?
2. If a given vertex v in a DT has 18 incident tetrahedra, how many vertices will its dual Voronoi cell contain?
3. Take a cube and try tetrahedralise it (not necessarily into Delaunay tetrahedra). How many tetrahedra do you get?
4. If $a = (1, 1, 2)$, $b = (4, 2, 2)$, $c = (3, 3, 2)$, and $d = (4, 3, 3)$. Is the value returned by Orient (a, b, c, d) positive, negative, or 0?

Bibliography

Cignoni, P., C. Montani, and R. Scopigno (1998). DeWall: A fast divide & conquer Delaunay triangulation algorithm in E^d. *Computer-Aided Design* 30.5, pp. 333–341.

Edelsbrunner, H. and N. R. Shah (1996). Incremental topological flipping works for regular triangulations. *Algorithmica* 15, pp. 223–241.

Joe, B. (1991). Construction of three-dimensional Delaunay triangulations using local transformations. *Computer Aided Geometric Design* 8, pp. 123–142.

Ledoux, H. (2007). Computing the 3D Voronoi Diagram Robustly: An Easy Explanation. *Proceedings 4th International Symposium on Voronoi Diagrams in Science and Engineering*. Pontypridd, Wales, UK: IEEE Computer Society, pp. 117–129.

Ledoux, H. and C. M. Gold (2008). Modelling three-dimensional geoscientific fields with the Voronoi diagram and its dual. *International Journal of Geographical Information Science* 22.5, pp. 547–574.

Ledoux, H., K. A. Ohori, and R. Peters (2020). *Computational modelling of terrains*. https://doi.org/10.5281/zenodo.3992107. Self-published.

Rajan, V. T. (1991). Optimality of the Delaunay triangulation in $\mathbb{R}^d$. *Proceedings 7th Annual Symposium on Computational Geometry*. North Conway, New Hampshire, USA: ACM Press, pp. 357–363.

Schönhardt, E. (1928). Über die zerlegung von dreieckspolyedern in tetraeder. *Mathematische Annalen* 98, pp. 309–312.