Journal of Convex Analysis
Volume 29 (2022), No. 1, 143–156

The Lifting Projection of Convex Polyhedra for Finding Delaunay Triangulations

Phan Thanh An*
Ho Chi Minh City University of Technology, Ho Chi Minh City, Vietnam; and: Vietnam National University, Ho Chi Minh City, Vietnam
thanhan@hcmut.edu.vn

Nam Dung Hoang
University of Science, Vietnam National University, Thanh Xuan, Hanoi, Vietnam
hoangnamdung@hus.edu.vn

Nguyen Kieu Linh
Posts and Telecommunications Institute of Technology, Nguyen Trai, Hanoi, Vietnam
linhnk@ptit.edu.vn

Received: October 12, 2020
Accepted: March 2, 2021

D. Walkup and R. J-B Wets's lifting projection of convex polyhedra in 1969 is used for finding the Delaunay triangulation of a finite planar point set. In concrete, the finite planar point set is lifted on the surface of a paraboloid in $\mathbb{R}^3$ with center outside the convex hull of the set. To find quickly the lower convex hull of these points on the paraboloid a restricted region is proposed, thereby eliminating a large number of points to be calculated. The numerical experiments also show that our new version algorithm significantly reduces the running time.

Keywords: Computing science, convex hull, Delaunay triangulation, extreme edge, gift-wrapping algorithm, lifting projection, lower convex hull, pattern recognition, restricted region, Voronoi diagram.

2010 Mathematics Subject Classification: Primary 52A15, 52B55; secondary 52A30, 52B10, 68U05.

1. Introduction

Delaunay triangulation and Voronoi diagram (two dual structures) are classical problems and important structures in convex geometry and computational geometry. They have many applications in different areas such as pattern recognition [1], computer graphics [10], image processing [26], [29], modeling a surface wildfire propagation through a complex landscape [27], and further applications. Many algorithms for computing Delaunay triangulation and Voronoi diagrams have been proposed by computer scientists as well as mathematicians. Guibas and Stolfi proposed in 1985 a Divide and Conquer algorithm [16], Fortune introduced a Sweepline algorithm for Voronoi diagrams [14] in 1987. However, these algorithms are not easy to implement on a computer. To overcome this drawback, Guibas presented an $O(n \log n)$ optimal randomized incremental algorithm [15] in 1992 and an incremental algorithm [20] was given by Mucke in 1996. Sikiric, Schuermann, and Vallentin dealt with finding the Voronoi diagram on a Euclidean lattice in low dimensional ([25]). Some

* Corresponding author

ISSN 0944-6532 / $ 2.50

generalizations of Voronoi diagam and Delaunay triangulations are introduced, Aurenhammer introduced power diagrams [9]. Joe considered k-dimensional Delaunay triangulations [17]. Demaret and Iske dealt with anisotropic Delaunay triangulations [13].

Many properties of Delaunay triangulations were given in [7], [11], [18], [21], [24]. Among the others, [12] states that determining the Delaunay triangulation of a finite planar point set is equivalent to determining the lower convex hull of the corresponding point set in $\mathbb{R}^3$ by a lifting projection of a polytope in 3D to the convex hull of the point set, here the lifting projection of convex polyhedra was introduced by Walkup and Wets in [28]. Motivated by the relationship between the Delaunay triangulation and the lower convex hull, in 2015, An and Giang proposed a direct method for determining the lower convex hull of a finite point set in $\mathbb{R}^3$ and determining Delaunay triangulation via the corresponding lower convex hull in $\mathbb{R}^3$ [5]. The algorithm introduced in [5] is more efficient than the previous proposed algorithms.

Since 2010 the restricted region technique of orienting curves has been used successfully for solving convex hull, shortest path and Delaunay triangulation problems (see [2], [3], [4], [6], [7], [8]).

In this paper, we present the restricted region technique of lifting projection between a polytope in 3D and a convex polygon in 2D for computing the lower convex hull in $\mathbb{R}^3$ in finding Delaunay triangulation of a finite set of planar points. In concrete, recognizing some special characteristics of this problem class, we propose a projection on planes parallel to the coordinate planes (see Propositions 4.3 and 4.4), thereby eliminating a large number of points to be calculated. We apply this technique to the algorithm introduced in [5] to get an improved version (Procedure 3). Our numerical experiments on randomly generated test instances show that the modification reduces the computation time of the original algorithm by a factor from 2.03 to 3.60 in case q is chosen as the origin $O(0, 0)$ (outside the convex hull of the point set) and from 1.70 to 2.19 in case q is the median point of P (Tables 5.1, 5.2 and Figures 6.1, 6.2).

The paper consists of several sections. Section 2 gives some geometrical notions that will be used in this paper. We also present a special case of the lifting projection: lifting a convex hull of points on a paraboloid to a convex polygon. Section 3 introduces a lower convex hull algorithm of a finite point set in $\mathbb{R}^3$. Section 4 presents the restricted region technique in lifting projection for computing the lower convex hull in a class of finding Delaunay triangulation of a finite set of planar points. Section 5 closes the paper with some numerical experiments and comparisons with the algorithm introduced in [5].

2. Preliminaries

Throughout this paper, we focus on the problem of determining the lower convex hull of a finite point set in $\mathbb{R}^3$. For convenience of the readers, in this section we review some necessary concepts.

For any points p, q in $\mathbb{R}^3$, set $[p, q] := \{(1 - \lambda)p + \lambda q : 0 \leq \lambda \leq 1\}$ and denote pq ($[p, q]$, $\mathcal{L}_{pq}$, respectively) the line (segment, the length of the segment, respectively)

through the points p and q. We assume that the finite point set in $\mathbb{R}^3$, mentioned in this paper, does not have any four coplanar points. The general case is presented in [5].

Given three points $p = (p_x, p_y, p_z)$, $q = (q_x, q_y, q_z)$, $t = (t_x, t_y, t_z)$, then the plane through the three points p, q, t, denoted by (p, q, t), has the form

$$xn_x + yn_y + zn_z = d,$$

where $\overrightarrow{n} = (n_x, n_y, n_z)$ is a vector normal and $d := q_x n_x + q_y n_y + q_z n_z$. Suppose that $n_z \neq 0$ and $w = (w_x, w_y, w_z)$.

If $w_z > w_z^* := (d - w_x n_x - w_y n_y)/n_z$ ($w_z < w_z^*$, $w_z = w_z^*$, respectively) then w is called *above* (*below, on*, respectively) the plane (p, q, t) (see Figure 2.1).

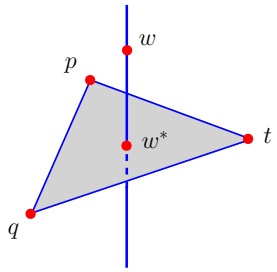

Figure 2.1: The point w is above the plane (p, q, t).

Let P be a finite point set in $\mathbb{R}^3$, the *convex hull* of P, denoted by $\mathrm{CH}(P)$, is the smallest convex set containing all points of P (in the sense of set inclusion). We are interested in the *lower convex hull* of P as defined in Definition 2.1.

Definition 2.1. (see [5, 22, 23]) A *lower facet* of P is a triangle whose three vertices belong to P such that all points of P are on or above the plane passing these vertices. *Lower convex hull* of P, denoted by $\mathrm{CH}_L(P)$, consists of all lower facets of the convex hull. We call a facet of the convex hull of P, which is not a lower facet, an *upper facet*.

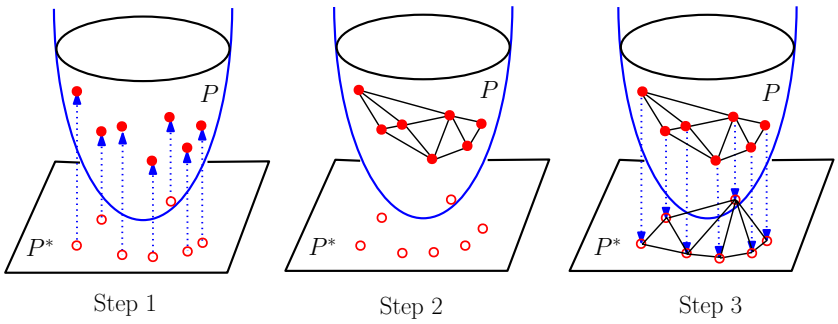

Figure 2.2: The Delaunay triangulation and the lower convex hull

The segment connecting two vertices of a lower facet is called an *edge* of the lower convex hull. An edge is shared by exactly either two lower facets or by one lower facet and one upper facet (see [5]). We now determine the Delaunay triangulation of a finite point set $P^* = \{p_1^*, p_2^*, ..., p_n^*\}$ in $\mathbb{R}^2$ as follows (due to [12]):

1. Consider $P = \{p_i(x_i, y_i, z_i) \in \mathbb{R}^3, i = 1, 2, ..., n\}$ such that $x_i = x_i^*, y_i = y_i^*$, and $z_i = (x_i^* - q_x)^2 + (y_i^* - q_y)^2$, where $q(q_x, q_y) \in \mathbb{R}^2$. Denote by μ this mapping. Thus, any point u of P^* in $\mathbb{R}^2$ is lifted onto $\mu(u)$ on a paraboloid in $\mathbb{R}^3$.
2. Construct the lower convex hull $\mathrm{CH}_L(P)$ of P.
3. Project all the lower facets of $\mathrm{CH}_L(P)$ in the direction parallel to the z-axis onto the plane Oxy (which contains the set P^*) and return the resulting Delaunay triangulation of P^*. We denote this projection as μ^*

Definition 2.2. [28] Suppose τ is an affine transformation taking a closed convex polyhedron L onto a closed convex polyhedron Q. A function τ^* taking Q into L is said to be a *lifting* of Q into L relative to τ if

(i) τ^* is a single-valued inverse of τ, that is, $\tau^*(q)$ is a point of $\tau^{-1}(q)$ for each $q \in Q$, and

(ii) $\tau^*(Q)$ is the union of closed faces of L.

Clearly, the lifting μ^* above projects the lower convex hull Q of P (a convex polytope in 3D) on the convex hull L of P^* (a convex polygon in 2D). Note that L is formed by Delaunay triangles of P^* (see Sect. 5.3.1 [22]). It follows that μ^* is a special case of the lifting projection of convex polyhedra in Definition 2.2. Similarly, the transform that embeddes power diagrams in one dimension higher of Aurenhammer [9] is also a special case of the lifting projection.

The point q is usually chosen as the origin $O(0, 0)$ (see [7], [11], [12], [21]) that is outside the convex hull L. However, in [7] and [11], the authors use the median point of P as q. The problem of finding the lower convex hull is taken from the result of the convex hull problem which can be solved by many simple algorithms [7], [11], [12], [21].

3. Lower convex hulls and a solving algorithm

To find the lower convex hull $\mathrm{CH}_L(P)$ of a finite point set P, it is usual to find the convex hull $\mathrm{CH}(P)$ of P. Then from the set of facets of $\mathrm{CH}(P)$, one will take out the lower facets of $\mathrm{CH}_L(P)$ based on the following property.

Proposition 3.1. (see [5, 22, 23]) *Let P be a finite point set in $\mathbb{R}^3$ and $a, b, p \in P$, suppose that $\overrightarrow{n} = (n_x, n_y, n_z) = \overrightarrow{ab} \times \overrightarrow{ap}$, then we have:*

(i) *if (a, b, p) is a facet of $\mathrm{CH}(P)$ and $n_z < 0$ then it is a lower facet of $\mathrm{CH}_L(P)$;*

(ii) *if $n_z \geq 0$ then (a, b, p) is not a lower facet of $\mathrm{CH}_L(P)$.*

In 2015, An and Giang proposed an algorithm to find the lower convex hull directly without finding the convex hull. By the proposition above, they showed the following result.

Proposition 3.2. (see [5]) *Let $e := [a, b]$ be an edge of $CH_L(P)$, and suppose that $\overrightarrow{n} = (n_x, n_y, n_z) = \overrightarrow{ab} \times \overrightarrow{ap}$. Then,*

(i) *if $n_z < 0$ and $n_z v_z < d - v_x n_x - v_y n_y$ then v is above the plane passing (e, p),*

(ii) *if $n_z \geq 0$ or $n_z < 0$ and $n_z v_z \geq d - v_x n_x - v_y n_y$ then (e, p) is not a lower facet,*

where $d := p_x n_x + p_y n_y + p_z n_z$.

The lower convex hull algorithm starts with finding the first edge e_1 in the set of edges $\mathcal{E}_L(P)$ of the lower convex hull of P (see [5]). The next step is to find a lower facet F of $\mathrm{CH}_L(P)$ through the edge e_1. From the edges of the lower facet F, the algorithm continues to find other facets of the lower convex until all facets of $\mathrm{CH}_L(P)$ are found. First, we discuss how to find the first edge of the lower convex hull. Let

$$P = \{p_i = (x_i, y_i, z_i) \in I\!R^3,\ i = 1, 2, \ldots, n\}, n \geq 4$$

and $P' = \{p'_1, p'_2, \ldots, p'_n\}$ with the coordinates $(0, y_i, z_i)$, $i = 1, 2, \ldots, n$ be the projection of P in the direction parallel to the x-coordinate axis onto the plane $0yz$. We view the plane $0yz$ from $x \approx +\infty$ in $\mathbb{R}^3$ and Oy axis is from right to left. We choose the point in P' as the rightmost lowest point of P' and label it v'_1.

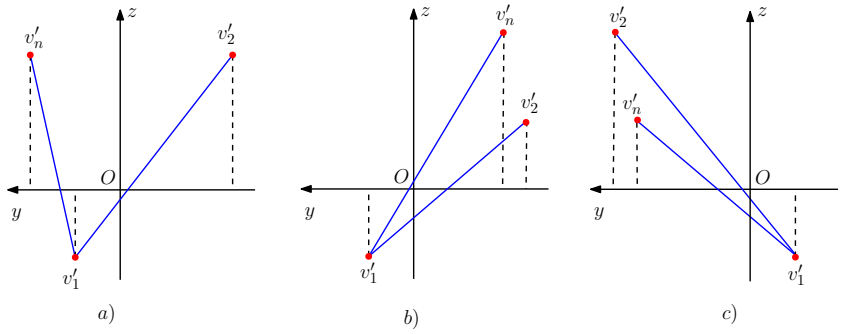

Figure 3.1: Three cases of v'_1, v'_2, and v'_n.

Procedure 1 LF(e, P) (see [5])

Input: Let $P = \{p_i = (x_i, y_i, z_i) \in \mathbb{R}^3,\ i = 1, 2, \ldots, n\}, n \geq 4$, $e := [a, b]$ is an edge of $\mathrm{CH}_L(P)$.

Output: A facet F of $\mathrm{CH}_L(P)$ through e or return e.

1: $l := 1$.
2: **while** $(p_l \in \{a, b\})$ **do**
3: set $l := l + 1$.
4: Consider (abp_l) with $p_l = (p_{lx}, p_{ly}, p_{lz})$, compute $\overrightarrow{n_l} = (n_{lx}, n_{ly}, n_{lz}) = \overrightarrow{ab} \times \overrightarrow{ap_l}$ and $d = p_{l_x} n_x + p_{l_y} n_y + p_{l_z} n_z$
5: **if** $n_{lz} < 0$
6: **for all** $(p \in P \setminus \{a, b, p_l\})$ **do**
7: **if** $(n_{l_x} p_x + n_{l_y} p_y + n_{l_z} p_z > d)$ **then goto** 9 ▷ *p is below the plane* (abp_l)
8: **return** $F = (abp_l)$
9: set $l := l + 1$
10: **if** $(l \leq n)$ **then goto** 2
11: **return** e

Suppose that P' contains at least three noncollinear points. Then, take $v'_2(v'_n$, respectively) such that all other points of P' are on or to the left (right, respectively) of $v'_1v'_2(v'_1v'_n$, respectively). In [5], the authors show that at least one of the two segments $[v_1, v_2]$ and $[v_1, v_n]$ is an edge of $\mathrm{CH}_L(P)$. Specifically, we have three cases as follows.

- If $v_{2y} \leq v_{1y} \leq v_{ny}$ or $v_{ny} \leq v_{1y} \leq v_{2y}$ then both of the two segments $[v_1, v_2]$ and $[v_1, v_n]$ are edges of $\mathrm{CH}_L(P)$ (see Figure 3.1a).
- If $v_{1y} > \max\{v_{2y}, v_{ny}\}$ then $[v_1, v_2]$ is an edge of $\mathrm{CH}_L(P)$ (see Figure 3.1b).
- If $v_{1y} < \min\{v_{2y}, v_{ny}\}$ then $[v_1, v_n]$ is an edge of $\mathrm{CH}_L(P)$ (see Figure 3.1c).

Where $v'_1 = (0, v_{1y}, v_{1z}), v'_2 = (0, v_{2y}, v_{2z})$ and $v'_n = (0, v_{ny}, v_{nz})$.

Similar to the gift-wrapping algorithm for finding convex hull of a finite point set (see [22] and [6]), the most important procedure in the lower convex hull algorithm is to find a lower facet of $\mathrm{CH}_L(P)$ through a given edge $e \in \mathcal{E}_L(P)$, which is illustrated in the Procedure 1. However, if e is a bound-edge (an edge of the lower convex hull which is shared by exactly one lower facet and one upper facet) then Procedure 1 returns e.

An algorithm for finding all the facets of $\mathrm{CH}_L(P)$ modified from the gift-wrapping algorithm is Algorithm 2. The time complexity of this algorithm in the worst case is $\mathrm{O}(n^2)$.

Algorithm 2 The lower convex hull algorithm (see [5])

Input: $P := \{p_i = (x_i, y_i, z_i) \in \mathbb{R}^3, i = 1, 2, \ldots, n\}, n \geq 4$.
Output: Set $\mathcal{Q}$ of all the faces of $\mathrm{CH}_L(P)$.

1: Find the first edge e_1 of $\mathrm{CH}_L(P)$
2: Consider three sets $\mathcal{E}_L(P) := \emptyset$, $\mathcal{E}'_L(P) := \emptyset$, $\mathcal{Q} := \emptyset$ and a queue $\mathcal{Q}' := \emptyset$. Call $\mathbf{LF}(e_1, P)$ to obtain the first lower facet F_1; all edges of F_1 except e_1 are pushed into $\mathcal{E}_L(P)$; e_1 is pushed in to $\mathcal{E}'_L(P)$ and F_1 is pushed into $\mathcal{Q}'$
3: **while** $(\mathcal{Q}' \neq \emptyset)$ **do**
4: F is extracted from the front of $\mathcal{Q}'$
5: $T :=$ edges of F
6: **for each** $(e \in T$ and $e \notin \mathcal{E}'_L(P))$ **do**
7: call $\mathbf{LF}(e, P)$
8: **if** $\mathbf{LF}(e, P)$ returns a facet F' sharing e with F **then**
9: insert into $\mathcal{E}_L(P)$ all edges $e' \neq e$ of F' not yet present in $\mathcal{E}_L(P)$ and re-
10: move all those already present in $\mathcal{E}_L(P)$ to $\mathcal{E}'_L(P)$; facet F' is pushed into $\mathcal{Q}'$
11: **else** remove e in $\mathcal{E}_L(P)$ to $\mathcal{E}'_L(P)$
12: F is pushed into $\mathcal{Q}$
13: **return** $\mathcal{Q}$

4. The restricted region technique for computing the lower convex hull in finding Delaunay triangulations

In this section we introduce a technique to speed up $\mathbf{LF}(e, P)$ of Algorithm 2 for computing the lower convex hull in finding Delaunay triangulation. Based on the special characteristics of the input point set in this problem class, we propose a restricted region technique to reduce a large number of points to be processed.

We recall that all the points in the input set P of the lower convex hull problem lie on a paraboloid $(\mathcal{P})$ given by the equation

$$z = (x - q_x)^2 + (y - q_y)^2.$$

The most important part in the lower convex hull algorithm (Procedure $\mathbf{LF}(e, P)$) is to find a lower facet of $\mathrm{CH}_L(P)$ through a given edge $e \in \mathcal{E}_L(P)$. On the other hand, we see that each $\mathbf{LF}(e, P)$ call, to determine whether a plane (a, b, p) is a lower facet, one needs to consider whether all points of P are on or above (a, b, p). Therefore, calculations used to consider the position of a point with respect to a plane are repeated many times. The number of these calculations is the main factor of time consumption by executing the algorithm. In this section, we present a technique to reduce the number of these calculations. Recall that a point $v \in P$ is above (on, below, respectively) the plane (a, b, p) (in which $a, b, p \in P$) if

$$d - n_x v_x - n_y v_y - n_z v_z > 0 (= 0, < 0, \text{ respectively}),$$

where $\overrightarrow{n} = (n_x, n_y, n_z)$ is a vector normal of (a, b, p) with $n_z < 0$ and $d = q_x n_x + q_y n_y + q_z n_z$.

Suppose that the plane (a, b, p) has the form

$$x n_x + y n_y + z n_z = d.$$

Since $a, b, p \in P$, the plane (a, b, p) always cuts the paraboloid $(\mathcal{P})$ according by a cross-section. The boundary of the cross-section has the form of an ellipse $(\mathtt{E})$. We will find the maximum and minimum values of the x, y, and z-coordinates of the points belonging to the ellipse $(\mathtt{E})$. Based on these values, we can consider the relative position of a point of P with the plane (a, b, p) in a more convenient and simple way. First, let

$$\min x_{\mathtt{E}} := \min\{x : (x, y, z) \in (\mathtt{E})\},\ \max x_{\mathtt{E}} := \max\{x : (x, y, z) \in (\mathtt{E})\}$$

$$\min y_{\mathtt{E}} := \min\{y : (x, y, z) \in (\mathtt{E})\},\ \max y_{\mathtt{E}} := \max\{y : (x, y, z) \in (\mathtt{E})\},$$

$$\min z_{\mathtt{E}} := \min\{z : (x, y, z) \in (\mathtt{E})\},\ \max z_{\mathtt{E}} := \max\{z : (x, y, z) \in (\mathtt{E})\}.$$

By Propsition 3.1, we only need to consider planes which have $n_z < 0$ to check whether they are lower facets of the lower convex hull. Let

$$\alpha = -\frac{n_x}{n_z},\quad \beta = -\frac{n_y}{n_z},\quad \gamma = \frac{d}{n_z}$$

then the plane (a, b, p) through three points a, b, p has the form

$$z = \alpha x + \beta y + \gamma.$$

The equation of $(\mathtt{E})$ is
$$\begin{cases} z = (x - q_x)^2 + (y - q_y)^2 \\ z = \alpha x + \beta y + \gamma. \end{cases} \tag{1}$$

Equation (1) can be given as a circular equation $(\mathcal{C})$ as follows

$$(x - q_x - \frac{\alpha}{2})^2 + (y - q_y - \frac{\beta}{2})^2 = r^2,$$

where $r^2 = \alpha q_x + \beta q_y + \frac{\alpha^2}{4} + \frac{\beta^2}{4} + \gamma$. The circle $(\mathcal{C})$, which is the projection of $(\mathtt{E})$ in the direction parallel to the z-axis onto the coordinate plane Oxy (see Figure 4.1), has the center at $c(q_x + \frac{\alpha}{2}, q_y + \frac{\beta}{2})$ and the radius $r = \sqrt{\alpha q_x + \beta q_y + \frac{\alpha^2}{4} + \frac{\beta^2}{4} + \gamma}$.

Based on the characteristics of circle $(\mathcal{C})$, we can immediately calculate the following values

$$\min x_{\mathtt{E}} = c_x - r,\ \max x_{\mathtt{E}} = c_x + r,\ \min y_{\mathtt{E}} = c_y - r,\ \max y_{\mathtt{E}} = c_y + r.$$

Lemma 4.1 below gives us the formula for determining $\min z_{\mathtt{E}}$ and $\max z_{\mathtt{E}}$.

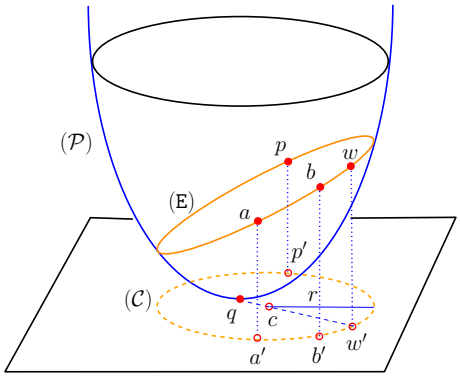

Figure 4.1: The projection of (E) onto Oxy.

Lemma 4.1. *We have* (i) $\min z_{\mathtt{E}} = (\mathcal{L}_{cq} - r)^2$, *and* (ii) $\max z_{\mathtt{E}} = (\mathcal{L}_{cq} + r)^2$, *where* $\mathcal{L}_{cq} = \sqrt{(\alpha^2 + \beta^2)/4}$ *is the length of the segment* $[c, q]$.

Proof. Suppose that $w(x, y, z)$ is a point of (E) and $w'(x, y)$ is its projection in the direction parallel to the Oz-axis onto the coordinate plane Oxy (see Figure 4.1), hence $w'(x, y) \in (\mathcal{C})$. We have

$$\mathcal{L}^2_{w'q} = (x - q_x)^2 + (y - q_y)^2 = z.$$

So we need to find the locations of the point w' on the circle $(\mathcal{C})$ such that $\mathcal{L}_{w'q}$ reaches the minimum and maximum value.

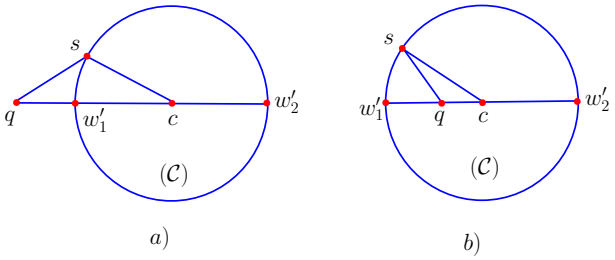

Figure 4.2: w'_1, w'_2 are the intersection points of the line cq and the circle $(\mathcal{C})$.

Suppose that w'_1 and w'_2 are the intersection points of cq and the circle $(\mathcal{C})$ as Figure 4.2. We will prove that $\mathcal{L}_{w'q}$ reaches the minimum value when $w' \equiv w'_1$ and maximum value when $w' \equiv w'_2$. Suppose that s is any point on the circle $(\mathcal{C})$. We will show

$$\mathcal{L}_{w'_1 q} \le \mathcal{L}_{sq} \le \mathcal{L}_{w'_2 q}. \tag{2}$$

We consider two cases of the point q and the circle $(\mathcal{C})$.

The first case: The point q is outside or on the circle $(\mathcal{C})$ (see Figure 4.2a). Applying triangle inequality, we have $\mathcal{L}_{cq} \leq \mathcal{L}_{cs} + \mathcal{L}_{sq}$. From $\mathcal{L}_{cq} = \mathcal{L}_{cw'_1} + \mathcal{L}_{w'_1 q}$ and $\mathcal{L}_{cs} = \mathcal{L}_{cw'_1}$, hence $\mathcal{L}_{w'_1 q} \leq \mathcal{L}_{qs}$. Equality occurs when $s \equiv w'_1$. So the first inequality of (2) is proved. The second inequality comes from the fact that $\mathcal{L}_{qs} \leq \mathcal{L}_{cq} + \mathcal{L}_{cs}$ (triangle inequality) and $\mathcal{L}_{cs} = \mathcal{L}_{cw'_2}$. Equality occurs when $s \equiv w'_2$.

The second case: The point q is inside the circle $(\mathcal{C})$ (see Figure 4.2b). In this case, to prove the first inequality of (2) we also apply the triangle inequality $\mathcal{L}_{cw'_1} = \mathcal{L}_{cs} \leq \mathcal{L}_{cq} + \mathcal{L}_{qs}$. Since $\mathcal{L}_{cw'_1} = \mathcal{L}_{cq} + \mathcal{L}_{qw'_1}$, $\mathcal{L}_{w'_1 q} \leq \mathcal{L}_{sq}$. Equality occurs when $s \equiv w'_1$. The second inequality of (2) is similarly proven in the first case.

So we conclude that $w'q$ reaches the minimum value when $w' \equiv w'_1$ and maximum value when $w' \equiv w'_2$. It follows that $\min z_{\mathtt{E}} = \min \mathcal{L}^2_{w'q} = \mathcal{L}^2_{w'_1 q} = (\mathcal{L}_{cq} - r)^2$ and $\max z_{\mathtt{E}} = \max \mathcal{L}^2_{w'q} = \mathcal{L}^2_{w'_2 q} = (\mathcal{L}_{cq} + r)^2$. □

Note that if the center of the paraboloid is outside the convex hull of the set then the first case in the proof above does not happen any more. Suppose that the projection of the points a, b, and p in the direction parallel to the Oz-axis onto the coordinate plane Oxy are a', b', and p', respectively. It is clearly that the circle $(\mathcal{C})$ contains a', b', and p' (see Figure 4.3). We have the following proposition.

Lemma 4.2. *The points of P that have the projections lying outside (inside, on, respectively) the circle $(\mathcal{C})$ are above (below, on, respectively) the plane (a, b, p).*

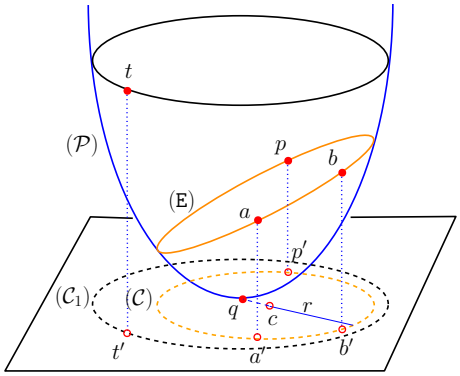

Figure 4.3: Ellipse $(\mathtt{E})$ and Circle $(\mathcal{C})$.

Proof. Consider any point $t(t_x, t_y, t_z)$ of P, whose projection $t'(t_x, t_y)$ is outside the circle $(\mathcal{C})$ (see Figure 4.3). We have $\mathcal{L}^2_{t'c} > r^2$, i.e.,

$$\left(t_x - q_x - \frac{\alpha}{2}\right)^2 + \left(t_y - q_y - \frac{\beta}{2}\right)^2 > \alpha q_x + \beta q_y + \frac{\alpha^2}{4} + \frac{\beta^2}{4} + \gamma. \tag{3}$$

Inequality (3) can be rewritten as.

$$(t_x - q_x)^2 + (t_y - q_y)^2 > \alpha t_x + \beta t_y + \gamma. \tag{4}$$

Hence t is above the plane (a, b, p) since $t_z = (t_x - q_x)^2 + (t_y - q_y)^2$. The same proof works for the other cases. □

The following propositions help us quickly identify a point lying above, below or on a plane based on coordinate comparisons.

Proposition 4.3. *If a point $t(t_x, t_y, t_z) \in P$ satisfies one of the following conditions then t is above the plane (a, b, p):*

$$\text{(i) } t_x < \min x_{\mathtt{E}}, \quad \text{(ii) } t_x > \max x_{\mathtt{E}}, \quad \text{(iii) } t_y < \min y_{\mathtt{E}},$$
$$\text{(iv) } t_y > \max y_{\mathtt{E}}, \quad \text{(v) } t_z > \max z_{\mathtt{E}}.$$

Proof. (i) We first prove that if $t_x < \min x_{\mathtt{E}}$ then the projection $t'(t_x, t_y)$ of t is outside the circle $(\mathcal{C})$. Indeed, we have $t_x < \min x_{\mathtt{E}} = c_x - r$, i.e., $c_x - t_x > r$. On the other hand, we have

$$\mathcal{L}^2_{t'c} = (c_x - t_x)^2 + (c_y - t_y)^2 > r^2 + (c_y - t_y)^2 > r^2,$$

i.e., t' is outside the circle $(\mathcal{C})$. According to Lemma 4.2, we conclude that t is above the plane (a, b, p). Similar arguments apply to the cases (ii)–(iv).

(v) Suppose that $t(t_x, t_y, t_z) \in P$ satisfies

$$t_z = (t_x - q_x)^2 + (t_y - q_y)^2 > \max z_{\mathtt{E}} = (\mathcal{L}_{cq} + r)^2.$$

So the projection $t'(t_x, t_y)$ of $t(t_x, t_y, t_z)$ lies on the circle $(\mathcal{C}_1)$ which has the center $q(q_x, q_y)$ and the radius $r_1 > \mathcal{L}_{cq} + r$, i.e., $r_1 - r > \mathcal{L}_{cq}$. From this inequality it follows that the circle $(\mathcal{C}_1)$ is outside the circle $(\mathcal{C})$. Since the point $t' \in (\mathcal{C}_1)$, t' is outside the circle $(\mathcal{C})$. According to Lemma 4.2, the point t is above the plane (a, b, p). □

Proposition 4.4. *If a point $t(t_x, t_y, t_z) \in P$ satisfies $t_z < \min z_{\mathtt{E}}$ and $\mathcal{L}_{cq} < r$, then t is below the plane (a, b, p).*

Proof. Suppose that $t(t_x, t_y, t_z) \in P$ satisfies

$$t_z = (t_x - q_x)^2 + (t_y - q_y)^2 < \min z_{\mathtt{E}} = (\mathcal{L}_{cq} - r)^2.$$

So the projection $t'(t_x, t_y)$ of $t(t_x, t_y, t_z)$ lies on the circle $(\mathcal{C}_2)$ which has the center $q(q_x, q_y)$ and the radius $r_2 < |\mathcal{L}_{cq} - r|$. Since $\mathcal{L}_{cq} < r$, $r_2 < r - \mathcal{L}_{cq}$ or $\mathcal{L}_{cq} < r - r_2$. This inequation implies that the circle $(\mathcal{C}_2)$ is inside the circle $(\mathcal{C})$. Since the point $t' \in (\mathcal{C}_2)$, t' is inside the circle $(\mathcal{C})$. According to Lemma 4.2, we conclude that the point t is below the plane (a, b, p). □

Based on Proposition 4.3 and Proposition 4.4, we improve Procedure $\mathbf{LF}(e, P)$ to get a new procedure called $\mathbf{LFRes}(e, P)$. Replacing Procedure $\mathbf{LF}(e, P)$ by $\mathbf{LFRes}(e, P)$ in Algorithm 2 we obtain a new version of Algorithm 2. Our new version also has the same $O(n^2)$ worst case complexity as the original one.

5. Implementation

Recall that our problem is to find the lower convex hull of a point set P, where all the points of P are "evenly spread" on the surface of the paraboloid $(\mathcal{P})$. Consider both cases of choosing the vertex q of the paraboloid P, namely as the origin or the median point. We call the former as case 1 and the latter as case 2. The algorithms are implemented in C and run on PC Core i5 1.8 GHz 3M with 8 GB RAM.

Procedure 3 LFRes(e, P)

Input: Let $P = \{p_i = (x_i, y_i, z_i) \in \mathbb{R}^3,\ i = 1, \ldots, n\}, n \geq 4$, $e := [a, b]$ is a lower edge of $\mathrm{CH}_L(P)$.

Output: A facet F of $\mathrm{CH}_L(P)$ through e or return e.

1: $l := 1$.
2: **while** $(p_l \in \{a, b\})$ **do**
3: set $l := l + 1$
4: Consider (abp_l) with $p_l = (p_{lx}, p_{ly}, p_{lz})$, compute $\overrightarrow{n_l} = (n_{lx}, n_{ly}, n_{lz}) = \overrightarrow{ab} \times \overrightarrow{ap_l}$ and $d = p_{l_x} n_x + p_{l_y} n_y + p_{l_z} n_z$
5: **if** $n_{lz} < 0$ **then**
6: compute the coefficients $\alpha = -\frac{n_{lx}}{n_{lz}},\ \beta = -\frac{n_{ly}}{n_{lz}},\ \gamma = \frac{d}{n_{l_z}}, \mathcal{L}_{cq} = \frac{1}{2}\sqrt{\alpha^2 + \beta^2}$, and $r = \sqrt{\gamma + \alpha q_x + \beta q_y + \mathcal{L}_{cq}^2}$
7: compute $\min x_{\mathtt{E}} = q_x + \frac{\alpha}{2} - r,\ \max x_{\mathtt{E}} = q_x + \frac{\alpha}{2} + r,\ \min y_{\mathtt{E}} = q_y + \frac{\beta}{2} - r,$ $\max y_{\mathtt{E}} = q_y + \frac{\beta}{2} + r,\ \min z_{\mathtt{E}} = (\mathcal{L}_{cq} - r)^2,\ \max z_{\mathtt{E}} = (\mathcal{L}_{cq} + r)^2$
8: **for all** $(p \in P \setminus \{a, b, p_l\})$ **do**
9: **if** $(\mathcal{L}_{cq} < r$ and $p_z < \min z_{\mathtt{E}})$ **then goto** 12 ▷ *Using Proposition 4.4*
10: **else if** $(\min x_{\mathtt{E}} \leq p_x \leq \max x_{\mathtt{E}}$ and $\min y_{\mathtt{E}} \leq p_y \leq \max y_{\mathtt{E}}$ and $p_z \leq \max z_{\mathtt{E}}$ and $n_{l_x} p_x + n_{l_y} p_y + n_{l_z} p_z > d)$ **then goto** 12 ▷ *Using Proposition 4.3*
11: **return** $F = (abp_l)$
12: set $l := l + 1$
13: **if** $(l \leq n)$ **then goto** 2
14: **return** e

The input data is the set of points

$$P = \{p_i = (x_i, y_i, z_i) : x_i, y_i, z_i \in \mathbb{R},\ z_i = (x_i - q_x)^2 + (y_i - q_y)^2,\ i = 1, 2, \ldots, n\} \subset \mathbb{R}^3,$$

where the set $\{(x_i, y_i), x_i, y_i \in \mathbb{R},\ i = 1, 2, \ldots, n\}$ is randomly positioned in the interior of a square of size $[0, 200] \times [0, 200]$ (on the uniform distribution). Figures 5.1 and 5.2 illustrate the data of 1000 points observed from two different directions in case 1 and case 2, respectively.

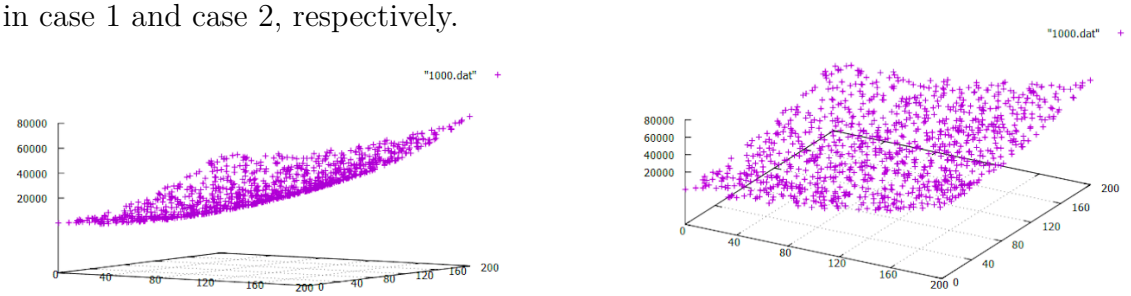

Figure 5.1: The data 1000 points in case 1.

Table 5.1 describes the actual running times (in seconds) of the algorithms in case 1 and Table 5.2 describes those in case 2.

Tables 5.1, 5.2 and Figures 6.1, 6.2 show that the new version of Algorithm 2 (using **LFRes**(e, P)) runs significantly faster than the Algorithm 2 (using **LF**(e, P)) in [5]. The speedup ratio of the new algorithm compared to the original one is from 2.03 to 3.60 in case 1 and from 1.70 to 2.19 in case 2. However, the computation time in case 2 is better.

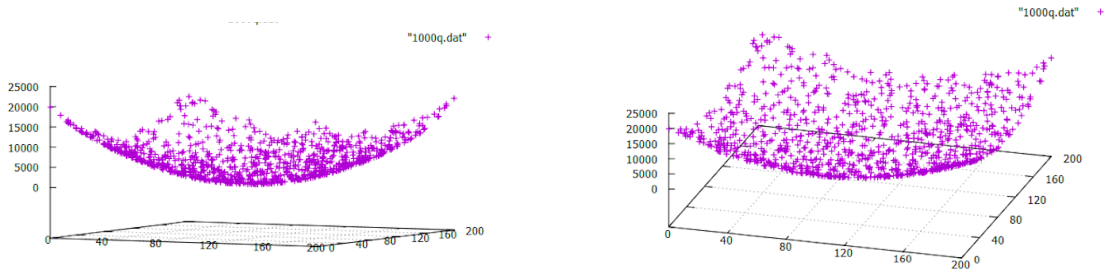

Figure 5.2: The data 1000 points with the vertex of $(\mathcal{P})$ at the median point.

Table 5.1: The actual running times (in seconds) of the algorithms in case 1.

Input	Algorithm 2 (using **LF**(e, P))	New version of Algorithm 2 (using **LFRes**(e, P))	Speedup ratio
1,000	0.162	0.078	2.09
2,000	0.672	0.301	2.23
3,000	1.525	0.666	2.29
5,000	4.534	1.790	2.53
7,000	10.843	3.014	3.60
8,000	13.110	5.100	2.57
10,000	22.560	9.740	2.32
15,000	70.237	34.567	2.03
20,000	102.257	49.339	2.07

Table 5.2: The actual running times (in seconds) of the algorithms in case 2.

Input	Median point	Algorithm 2 (using **LF**(e, P))	New version of Algorithm 2 (using **LFRes**(e, P))	Speedup ratio
1,000	(100.82, 98.73)	0.088	0.052	1.70
2,000	(99.58, 98.58)	0.368	0.210	1.75
3,000	(99.59, 97.56)	0.376	0.235	1.60
5,000	(98.30, 100.60)	2.341	1.067	2.19
7,000	(98.65, 99.88)	4.611	2.144	2.15
8,000	(100.24, 101.09)	6.287	3.023	2.08
10,000	(101.39, 98.41)	11.789	6.015	1.96
15,000	(98.63, 100.53)	29.132	15.513	1.88
20,000	(100.07, 99.86)	53.417	31.183	1.71

6. Concluding remarks

To find the Delaunay triangulation of disks in the plane, we can establish a relation between the lower convex hull of disks lying on a paraboloid in $\mathbb{R}^3$ and the Delaunay triangulation of disks in the plane then use the restricted region technique for finding such a lower convex hull, here the convex hull of disks was presented in [19]. Similar to the lifting projection on convex polyhedra (Definition 2.2), the technique can be used in higher dimensions. It can be also used for finding the anisotropic Delaunay triangulations given in [13] and for finding the power diagrams given in [9]. These will be the subject of another paper.

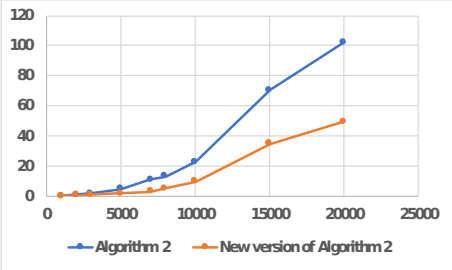

Figure 6.1: The running time comparison of Algorithm 2 and the new version in case 1.

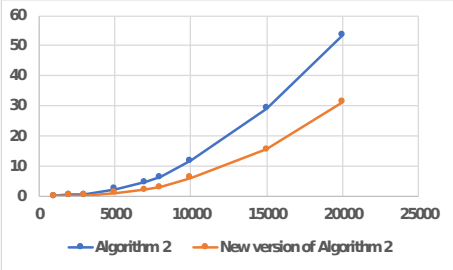

Figure 6.2: The running time comparison of Algorithm 2 and the new version in case 2.

Acknowledgments. The first author would like to thank the Ho Chi Minh City University of Technology (HCMUT), VNU-HCM for the support of time and facilities for this study.

References

[1] S. G. Akl, G. T. Toussaint: *Efficient convex hull algorithms for pattern recognition applications*, in: *International Joint Conference on Pattern Recognition*, Kyoto, Japan (1978) 483–487.

[2] P. T. An: *Method of orienting curves for determining the convex hull of a finite set of points in the plane*, Optimization 59/2 (2010) 175–179.

[3] P. T. An: *Optimization Approaches for Computational Geometry*, Publishing House for Science and Technology, Vietnam Academy of Science and Technology, Hanoi (2017).

[4] P. T. An: *Finding shortest paths in a sequence of triangles in 3D by the method of orienting curves*, Optimization 67 (2018) 159–177.

[5] P. T. An, D. T. Giang: *A direct method for determining the lower convex hull of a finite point set in 3D*, in: *Advanced Computational Methods for Knowledge Engineering*, Proc. 3rd Int. Conf. Computer Science, Advances Intell. Syst. Computing 358 (2015) 15–26.

[6] P. T. An, N.-D. Hoang, N. K. Linh: *An efficient improvement of gift-wrapping algorithm for computing the convex hull of a finite set of points in* $\mathbb{R}^n$, Numer. Algorithms 85 (2020) 1499–1518.

[7] P. T. An, L. H. Trang: *A parallel algorithm based on convexity for the computing of Delaunay tessellation*, Numer. Algorithm 59 (2012) 347–357.

[8] P. T. An, L. H. Trang: *An efficient convex hull algorithm for finite point sets in 3D based on the Method of Orienting Curves*, Optimization 62 (2013) 975–988.

[9] F. Aurenhammer: *Power diagrams: properties, algorithms and applications*, SIAM J. Computing 16 (1987) 78–96.

[10] P. Bhaniramka, R. Wenger, R. Crawfis: *Isosurface construction in any dimension using convex hulls*, IEEE Trans. Visualization Computer Graphics 10/2 (2004) 130–141.

[11] G. E. Blelloch, J. C. Hardwick, G. L. Miller, D. Talmor: *Design and implementation of a practical parallel Delaunay algorithm*, Algorithmica 24 (1999) 243–269.

[12] K. Q. Brown: *Voronoi diagrams from convex hulls*, Inf. Proc. Letters 9/5 (1979) 223–228.

[13] L. Demaret, A. Iske: *Optimal N-term approximation by linear splines over anisotropic Delaunay triangulations*, Math. Computation 84 (2015) 1241–1264.

[14] S. Fortune: *A sweepline algorithm for the Voronoi diagrams*, Algorithmica 2 (1987) 153–174.

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

[16] L. J. Guibas, J. Stolfi: *Primitives for the manipulation of general subdivisions and the computation of Voronoi diagrams*, ACM Trans. Graphics 4 (1985) 74–123.

[17] B. Joe: *Construction of k-dimensional Delaunay triangulations using local transformations*, SIAM J. Sci. Computing 14 (1993) 1415–1436.

[18] M. Koiso, B. Palmer: *Rolling construction for anisotropic Delaunay surfaces*, Pacific J. Math. 267/1 (2008) 345–378.

[19] N. K. Linh, C. Song, J. Ryu, P. T. An, N.-D. Hoang, D.-S. Kim: *QuickhullDisk: A faster convex hull algorithm for disks*, Appl. Math. Computation 363 (2019) 124626.

[20] E. P. Mucke, I. Saias: *Fast randomized point location without preprocessing in two- and three-dimensional Delaunay triangulations*, CompGeom'96, Philadelphia (1996) 274–283.

[21] A. Okabe, B. Boots, K. Sugihara: *Spatial Tessellations: Concepts and Applications of Voronoi Diagrams*, 2nd ed., John Wiley & Sons, Hoboken (2000).

[22] J. O'Rourke: *Computational Geometry in C*, 2nd ed., Cambridge University Press, Cambridge (1998).

[23] F. P. Preparata, M. I. Shamos: *Computational Geometry*, 2nd ed., Springer, New York (1985).

[24] A. Przeworski: *Delaunay cells for arrangements of flats in hyperbolic space*, Pacific J. Math. 258/1 (2012) 223–256.

[25] M. D. Sikiric, A. Schuermann, F. Vallentin: *Complexity and algorithms for computing Voronoi cells of lattices*, Math. Computation 267 (2009) 1713–1731.

[26] N. M. Sirakov, P. A. Mlsna: *Search space partitioning using convex hull and concavity features for fast medical image retrieval*, in: *Proc. 2nd IEEE Int. Symp. Biomedical Imaging: Nano to Macro*, Arlington (2004) 796–799.

[27] A. Stepanova, J. M. G. Smith: *Modeling wildfire propagation with Delaunay triangulation and shortest path algorithms*, Eur. J. Oper. Res. 218/3 (2012) 775–788.

[28] D. Walkup, R. J.-B. Wets: *Lifting projections of convex polyhedra*, Pacific J. Math. 28/2 (1969) 465–475.

[29] B. Yuan, C. L. Tan: *Convex hull based skew estimation*, Pattern Recognition 40/2 (2007) 456–475.