CONVEX HULLS IN 3D

Vera Sacristán

Discrete and Algorithmic Geometry
Facultat de Matemàtiques i Estadística
Universitat Politècnica de Catalunya

STORING A CONVEX POLYHEDRON

CONVEX HULLS IN 3D

STORING A CONVEX POLYHEDRON

Planarity: Any convex polyhedron is a planar graph.

STORING A CONVEX POLYHEDRON

Planarity: Any convex polyhedron is a planar graph.

Proof: Consider a convex polyhedron P. Let p be a point in the interior of P and S be a sphere containing P. Due to its convexity, the central projection of P from p onto S is a graph G_1 isomorphic to P. Let q be a point in S interior to a face of G_1 and π be the plane tangent to S in the point diametrically opposed to q. The stereographic projection of S from q onto π maps G_1 into a graph G_2 which is isomorphic to G_1 and plane. Therefore, P is a planar graph.

STORING A CONVEX POLYHEDRON

Planarity: Any convex polyhedron is a planar graph.

Proof: Consider a convex polyhedron P. Let p be a point in the interior of P and S be a sphere containing P. Due to its convexity, the central projection of P from p onto S is a graph G_1 isomorphic to P. Let q be a point in S interior to a face of G_1 and π be the plane tangent to S in the point diametrically opposed to q. The stereographic projection of S from q onto π maps G_1 into a graph G_2 which is isomorphic to G_1 and plane. Therefore, P is a planar graph.

Euler's relation: If G is a connected planar graph with v vertices, e edges, and f faces, $v + f = e + 2$.

STORING A CONVEX POLYHEDRON

Planarity: Any convex polyhedron is a planar graph.

Proof: Consider a convex polyhedron P. Let p be a point in the interior of P and S be a sphere containing P. Due to its convexity, the central projection of P from p onto S is a graph G_1 isomorphic to P. Let q be a point in S interior to a face of G_1 and π be the plane tangent to S in the point diametrically opposed to q. The stereographic projection of S from q onto π maps G_1 into a graph G_2 which is isomorphic to G_1 and plane. Therefore, P is a planar graph.

Euler's relation: If G is a connected planar graph with v vertices, e edges, and f faces, $v + f = e + 2$.

Proof: If G is a tree, we proceed by induction on v. If $v = 1$, then $e = 0$, $f = 1$ and the relation holds. If G has $v > 1$ vertices, let G' be the result of eliminating a leaf from G, together with its incident edge. By inductive hypothesis: $v + f = (v' + 1) + f' = (e' + 1) + 2 = e + 2$.
For arbitrary graphs, we apply induction on e. If G contains a cycle, let G' be the result of removing an edge from a cycle of G. By inductive hypothesis, $v + f = v' + (f' + 1) = (e' + 1) + 2 = e + 2$.

CONVEX HULLS IN 3D

STORING A CONVEX POLYHEDRON

Planarity: Any convex polyhedron is a planar graph.

Proof: Consider a convex polyhedron P. Let p be a point in the interior of P and S be a sphere containing P. Due to its convexity, the central projection of P from p onto S is a graph G_1 isomorphic to P. Let q be a point in S interior to a face of G_1 and π be the plane tangent to S in the point diametrically opposed to q. The stereographic projection of S from q onto π maps G_1 into a graph G_2 which is isomorphic to G_1 and plane. Therefore, P is a planar graph.

Euler's relation: If G is a connected planar graph with v vertices, e edges, and f faces, $v + f = e + 2$.

Proof: If G is a tree, we proceed by induction on v. If $v = 1$, then $e = 0$, $f = 1$ and the relation holds. If G has $v > 1$ vertices, let G' be the result of eliminating a leaf from G, together with its incident edge. By inductive hypothesis: $v + f = (v' + 1) + f' = (e' + 1) + 2 = e + 2$.
For arbitrary graphs, we apply induction on e. If G contains a cycle, let G' be the result of removing an edge from a cycle of G. By inductive hypothesis, $v + f = v' + (f' + 1) = (e' + 1) + 2 = e + 2$.

Complexity: Any connected planar graph with n vertices has at most $3n - 6$ vertices and $2n - 4$ faces.

CONVEX HULLS IN 3D

STORING A CONVEX POLYHEDRON

Planarity: Any convex polyhedron is a planar graph.

Proof: Consider a convex polyhedron P. Let p be a point in the interior of P and S be a sphere containing P. Due to its convexity, the central projection of P from p onto S is a graph G_1 isomorphic to P. Let q be a point in S interior to a face of G_1 and π be the plane tangent to S in the point diametrically opposed to q. The stereographic projection of S from q onto π maps G_1 into a graph G_2 which is isomorphic to G_1 and plane. Therefore, P is a planar graph.

Euler's relation: If G is a connected planar graph with v vertices, e edges, and f faces, $v + f = e + 2$.

Proof: If G is a tree, we proceed by induction on v. If $v = 1$, then $e = 0$, $f = 1$ and the relation holds. If G has $v > 1$ vertices, let G' be the result of eliminating a leaf from G, together with its incident edge. By inductive hypothesis: $v + f = (v' + 1) + f' = (e' + 1) + 2 = e + 2$.
For arbitrary graphs, we apply induction on e. If G contains a cycle, let G' be the result of removing an edge from a cycle of G. By inductive hypothesis, $v + f = v' + (f' + 1) = (e' + 1) + 2 = e + 2$.

Complexity: Any connected planar graph with n vertices has at most $3n - 6$ vertices and $2n - 4$ faces.

Proof: The sum of the complexities of all faces is $2e$, and each face has at least 3 edges. Therefore: $2e \geq 3f$. Plugging this inequation into Euler's relation: $\frac{3}{2}f + 2 \leq e + 2 = v + f \leq v + \frac{2}{3}e$, from where we obtain the desired result.

CONVEX HULLS IN 3D

STORING A CONVEX POLYHEDRON

Planarity: Any convex polyhedron is a planar graph.

Proof: Consider a convex polyhedron P. Let p be a point in the interior of P and S be a sphere containing P. Due to its convexity, the central projection of P from p onto S is a graph G_1 isomorphic to P. Let q be a point in S interior to a face of G_1 and π be the plane tangent to S in the point diametrically opposed to q. The stereographic projection of S from q onto π maps G_1 into a graph G_2 which is isomorphic to G_1 and plane. Therefore, P is a planar graph.

Euler's relation: If G is a connected planar graph with v vertices, e edges, and f faces, $v + f = e + 2$.

Proof: If G is a tree, we proceed by induction on v. If $v = 1$, then $e = 0$, $f = 1$ and the relation holds. If G has $v > 1$ vertices, let G' be the result of eliminating a leaf from G, together with its incident edge. By inductive hypothesis: $v + f = (v' + 1) + f' = (e' + 1) + 2 = e + 2$.
For arbitrary graphs, we apply induction on e. If G contains a cycle, let G' be the result of removing an edge from a cycle of G. By inductive hypothesis, $v + f = v' + (f' + 1) = (e' + 1) + 2 = e + 2$.

Complexity: Any connected planar graph with n vertices has at most $3n - 6$ vertices and $2n - 4$ faces.

Proof: The sum of the complexities of all faces is $2e$, and each face has at least 3 edges. Therefore: $2e \geq 3f$. Plugging this inequation into Euler's relation: $\frac{3}{2}f + 2 \leq e + 2 = v + f \leq v + \frac{2}{3}e$, from where we obtain the desired result.

Corollary: The convex hull of n points in E^3 is a convex polyhedron wich can be stored in a DCEL using $O(n)$ space.

COMPUTING A CONVEX HULL IN 3D

Algorithm

Input: $p_1, \ldots, p_n \in \mathbb{R}^3$
Output: $ch(p_1, \ldots, p_n)$

1. Initialization
   Sort $p_1, \ldots, p_n$ by abscissa.

2. Division
   Partition set $P = \{p_1, \ldots, p_n\}$ into two equally sized subsets P_1 and P_2 by means of a vertical plane h_0.

3. Recursion
   Compute $C_1 = ch(P_1)$ and $C_2 = ch(P_2)$.

4. Merging
   Compute $C = ch(C_1 \cup C_2)$.

CONVEX HULLS IN 3D

COMPUTING A CONVEX HULL IN 3D

Algorithm

Input: $p_1, \ldots, p_n \in \mathbb{R}^3$
Output: $ch(p_1, \ldots, p_n)$

1. Initialization
   Sort $p_1, \ldots, p_n$ by abscissa.

2. Division
   Partition set $P = \{p_1, \ldots, p_n\}$ into two equally sized subsets P_1 and P_2 by means of a vertical plane h_0.

3. Recursion
   Compute $C_1 = ch(P_1)$ and $C_2 = ch(P_2)$.

4. Merging
   Compute $C = ch(C_1 \cup C_2)$.

Theorem: The algorithm computes the convex hull of n points in E^3 in $O(n \log n)$ time and $O(n)$ space. These bounds are optimal.

CONVEX HULLS IN 3D

COMPUTING A CONVEX HULL IN 3D

Algorithm

Input: $p_1, \dots, p_n \in \mathbb{R}^3$
Output: $ch(p_1, \dots, p_n)$

1. Initialization
   Sort $p_1, \dots, p_n$ by abscissa.

2. Division
   Partition set $P = \{p_1, \dots, p_n\}$ into two equally sized subsets P_1 and P_2 by means of a vertical plane h_0.

3. Recursion
   Compute $C_1 = ch(P_1)$ and $C_2 = ch(P_2)$.

4. Merging
   Compute $C = ch(C_1 \cup C_2)$. $O(n)$

Theorem: The algorithm computes the convex hull of n points in E^3 in $O(n \log n)$ time and $O(n)$ space. These bounds are optimal.

COMPUTING A CONVEX HULL IN 3D

Algorithm

Input: $p_1, \ldots, p_n \in \mathbb{R}^3$
Output: $ch(p_1, \ldots, p_n)$

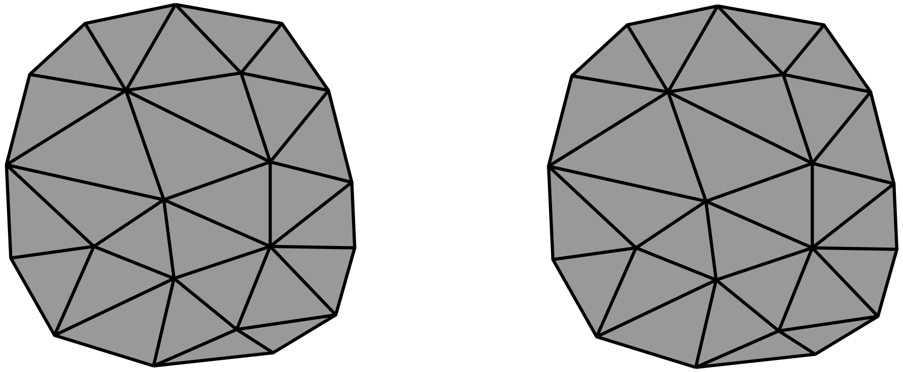
1. Initialization
   Sort $p_1, \ldots, p_n$ by abscissa.

2. Division
   Partition set $P = \{p_1, \ldots, p_n\}$ into two equally sized subsets P_1 and P_2 by means of a vertical plane h_0.

3. Recursion
   Compute $C_1 = ch(P_1)$ and $C_2 = ch(P_2)$.

4. Merging
   Compute $C = ch(C_1 \cup C_2)$.

$O(n)$



Theorem: The algorithm computes the convex hull of n points in E^3 in $O(n \log n)$ time and $O(n)$ space. These bounds are optimal.

CONVEX HULLS IN 3D

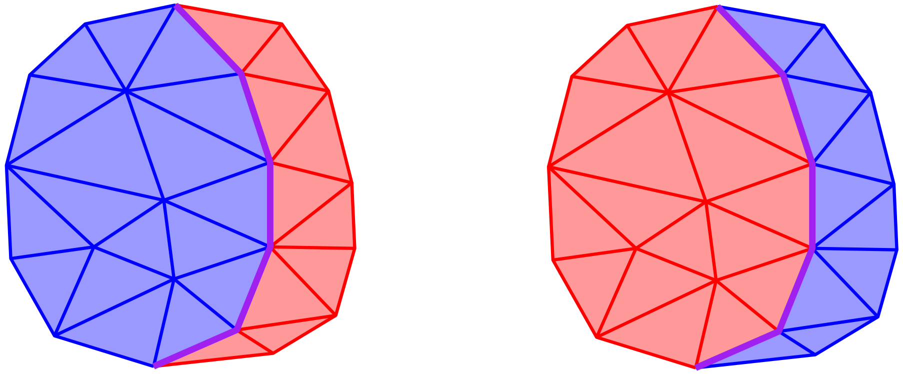
COMPUTING A CONVEX HULL IN 3D

Algorithm

Input: $p_1, \ldots, p_n \in \mathbb{R}^3$
Output: $ch(p_1, \ldots, p_n)$

1. Initialization
   Sort $p_1, \ldots, p_n$ by abscissa.
2. Division
   Partition set $P = \{p_1, \ldots, p_n\}$ into two equally sized subsets P_1 and P_2 by means of a vertical plane h_0.
3. Recursion
   Compute $C_1 = ch(P_1)$ and $C_2 = ch(P_2)$.
4. Merging
   Compute $C = ch(C_1 \cup C_2)$.

$O(n)$



Theorem: The algorithm computes the convex hull of n points in E^3 in $O(n \log n)$ time and $O(n)$ space. These bounds are optimal.

CONVEX HULLS IN 3D

COMPUTING A CONVEX HULL IN 3D

Algorithm

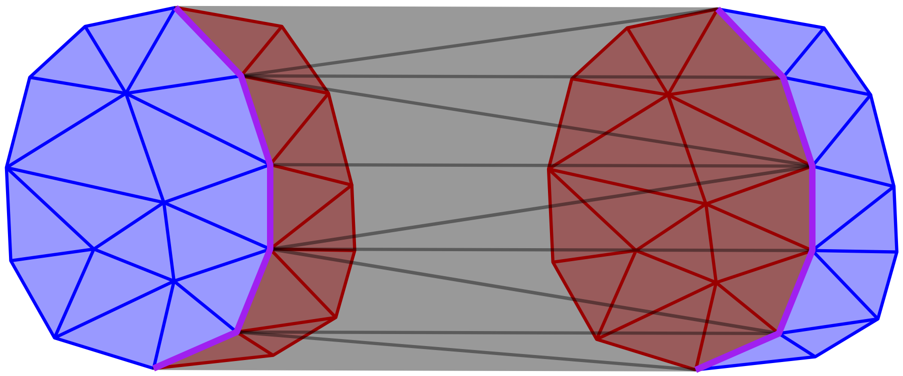
Input: $p_1, \ldots, p_n \in \mathbb{R}^3$
Output: $ch(p_1, \ldots, p_n)$

1. Initialization
   Sort $p_1, \ldots, p_n$ by abscissa.

2. Division
   Partition set $P = \{p_1, \ldots, p_n\}$ into two equally sized subsets P_1 and P_2 by means of a vertical plane h_0.

3. Recursion
   Compute $C_1 = ch(P_1)$ and $C_2 = ch(P_2)$.

4. Merging
   Compute $C = ch(C_1 \cup C_2)$. $O(n)$

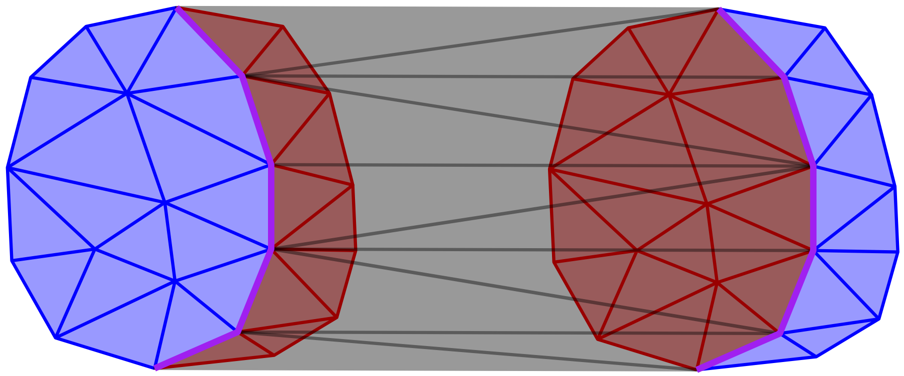


Theorem: The algorithm computes the convex hull of n points in E^3 in $O(n \log n)$ time and $O(n)$ space. These bounds are optimal.

COMPUTING A CONVEX HULL IN 3D

Notation

- Faces of C_1 and C_2 that appear in C are called blue.
- Faces of C_1 and C_2 that do not appear in C are called red.
- Edges and vertices are colored with the intersection color (blue, red and purple) of their incident faces.



CONVEX HULLS IN 3D

COMPUTING A CONVEX HULL IN 3D

Notation

- Faces of C_1 and C_2 that appear in C are called blue.
- Faces of C_1 and C_2 that do not appear in C are called red.
- Edges and vertices are colored with the intersection color (blue, red and purple) of their incident faces.

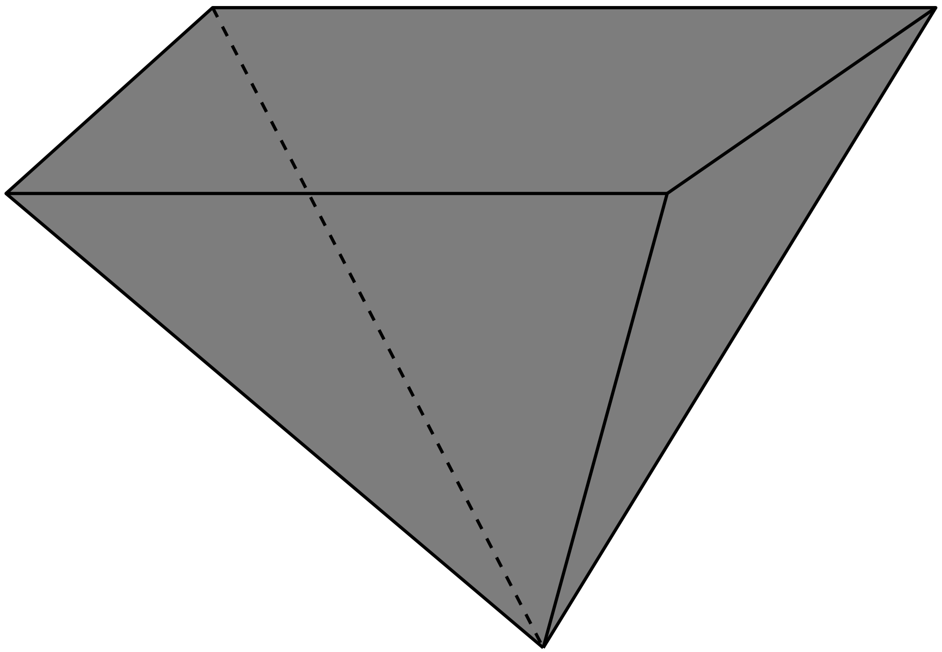
Be careful!

COMPUTING A CONVEX HULL IN 3D

Notation

- Faces of C_1 and C_2 that appear in C are called blue.
- Faces of C_1 and C_2 that do not appear in C are called red.
- Edges and vertices are colored with the intersection color (blue, red and purple) of their incident faces.

Be careful!

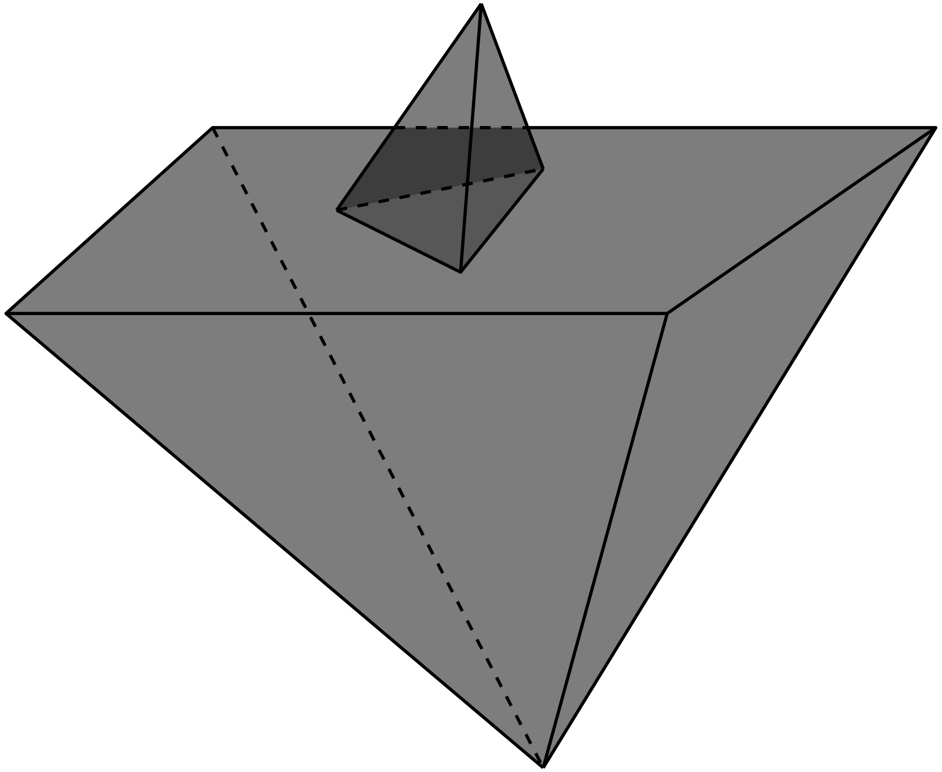


COMPUTING A CONVEX HULL IN 3D

Notation

- Faces of C_1 and C_2 that appear in C are called blue.
- Faces of C_1 and C_2 that do not appear in C are called red.
- Edges and vertices are colored with the intersection color (blue, red and purple) of their incident faces.

Be careful!



CONVEX HULLS IN 3D

COMPUTING A CONVEX HULL IN 3D

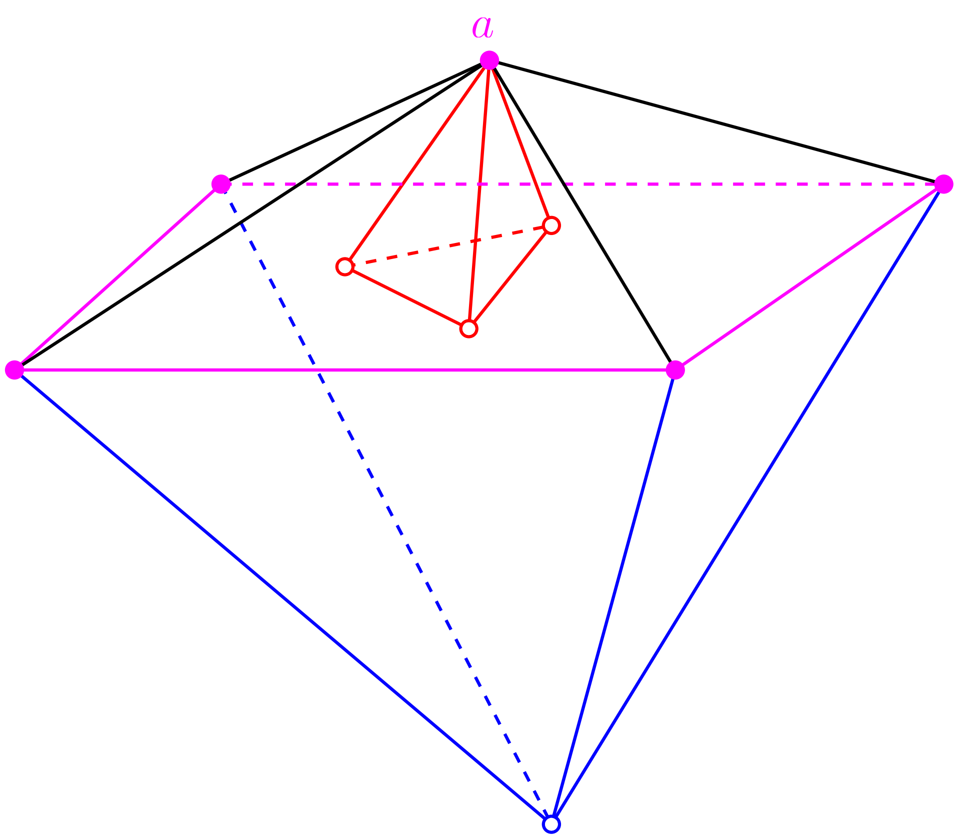
Notation

- Faces of C_1 and C_2 that appear in C are called blue.
- Faces of C_1 and C_2 that do not appear in C are called red.
- Edges and vertices are colored with the intersection color (blue, red and purple) of their incident faces.

Be careful!

Vertex a is not incident to any purple edge.

Purple is not connected.



COMPUTING A CONVEX HULL IN 3D

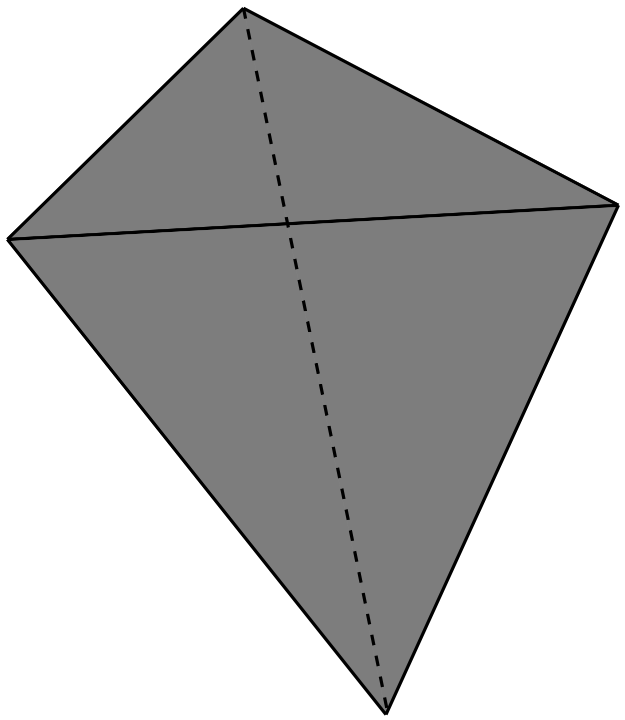
Notation

- Faces of C_1 and C_2 that appear in C are called blue.
- Faces of C_1 and C_2 that do not appear in C are called red.
- Edges and vertices are colored with the intersection color (blue, red and purple) of their incident faces.

Be careful!

Vertex a is not incident to any purple edge.

Purple is not connected.

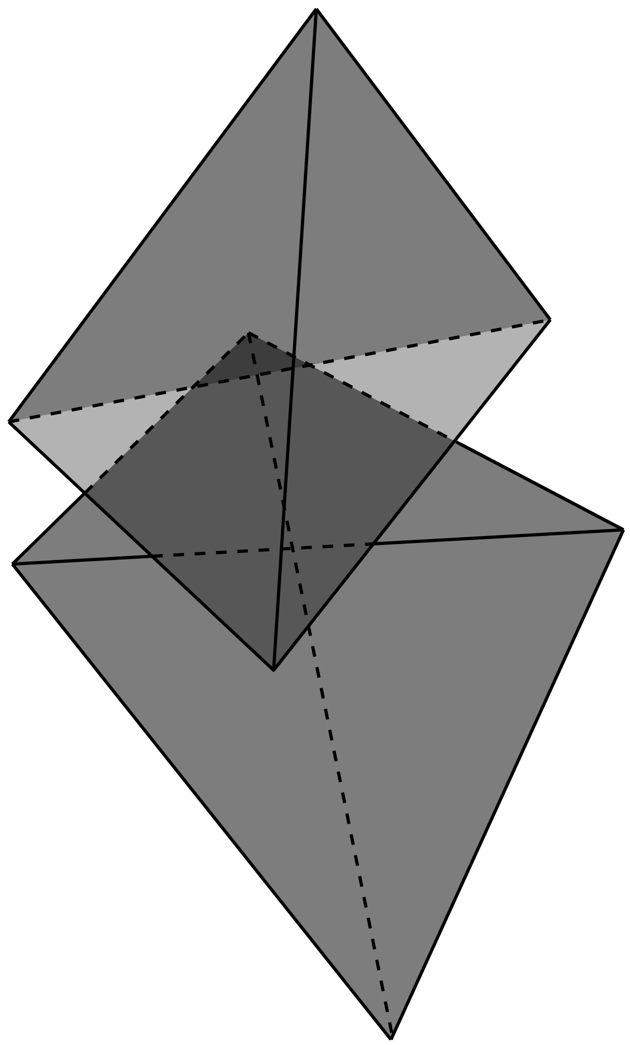


CONVEX HULLS IN 3D

COMPUTING A CONVEX HULL IN 3D

Notation

- Faces of C_1 and C_2 that appear in C are called blue.
- Faces of C_1 and C_2 that do not appear in C are called red.
- Edges and vertices are colored with the intersection color (blue, red and purple) of their incident faces.



Be careful!

Vertex a is not incident to any purple edge.

Purple is not connected.

CONVEX HULLS IN 3D

COMPUTING A CONVEX HULL IN 3D

Notation

- Faces of C_1 and C_2 that appear in C are called blue.
- Faces of C_1 and C_2 that do not appear in C are called red.
- Edges and vertices are colored with the intersection color (blue, red and purple) of their incident faces.

Be careful!

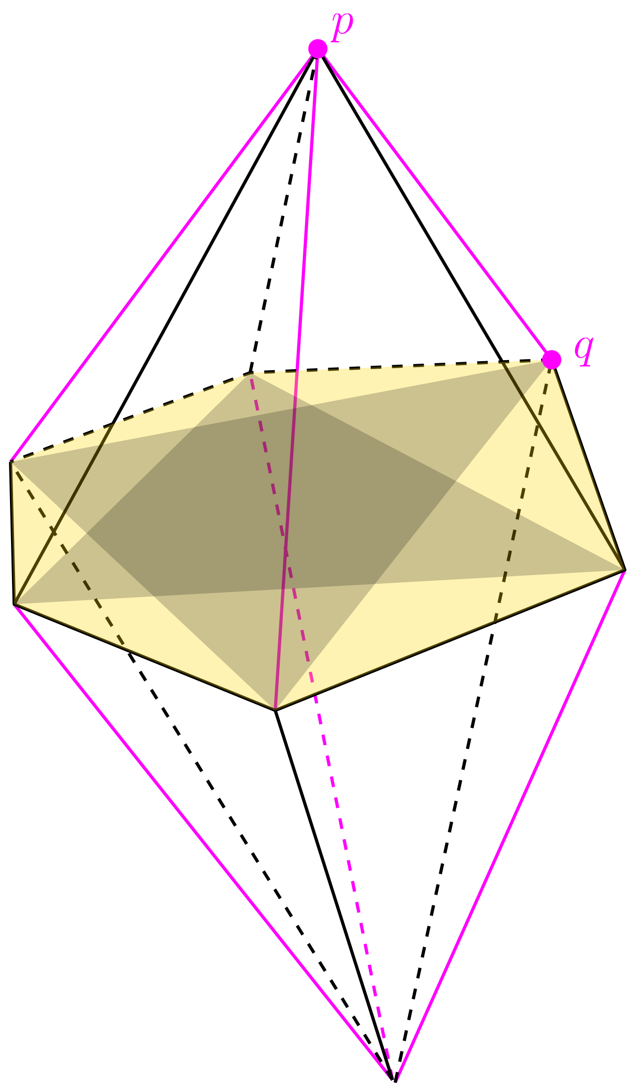
Vertex a is not incident to any purple edge.

Purple is not connected.

Vertex p is incident to more than 2 purple edges

Edge pq is incident to two red faces

There are no blue faces



COMPUTING A CONVEX HULL IN 3D

Notation

- Faces of C_1 and C_2 that appear in C are called blue.
- Faces of C_1 and C_2 that do not appear in C are called red.
- Edges and vertices are colored with the intersection color (blue, red and purple) of their incident faces.

Be careful!

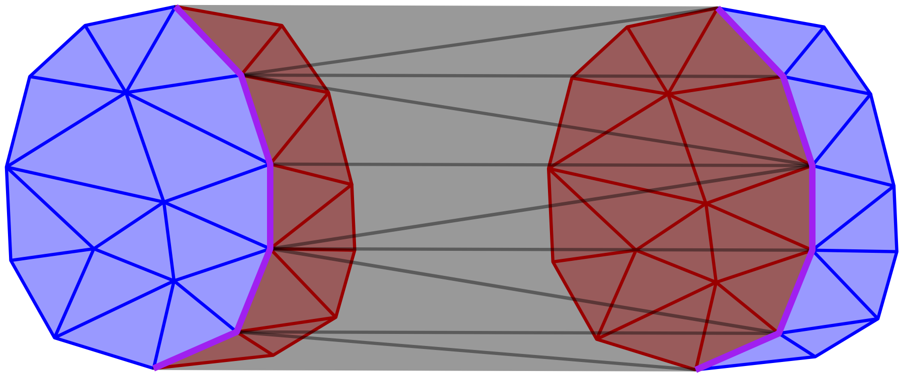
Vertex a is not incident to any purple edge.

Purple is not connected.

Vertex p is incident to more than 2 purple edges

Edge pq is incident to two red faces

There are no blue faces



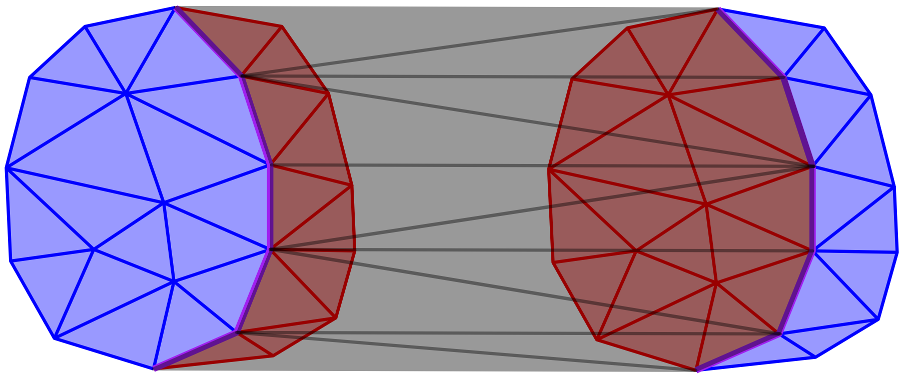
Corollary

It is not always true that the purple edges and vertices form a cycle.

COMPUTING A CONVEX HULL IN 3D

Proposition 1

The edges of $C \setminus C_1 \cup C_2$ are the convex hull of two purple vertices, one from C_1 and the other from C_2.
The faces of $C \setminus C_1 \cup C_2$ are the convex hull of a purple vertex from C_i and a purple edge from C_j, $i \neq j$.



CONVEX HULLS IN 3D

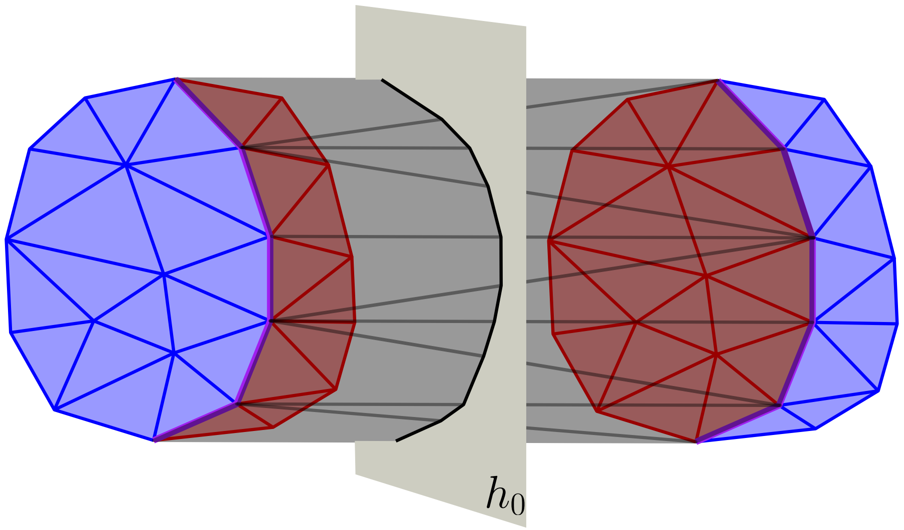
COMPUTING A CONVEX HULL IN 3D

Proposition 1

The edges of $C \setminus C_1 \cup C_2$ are the convex hull of two purple vertices, one from C_1 and the other from C_2.
The faces of $C \setminus C_1 \cup C_2$ are the convex hull of a purple vertex from C_i and a purple edge from C_j, $i \neq j$.

Proposition 2

Let h_0 be a plane separating C_1 from C_2. The faces and edges of $C \setminus C_1 \cup C_2$ intersect h_0 in a convex polygon C_0, and the circular order on the edges and vertices of C_0 induce a circular order on the faces and edges of $C \setminus C_1 \cup C_2$.



COMPUTING A CONVEX HULL IN 3D

Proposition 1

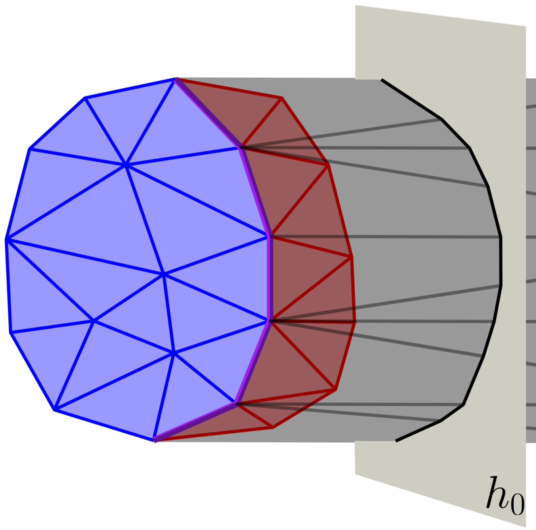
The edges of $C \setminus C_1 \cup C_2$ are the convex hull of two purple vertices, one from C_1 and the other from C_2. The faces of $C \setminus C_1 \cup C_2$ are the convex hull of a purple vertex from C_i and a purple edge from C_j, $i \neq j$.

Proposition 2

Let h_0 be a plane separating C_1 from C_2. The faces and edges of $C \setminus C_1 \cup C_2$ intersect h_0 in a convex polygon C_0, and the circular order on the edges and vertices of C_0 induce a circular order on the faces and edges of $C \setminus C_1 \cup C_2$.

Merging algorithm

1. Find an edge of $C \setminus C_1 \cup C_2$
2. Find the remaining new faces and edges in the order induced by C_0
3. Identify the red faces, edges and vertices and update the DCEL



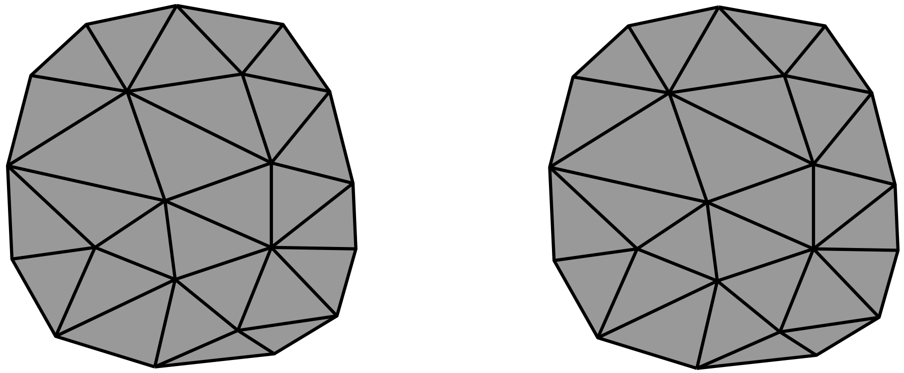
COMPUTING A CONVEX HULL IN 3D

Step 1: Finding the first new edge

COMPUTING A CONVEX HULL IN 3D

Step 1: Finding the first new edge

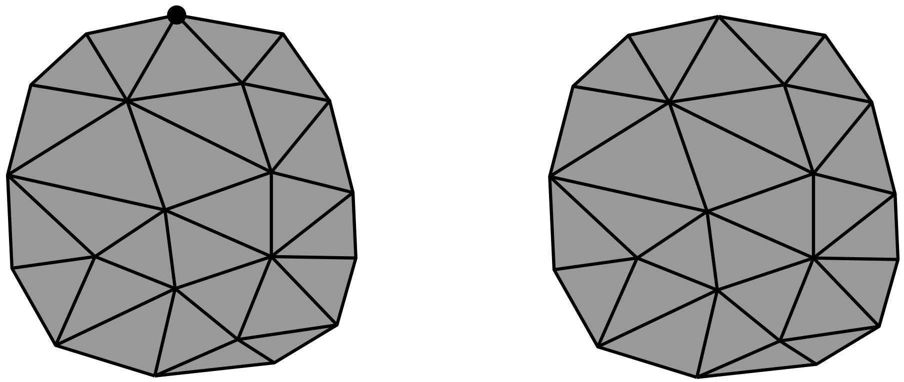
1. Orthogonally project C_1 and C_2 onto the plane $z = 0$. Let C'_1 and C'_2 respectively be their projections.
   - The vertex v of greatest abscissa in C_i projects onto the vertex v' of greatest abscissa in C'_i.
   - The following vertex of C'_i is the projection of one of the neigbors of v in C_i.
2. Find $v'_1 v'_2$, one of the external bitangents of C'_1 and C'_2.
3. Then $v_1 v_2$ is an edge of $C \setminus C_1 \cup C_2$.



COMPUTING A CONVEX HULL IN 3D

Step 1: Finding the first new edge

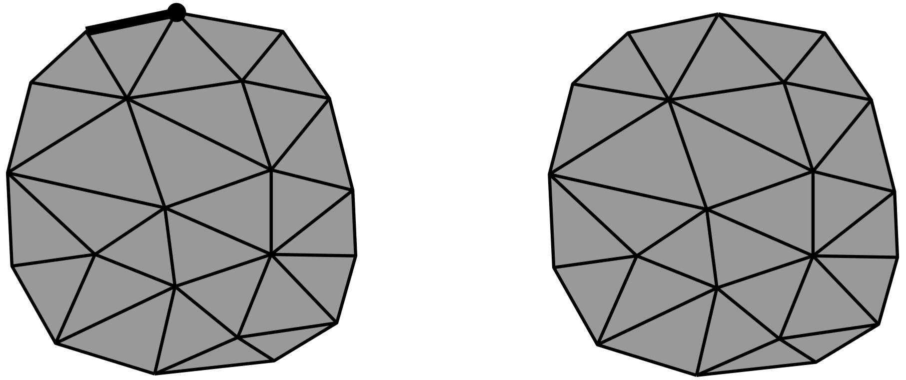
1. Orthogonally project C_1 and C_2 onto the plane $z = 0$. Let C_1' and C_2' respectively be their projections.
   - The vertex v of greatest abscissa in C_i projects onto the vertex v' of greatest abscissa in C_i'.
   - The following vertex of C_i' is the projection of one of the neigbors of v in C_i.
2. Find $v_1'v_2'$, one of the external bitangents of C_1' and C_2'.
3. Then v_1v_2 is an edge of $C \setminus C_1 \cup C_2$.



COMPUTING A CONVEX HULL IN 3D

Step 1: Finding the first new edge

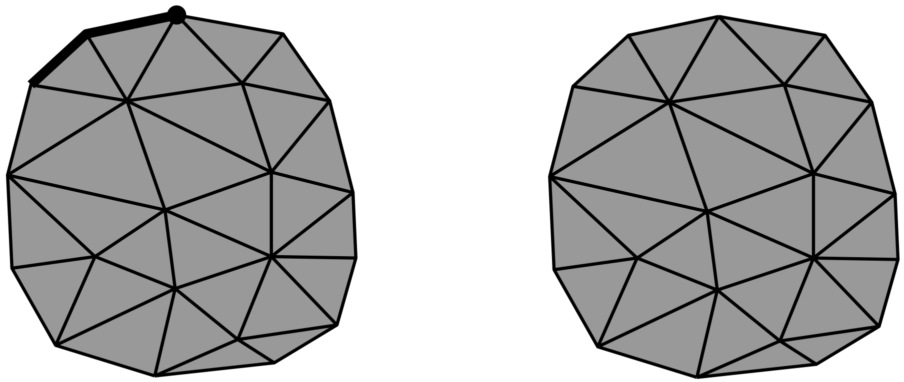
1. Orthogonally project C_1 and C_2 onto the plane $z = 0$. Let C_1' and C_2' respectively be their projections.
   - The vertex v of greatest abscissa in C_i projects onto the vertex v' of greatest abscissa in C_i'.
   - The following vertex of C_i' is the projection of one of the neigbors of v in C_i.
2. Find $v_1'v_2'$, one of the external bitangents of C_1' and C_2'.
3. Then v_1v_2 is an edge of $C \setminus C_1 \cup C_2$.



COMPUTING A CONVEX HULL IN 3D

Step 1: Finding the first new edge

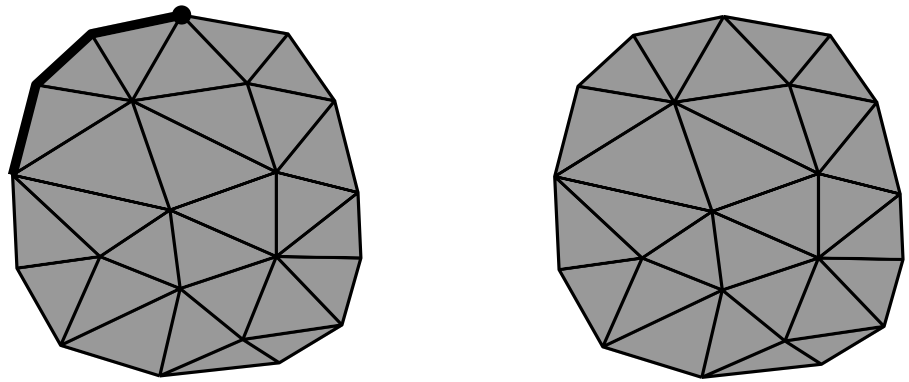
1. Orthogonally project C_1 and C_2 onto the plane $z = 0$. Let C_1' and C_2' respectively be their projections.
   - The vertex v of greatest abscissa in C_i projects onto the vertex v' of greatest abscissa in C_i'.
   - The following vertex of C_i' is the projection of one of the neigbors of v in C_i.
2. Find $v_1'v_2'$, one of the external bitangents of C_1' and C_2'.
3. Then v_1v_2 is an edge of $C \setminus C_1 \cup C_2$.



COMPUTING A CONVEX HULL IN 3D

Step 1: Finding the first new edge

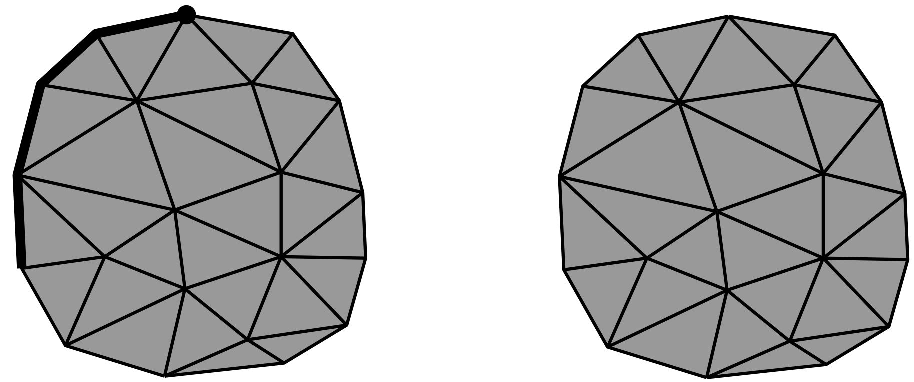
1. Orthogonally project C_1 and C_2 onto the plane $z = 0$. Let C_1' and C_2' respectively be their projections.
   - The vertex v of greatest abscissa in C_i projects onto the vertex v' of greatest abscissa in C_i'.
   - The following vertex of C_i' is the projection of one of the neigbors of v in C_i.
2. Find $v_1'v_2'$, one of the external bitangents of C_1' and C_2'.
3. Then v_1v_2 is an edge of $C \setminus C_1 \cup C_2$.



COMPUTING A CONVEX HULL IN 3D

Step 1: Finding the first new edge

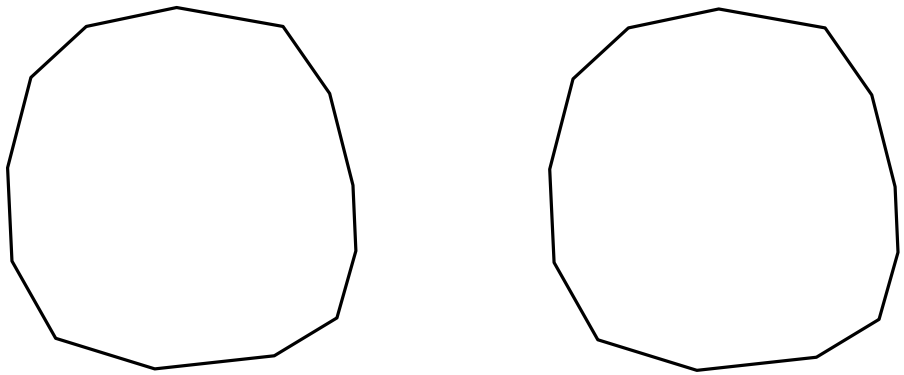
1. Orthogonally project C_1 and C_2 onto the plane $z = 0$. Let C'_1 and C'_2 respectively be their projections.
   - The vertex v of greatest abscissa in C_i projects onto the vertex v' of greatest abscissa in C'_i.
   - The following vertex of C'_i is the projection of one of the neigbors of v in C_i.
2. Find $v'_1 v'_2$, one of the external bitangents of C'_1 and C'_2.
3. Then $v_1 v_2$ is an edge of $C \setminus C_1 \cup C_2$.



COMPUTING A CONVEX HULL IN 3D

Step 1: Finding the first new edge

1. Orthogonally project C_1 and C_2 onto the plane $z = 0$. Let C_1' and C_2' respectively be their projections.
   - The vertex v of greatest abscissa in C_i projects onto the vertex v' of greatest abscissa in C_i'.
   - The following vertex of C_i' is the projection of one of the neigbors of v in C_i.
2. Find $v_1'v_2'$, one of the external bitangents of C_1' and C_2'.
3. Then v_1v_2 is an edge of $C \setminus C_1 \cup C_2$.

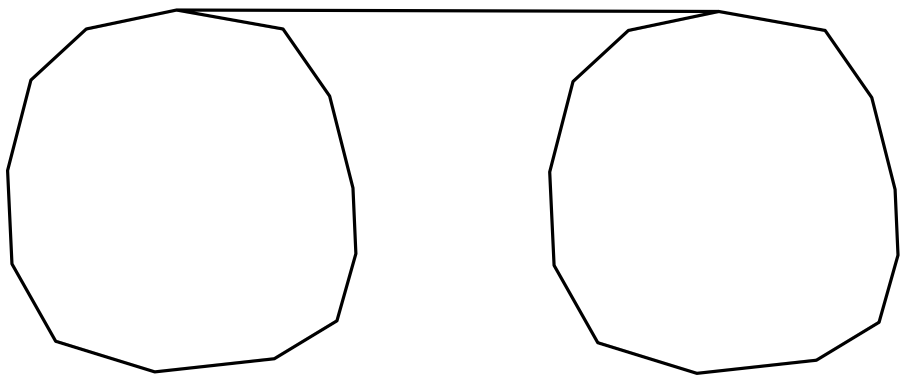


CONVEX HULLS IN 3D

COMPUTING A CONVEX HULL IN 3D

Step 1: Finding the first new edge

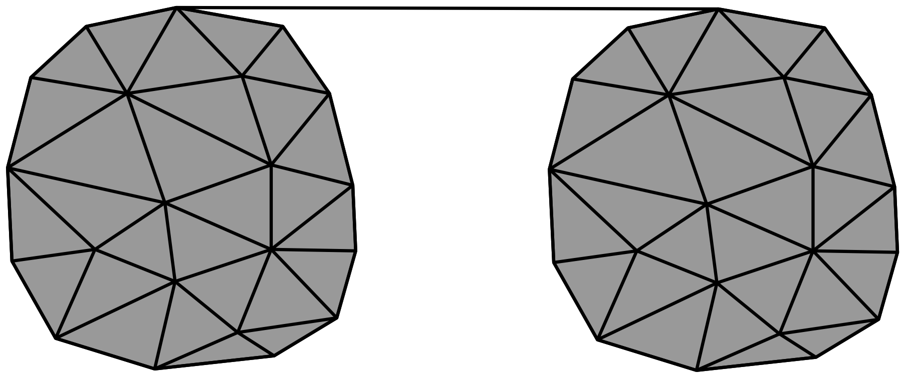
1. Orthogonally project C_1 and C_2 onto the plane $z = 0$. Let C_1' and C_2' respectively be their projections.
   - The vertex v of greatest abscissa in C_i projects onto the vertex v' of greatest abscissa in C_i'.
   - The following vertex of C_i' is the projection of one of the neigbors of v in C_i.
2. Find $v_1'v_2'$, one of the external bitangents of C_1' and C_2'.
3. Then v_1v_2 is an edge of $C \setminus C_1 \cup C_2$.



COMPUTING A CONVEX HULL IN 3D

Step 1: Finding the first new edge

1. Orthogonally project C_1 and C_2 onto the plane $z = 0$. Let C_1' and C_2' respectively be their projections.
   - The vertex v of greatest abscissa in C_i projects onto the vertex v' of greatest abscissa in C_i'.
   - The following vertex of C_i' is the projection of one of the neigbors of v in C_i.
2. Find $v_1'v_2'$, one of the external bitangents of C_1' and C_2'.
3. Then v_1v_2 is an edge of $C \setminus C_1 \cup C_2$.



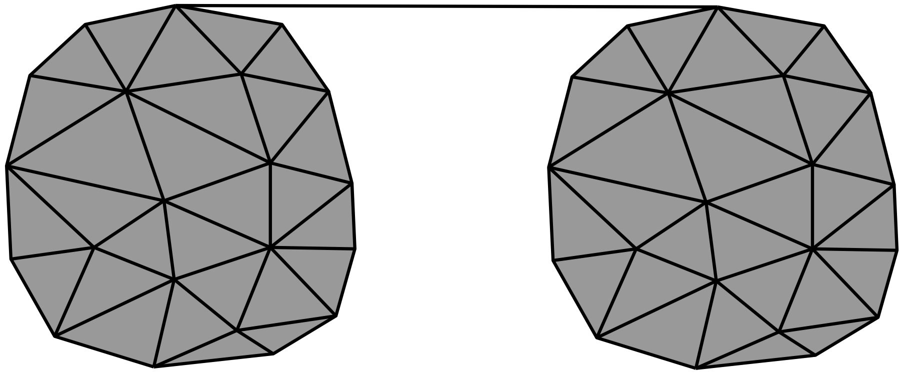
CONVEX HULLS IN 3D

COMPUTING A CONVEX HULL IN 3D

Step 1: Finding the first new edge

1. Orthogonally project C_1 and C_2 onto the plane $z = 0$. Let C_1' and C_2' respectively be their projections.
   - The vertex v of greatest abscissa in C_i projects onto the vertex v' of greatest abscissa in C_i'.
   - The following vertex of C_i' is the projection of one of the neigbors of v in C_i.
2. Find $v_1'v_2'$, one of the external bitangents of C_1' and C_2'.
3. Then v_1v_2 is an edge of $C \setminus C_1 \cup C_2$.

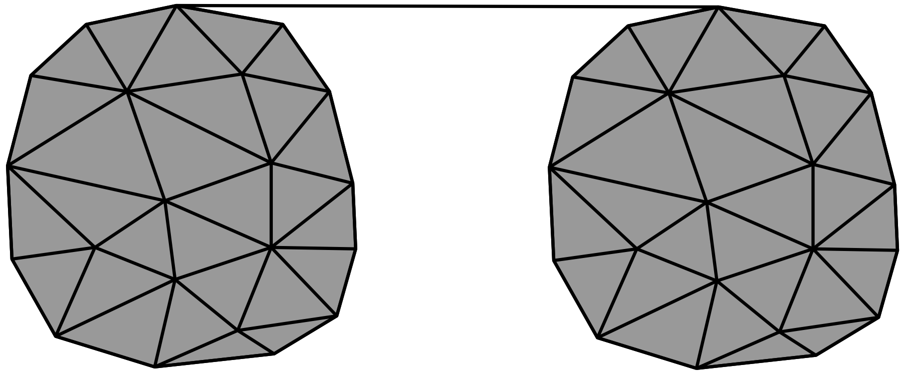
Running time: $O(n)$



COMPUTING A CONVEX HULL IN 3D

Step 1: Finding the first new edge

1. Orthogonally project C_1 and C_2 onto the plane $z = 0$. Let C_1' and C_2' respectively be their projections.
   - The vertex v of greatest abscissa in C_i projects onto the vertex v' of greatest abscissa in C_i'.
   - The following vertex of C_i' is the projection of one of the neigbors of v in C_i.
2. Find $v_1'v_2'$, one of the external bitangents of C_1' and C_2'.
3. Then v_1v_2 is an edge of $C \setminus C_1 \cup C_2$.



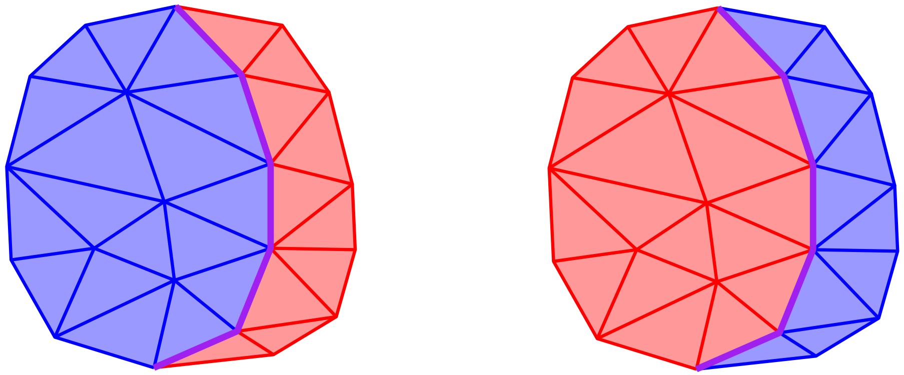
Running time: $O(n)$

1. Each edge of C_i is tested at most twice.
2. Common external tangents can be found in linear time.
3. Retrieving v_i from v_i' is done in constant time.

CONVEX HULLS IN 3D

COMPUTING A CONVEX HULL IN 3D

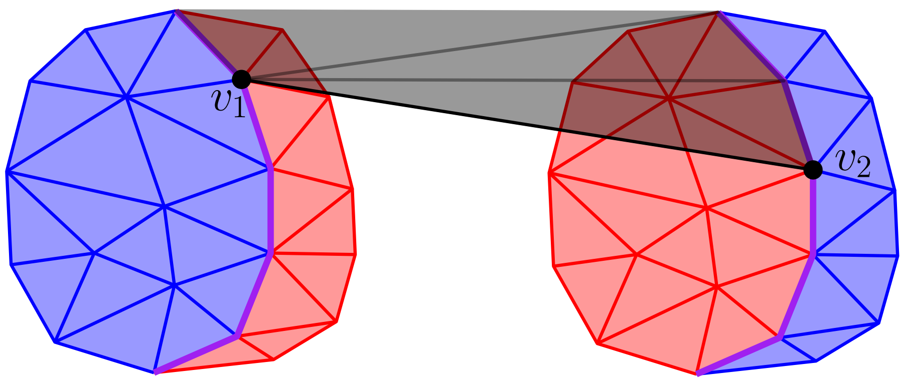
Step 2: Finding the remaining new faces and edges in the order induced by C_0



COMPUTING A CONVEX HULL IN 3D

Step 2: Finding the remaining new faces and edges in the order induced by C_0

Let $v_1 v_2$ be the last discovered new edge ($v_i \in C_i$).

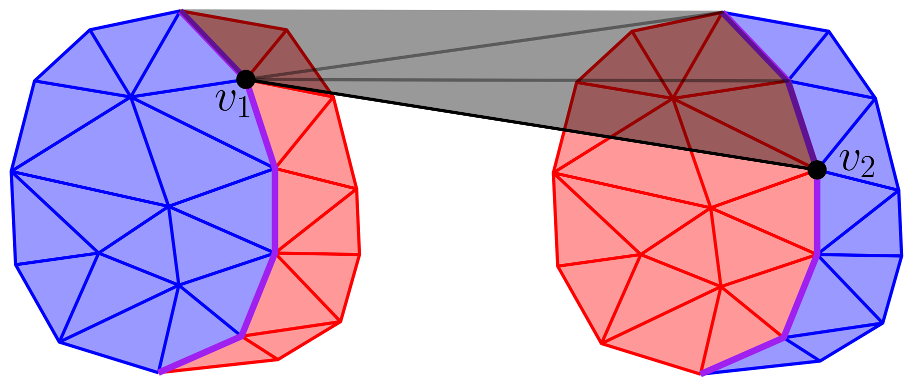


COMPUTING A CONVEX HULL IN 3D

Step 2: Finding the remaining new faces and edges in the order induced by C_0

Let v_1v_2 be the last discovered new edge ($v_i \in C_i$).

Let π be:
- If v_1v_2 is the first edge, π is the plane through v_1, v_2, v_1', v_2'.
- Otherwise, π is the plane containing the last discovered new face, which is incident to v_1v_2 on its left.



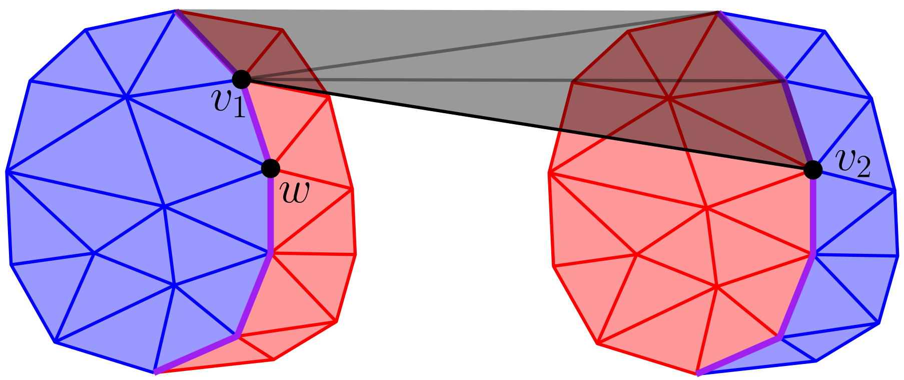
COMPUTING A CONVEX HULL IN 3D

Step 2: Finding the remaining new faces and edges in the order induced by C_0

Let v_1v_2 be the last discovered new edge ($v_i \in C_i$).

Let π be:
- If v_1v_2 is the first edge, π is the plane through v_1, v_2, v_1', v_2'.
- Otherwise, π is the plane containing the last discovered new face, which is incident to v_1v_2 on its left.

The next new face found is v_1v_2w. It is incident to v_1v_2 on its right, and w is the neighbor of either v_1 or v_2 forming smaller angle with π.



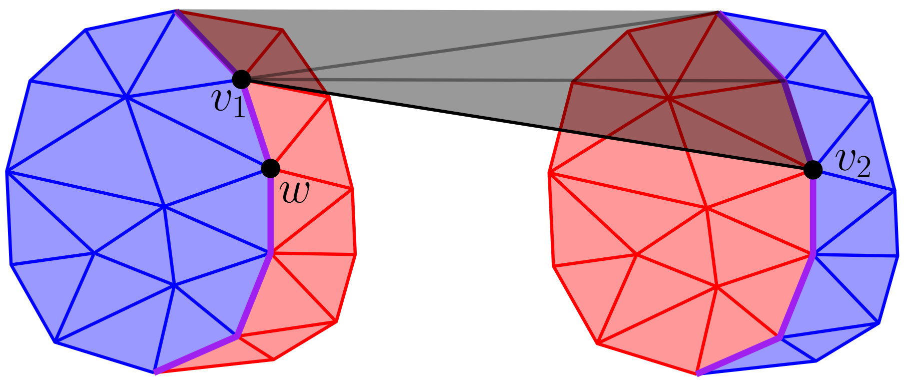
COMPUTING A CONVEX HULL IN 3D

Step 2: Finding the remaining new faces and edges in the order induced by C_0

Let v_1v_2 be the last discovered new edge ($v_i \in C_i$).

Let π be:
- If v_1v_2 is the first edge, π is the plane through v_1, v_2, v_1', v_2'.
- Otherwise, π is the plane containing the last discovered new face, which is incident to v_1v_2 on its left.

The next new face found is v_1v_2w. It is incident to v_1v_2 on its right, and w is the neighbor of either v_1 or v_2 forming smaller angle with π.



Can we avoid checking all neighbors of v_i again and again?

CONVEX HULLS IN 3D

COMPUTING A CONVEX HULL IN 3D

Step 2: Finding the remaining new faces and edges in the order induced by C_0

Lemma 1

There is only one possible candidate $w_i \in C_i$ and it can be caracterized locally.

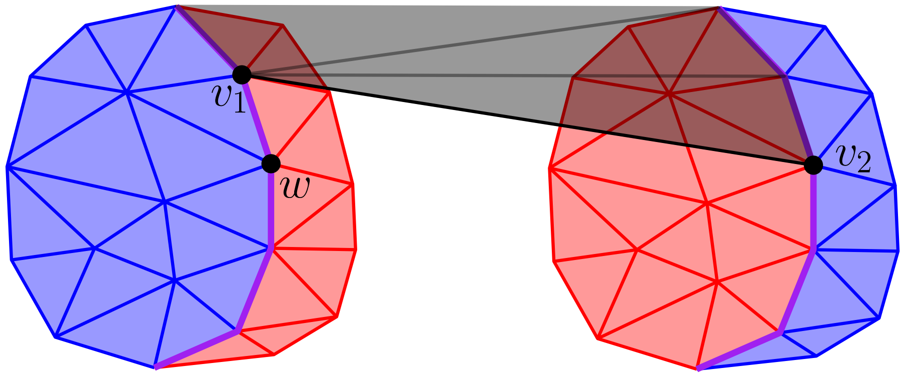
COMPUTING A CONVEX HULL IN 3D

Step 2: Finding the remaining new faces and edges in the order induced by C_0

Lemma 1

There is only one possible candidate $w_i \in C_i$ and it can be caracterized locally.

Proof: WLG, let w_1 be a vertex adjacent to v_1. Let $p(w_1), s(w_1)$ respectively be its predecessor and its successor in the circular order about v_1. Let h_1 be the plane $v_1w_1v_2$. Among the two halfspaces defined by h_1 let h_1^+ be the one opposite to vector $n = v_1w_1 \wedge v_1v_2$. If w_1 is a candidate, then h_1 supports C_1 and v_2. Therefore $p(w_1), s(w_1) \in h_1^+$. Reciprocally, if $p(w_1), s(w_1) \in h_1^+$ then $v_1w_1p(w_1)$ and $v_1w_1s(w_1)$ both support C_1. Therefore also h_1 supports C_1.



COMPUTING A CONVEX HULL IN 3D

Step 2: Finding the remaining new faces and edges in the order induced by C_0

Lemma 1

There is only one possible candidate $w_i \in C_i$ and it can be caracterized locally.

Proof: WLG, let w_1 be a vertex adjacent to v_1. Let $p(w_1), s(w_1)$ respectively be its predecessor and its successor in the circular order about v_1. Let h_1 be the plane $v_1 w_1 v_2$. Among the two halfspaces defined by h_1 let h_1^+ be the one opposite to vector $n = v_1 w_1 \wedge v_1 v_2$. If w_1 is a candidate, then h_1 supports C_1 and v_2. Therefore $p(w_1), s(w_1) \in h_1^+$. Reciprocally, if $p(w_1), s(w_1) \in h_1^+$ then $v_1 w_1 p(w_1)$ and $v_1 w_1 s(w_1)$ both support C_1. Therefore also h_1 supports C_1.

Lemma 2

When v_i is incident to several new edges, the successive candidates are found in circular order about v_i.

COMPUTING A CONVEX HULL IN 3D

Step 2: Finding the remaining new faces and edges in the order induced by C_0

Lemma 1

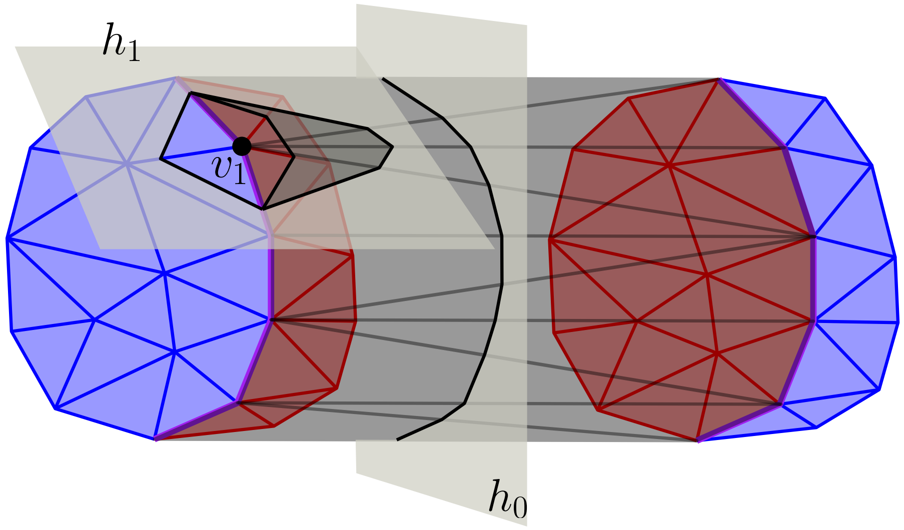
There is only one possible candidate $w_i \in C_i$ and it can be caracterized locally.

Proof: WLG, let w_1 be a vertex adjacent to v_1. Let $p(w_1), s(w_1)$ respectively be its predecessor and its successor in the circular order about v_1. Let h_1 be the plane $v_1w_1v_2$. Among the two halfspaces defined by h_1 let h_1^+ be the one opposite to vector $n = v_1w_1 \wedge v_1v_2$. If w_1 is a candidate, then h_1 supports C_1 and v_2. Therefore $p(w_1), s(w_1) \in h_1^+$. Reciprocally, if $p(w_1), s(w_1) \in h_1^+$ then $v_1w_1p(w_1)$ and $v_1w_1s(w_1)$ both support C_1. Therefore also h_1 supports C_1.

Lemma 2

When v_i is incident to several new edges, the successive candidates are found in circular order about v_i.

Proof: WLG, v_1 can be separated from the remaining vertices of C_1 and C_2 by a plane h_1, wich intersects all the edges of C_1 and C incident to v_1 forming two convex polygons. The circular order of the vertices of $C_1 \cap h_1$ and $C \cap h_1$ is the same, and it also coincides with the circular order of $C \cap h_0$.



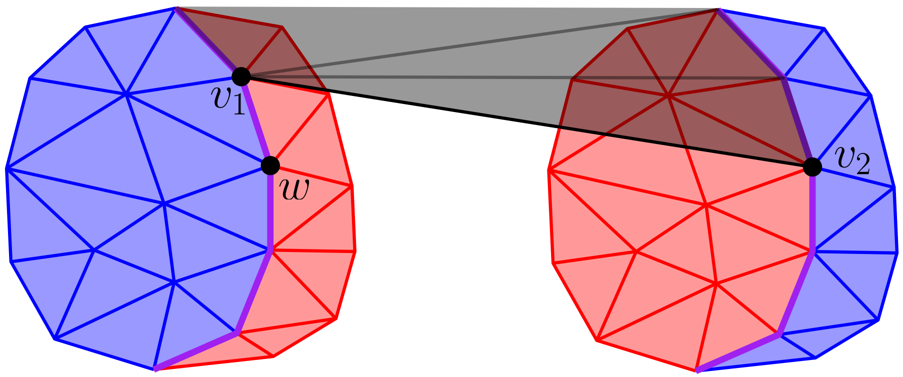
COMPUTING A CONVEX HULL IN 3D

Step 2: Finding the remaining new faces and edges in the order induced by C_0

At each step:

1. From face uv_1v_2 find the best candidates w_1, adjacent to v_1 and w_2 adjacent to v_2.
2. Choose w to be the best of w_1 and w_2.
3. You have found:
   - a purple vertex w
   - a purple edge v_iw
   - a new face v_1v_2w

   that can be added to the DCEL.



CONVEX HULLS IN 3D

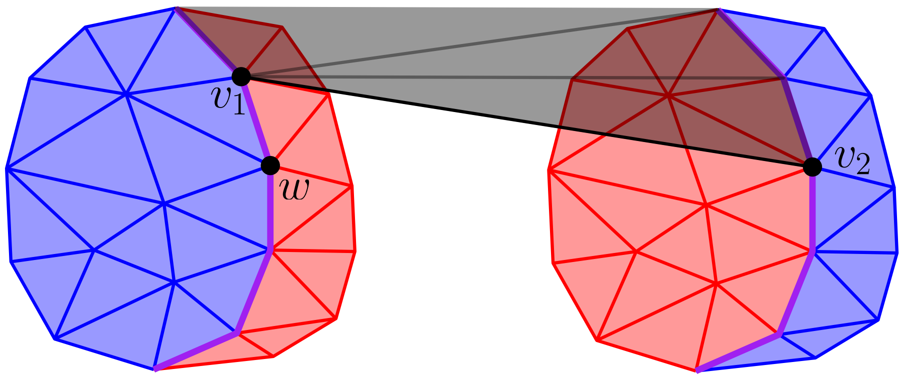
COMPUTING A CONVEX HULL IN 3D

Step 2: Finding the remaining new faces and edges in the order induced by C_0

At each step:

1. From face uv_1v_2 find the best candidates w_1, adjacent to v_1 and w_2 adjacent to v_2.
2. Choose w to be the best of w_1 and w_2.
3. You have found:
   - a purple vertex w
   - a purple edge v_iw
   - a new face v_1v_2w

   that can be added to the DCEL.



Running time: $O(n)$

At each step, the neighbors of v_i are tested in order (Lemma 2) and ecah test takes $O(1)$ time (Lemma 1).

CONVEX HULLS IN 3D

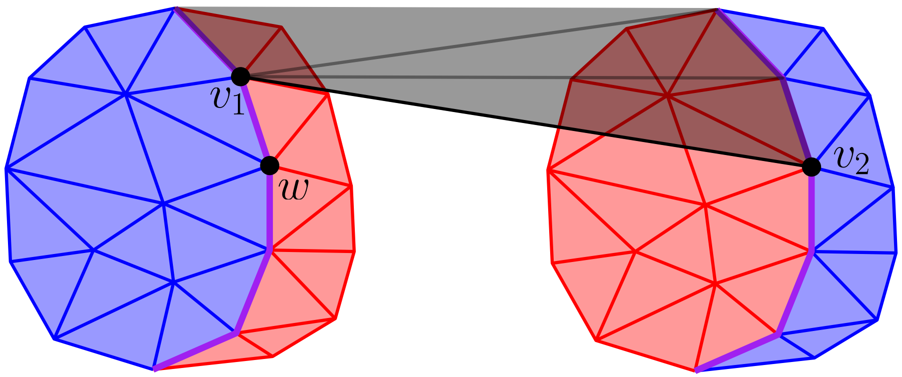
COMPUTING A CONVEX HULL IN 3D

Step 2: Finding the remaining new faces and edges in the order induced by C_0

At each step:

1. From face uv_1v_2 find the best candidates w_1, adjacent to v_1 and w_2 adjacent to v_2.
2. Choose w to be the best of w_1 and w_2.
3. You have found:
   - a purple vertex w
   - a purple edge v_iw
   - a new face v_1v_2w

   that can be added to the DCEL.



Running time: $O(n)$

At each step, the neighbors of v_i are tested in order (Lemma 2) and ecah test takes $O(1)$ time (Lemma 1).

Step 3: Identify the red faces, edges and vertices, and update the DCEL

CONVEX HULLS IN 3D

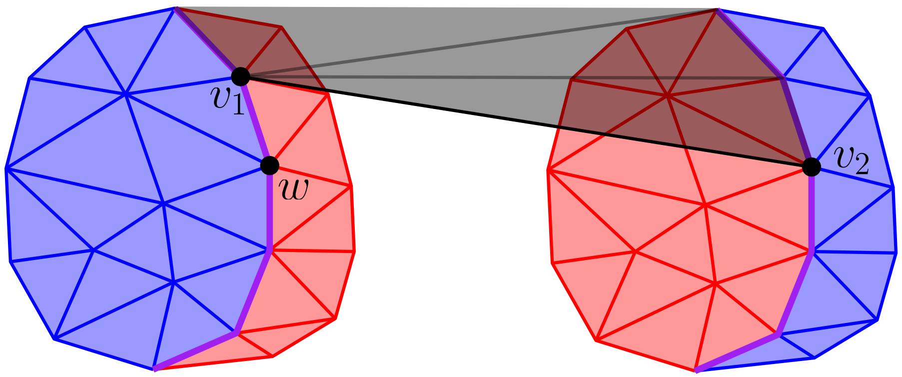
COMPUTING A CONVEX HULL IN 3D

Step 2: Finding the remaining new faces and edges in the order induced by C_0

At each step:

1. From face uv_1v_2 find the best candidates w_1, adjacent to v_1 and w_2 adjacent to v_2.
2. Choose w to be the best of w_1 and w_2.
3. You have found:
   - a purple vertex w
   - a purple edge v_iw
   - a new face v_1v_2w

   that can be added to the DCEL.



Running time: $O(n)$

At each step, the neighbors of v_i are tested in order (Lemma 2) and ecah test takes $O(1)$ time (Lemma 1).

Step 3: Identify the red faces, edges and vertices, and update the DCEL

In the previous step, each time a purple edge is found, its incident faces are labelled blue or red.

CONVEX HULLS IN 3D

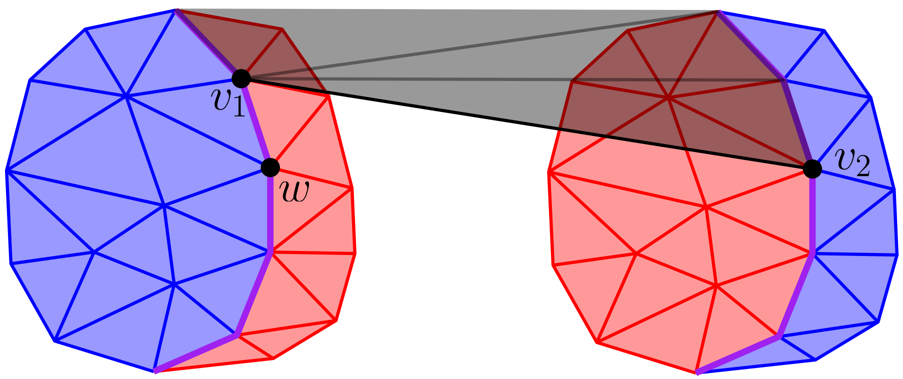
COMPUTING A CONVEX HULL IN 3D

Step 2: Finding the remaining new faces and edges in the order induced by C_0

At each step:

1. From face uv_1v_2 find the best candidates w_1, adjacent to v_1 and w_2 adjacent to v_2.
2. Choose w to be the best of w_1 and w_2.
3. You have found:
   - a purple vertex w
   - a purple edge v_iw
   - a new face v_1v_2w

   that can be added to the DCEL.



Running time: $O(n)$

At each step, the neighbors of v_i are tested in order (Lemma 2) and ecah test takes $O(1)$ time (Lemma 1).

Step 3: Identify the red faces, edges and vertices, and update the DCEL

In the previous step, each time a purple edge is found, its incident faces are labelled blue or red.

In this step:

1. All unlabelled faces adjacent to a red face are recursively labelled red.
2. All unlabelled edges incident to a red face are labelled red.
3. All unlabelled vertex incident to a red or purple edge is labelled red.
4. Remove all red faces, edges and vertices from the DCEL.

CONVEX HULLS IN 3D

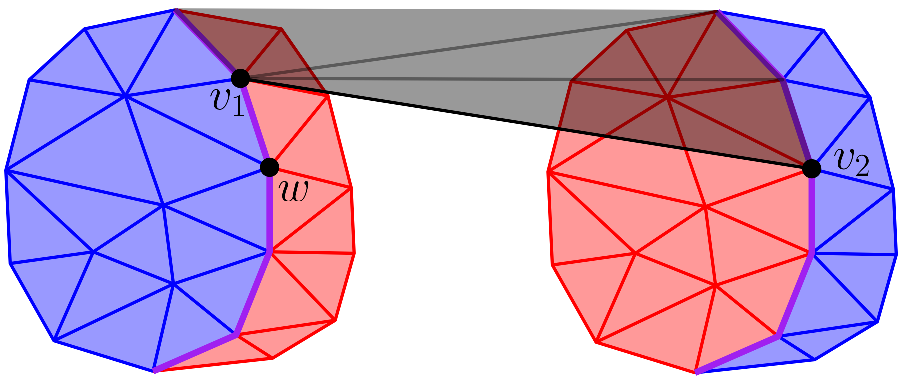
COMPUTING A CONVEX HULL IN 3D

Step 2: Finding the remaining new faces and edges in the order induced by C_0

At each step:

1. From face uv_1v_2 find the best candidates w_1, adjacent to v_1 and w_2 adjacent to v_2.
2. Choose w to be the best of w_1 and w_2.
3. You have found:
   - a purple vertex w
   - a purple edge v_iw
   - a new face v_1v_2w

   that can be added to the DCEL.



Running time: $O(n)$

At each step, the neighbors of v_i are tested in order (Lemma 2) and ecah test takes $O(1)$ time (Lemma 1).

Step 3: Identify the red faces, edges and vertices, and update the DCEL

In the previous step, each time a purple edge is found, its incident faces are labelled blue or red.

In this step:

1. All unlabelled faces adjacent to a red face are recursively labelled red.
2. All unlabelled edges incident to a red face are labelled red.
3. All unlabelled vertex incident to a red or purple edge is labelled red.
4. Remove all red faces, edges and vertices from the DCEL.

Running time: $O(n)$

CONVEX HULLS IN 3D

COMPUTING A CONVEX HULL IN 3D

Algorithm

Input: $p_1, \ldots, p_n \in \mathbb{R}^3$
Output: $ch(p_1, \ldots, p_n)$

1. Initialization
   Sort $p_1, \ldots, p_n$ by abscissa.

2. Division
   Partition set $P = \{p_1, \ldots, p_n\}$ into two equally sized subsets P_1 and P_2 by means of a vertical plane h_0.

3. Recursion
   Compute $C_1 = ch(P_1)$ and $C_2 = ch(P_2)$.

4. Merging
   Compute $C = ch(C_1 \cup C_2)$.

CONVEX HULLS IN 3D

COMPUTING A CONVEX HULL IN 3D

Algorithm

Input: $p_1, \ldots, p_n \in \mathbb{R}^3$
Output: $ch(p_1, \ldots, p_n)$

1. Initialization
   Sort $p_1, \ldots, p_n$ by abscissa.
2. Division
   Partition set $P = \{p_1, \ldots, p_n\}$ into two equally sized subsets P_1 and P_2 by means of a vertical plane h_0.
3. Recursion
   Compute $C_1 = ch(P_1)$ and $C_2 = ch(P_2)$.
4. Merging
   Compute $C = ch(C_1 \cup C_2)$.
   1. Find an edge of $C \setminus C_1 \cup C_2$
   2. Find the remaining new faces and edges in the order induced by C_0
   3. Identify the red faces, edges and vertices and update the DCEL

COMPUTING A CONVEX HULL IN 3D

Algorithm

Input: $p_1, \ldots, p_n \in \mathbb{R}^3$
Output: $ch(p_1, \ldots, p_n)$

1. Initialization
   Sort $p_1, \ldots, p_n$ by abscissa.

2. Division
   Partition set $P = \{p_1, \ldots, p_n\}$ into two equally sized subsets P_1 and P_2 by means of a vertical plane h_0.

3. Recursion
   Compute $C_1 = ch(P_1)$ and $C_2 = ch(P_2)$.

4. Merging
   Compute $C = ch(C_1 \cup C_2)$.

   $O(n)$

   1. Find an edge of $C \setminus C_1 \cup C_2$

   2. Find the remaining new faces and edges in the order induced by C_0

   3. Identify the red faces, edges and vertices and update the DCEL

COMPUTING A CONVEX HULL IN 3D

Algorithm

Input: $p_1, \ldots, p_n \in \mathbb{R}^3$
Output: $ch(p_1, \ldots, p_n)$

1. Initialization
   Sort $p_1, \ldots, p_n$ by abscissa.

2. Division
   Partition set $P = \{p_1, \ldots, p_n\}$ into two equally sized subsets P_1 and P_2 by means of a vertical plane h_0.

3. Recursion
   Compute $C_1 = ch(P_1)$ and $C_2 = ch(P_2)$.

4. Merging
   Compute $C = ch(C_1 \cup C_2)$. $O(n)$
   1. Find an edge of $C \setminus C_1 \cup C_2$
   2. Find the remaining new faces and edges in the order induced by C_0
   3. Identify the red faces, edges and vertices and update the DCEL

Theorem: The algorithm computes the convex hull of n points in E^3 in $O(n \log n)$ time and $O(n)$ space. These bounds are optimal.

FURTHER READING

J.-D. Boissonat. M. Yvinec, **Algorithmic Geometry**, Cambridge University Press, 1998.