The Quickhull Algorithm for Higher Dimensional Convex Hulls

Chirantan Mukherjee

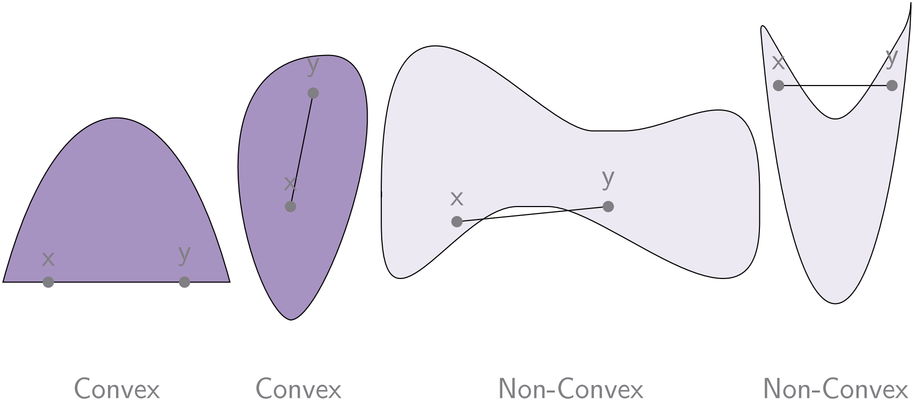
University of Western Ontario

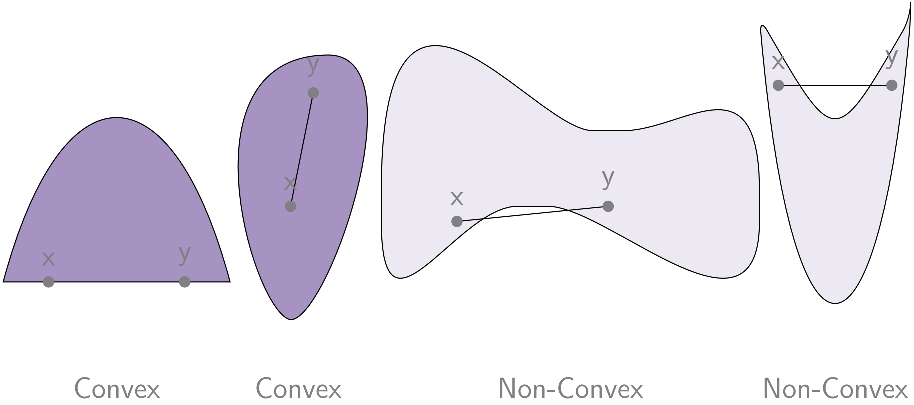
July 03, 2023

Saskatchewan is flat as a pancake, this is first time seeing a hill. In Québec, July 1st is also colloquially called 'moving day', because many leases for apartments get over in this time. Alaska is being tied and gagged, because it's very Canadian dream to kidnap Alaska!

2013 - r/polandball

Content


x
y
x
y
x
y
x
y
Convex
Convex
Non-Convex
Non-Convex

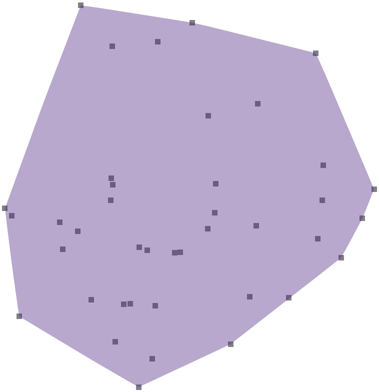


A set S is called convex if the line joining any two points in S is in S, i.e.,

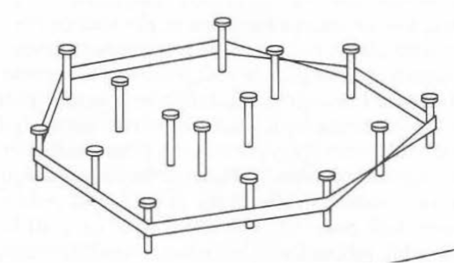
$$\forall x, y \in S, \forall \lambda \in [0, 1], \lambda x + (1 - \lambda) y \in S.$$

The convex hull of a set S the set of all convex combination of S, i.e.,

$$\textbf{conv}(S) := \{\Sigma_{i=1}^{n}\lambda_i x_i \mid \Sigma_{i=1}^{n}\lambda_i = 1, \lambda_i \in [0,1]\}.$$

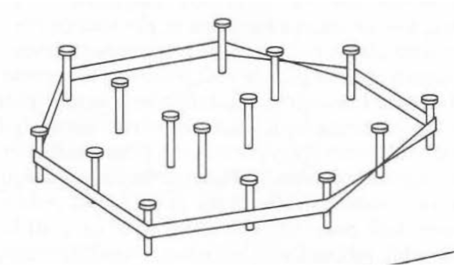


Existence and uniqueness of a convex hull.



1984 – Dewdney's Analog Gadgets

Existence and uniqueness of a convex hull.



1984 – Dewdney's Analog Gadgets

The **conv**(S) is the smallest convex set containing S, i.e. it is the intersection of all convex sets containing S.

- The first appearance of convex hull is in the form of a Newton Polygon, which is the lower half of a convex hull. It was corresponded in a letter from Isaac Newton to Henry Oldenburg in 1676.

- The first appearance of convex hull is in the form of a Newton Polygon, which is the lower half of a convex hull. It was corresponded in a letter from Isaac Newton to Henry Oldenburg in 1676.
- The term convex hull was coined by Garrett Birkhoff in 1935.

- The first appearance of convex hull is in the form of a Newton Polygon, which is the lower half of a convex hull. It was corresponded in a letter from Isaac Newton to Henry Oldenburg in 1676.
- The term convex hull was coined by Garrett Birkhoff in 1935.
- The first modern algorithm was published by Ronald Graham in 1972.

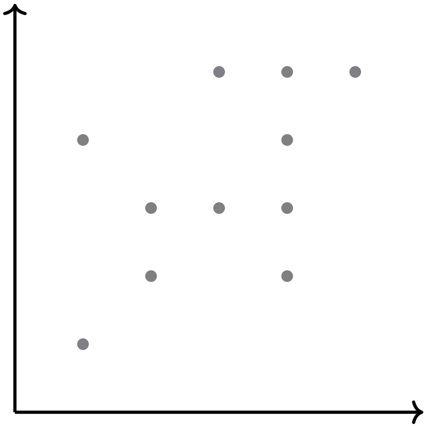
- The first appearance of convex hull is in the form of a Newton Polygon, which is the lower half of a convex hull. It was corresponded in a letter from Isaac Newton to Henry Oldenburg in 1676.
- The term convex hull was coined by Garrett Birkhoff in 1935.
- The first modern algorithm was published by Ronald Graham in 1972.
- The simplest algorithm was created independently by Chand and Kapur in 1970, and R. A. Jarvis in 1973.

- The first appearance of convex hull is in the form of a Newton Polygon, which is the lower half of a convex hull. It was corresponded in a letter from Isaac Newton to Henry Oldenburg in 1676.
- The term convex hull was coined by Garrett Birkhoff in 1935.
- The first modern algorithm was published by Ronald Graham in 1972.
- The simplest algorithm was created independently by Chand and Kapur in 1970, and R. A. Jarvis in 1973.
- The first optimal output-sensitive algorithm was published by Kirkpatrick and Seidel in 1986.

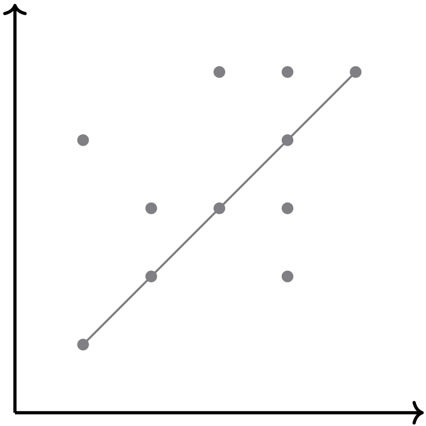
- The first appearance of convex hull is in the form of a Newton Polygon, which is the lower half of a convex hull. It was corresponded in a letter from Isaac Newton to Henry Oldenburg in 1676.
- The term convex hull was coined by Garrett Birkhoff in 1935.
- The first modern algorithm was published by Ronald Graham in 1972.
- The simplest algorithm was created independently by Chand and Kapur in 1970, and R. A. Jarvis in 1973.
- The first optimal output-sensitive algorithm was published by Kirkpatrick and Seidel in 1986.
- Best optimal output-sensitive algorithm created by Chan in 1996.

- The first appearance of convex hull is in the form of a Newton Polygon, which is the lower half of a convex hull. It was corresponded in a letter from Isaac Newton to Henry Oldenburg in 1676.
- The term convex hull was coined by Garrett Birkhoff in 1935.
- The first modern algorithm was published by Ronald Graham in 1972.
- The simplest algorithm was created independently by Chand and Kapur in 1970, and R. A. Jarvis in 1973.
- The first optimal output-sensitive algorithm was published by Kirkpatrick and Seidel in 1986.
- Best optimal output-sensitive algorithm created by Chan in 1996.
- Higher dimensional Quickhull was invented in 1996 by C. Bradford Barber, David P. Dobkin, and Hannu Huhdanpaa.

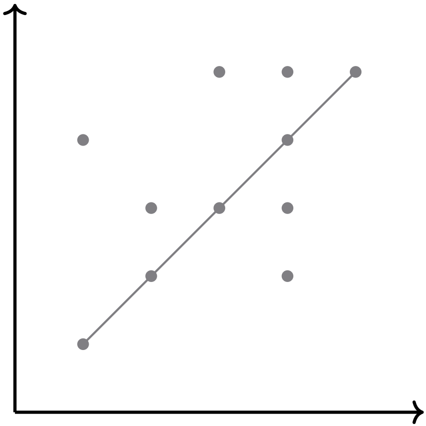
1. Find the point min-x, and another point max-x.



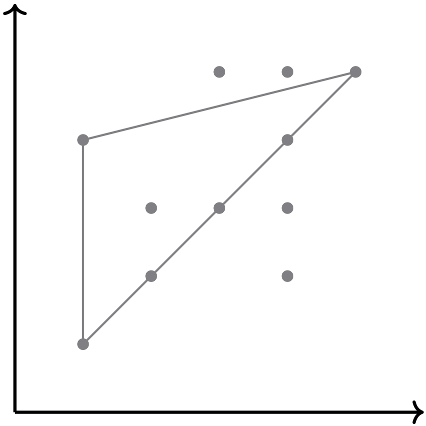
1. Find the point min-x, and another point max-x.
2. Join these two points with a line L that divides the shape into two parts.



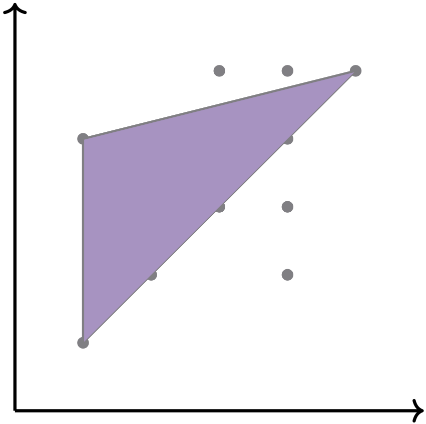
1. Find the point min-x, and another point max-x.
2. Join these two points with a line L that divides the shape into two parts.
3. Find the point P outwards with the maximum distance from L.



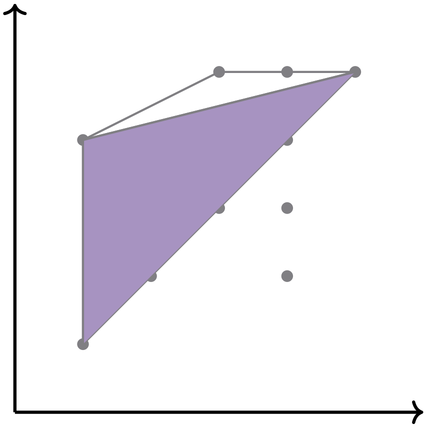
1. Find the point min-x, and another point max-x.
2. Join these two points with a line L that divides the shape into two parts.
3. Find the point P outwards with the maximum distance from L.
4. Join P with min-x and max-x, ignore all other points inside the triangle (convex hull).



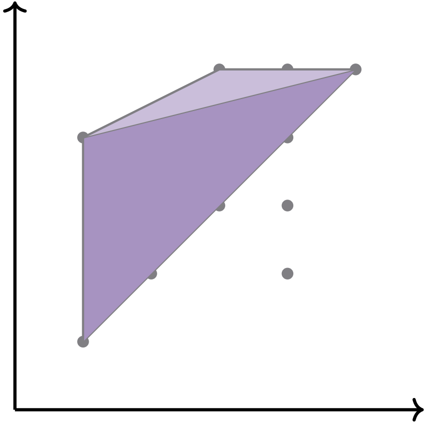
1. Find the point min-x, and another point max-x.
2. Join these two points with a line L that divides the shape into two parts.
3. Find the point P outwards with the maximum distance from L.
4. Join P with min-x and max-x, ignore all other points inside the triangle (convex hull).
5. Repeat step 3 and step 4 on the two lines formed by the two new sides of the triangle until all points are inside the convex hull.



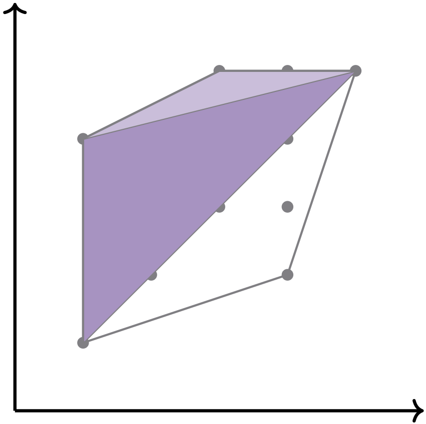
1. Find the point min-x, and another point max-x.
2. Join these two points with a line L that divides the shape into two parts.
3. Find the point P outwards with the maximum distance from L.
4. Join P with min-x and max-x, ignore all other points inside the triangle (convex hull).
5. Repeat step 3 and step 4 on the two lines formed by the two new sides of the triangle until all points are inside the convex hull.



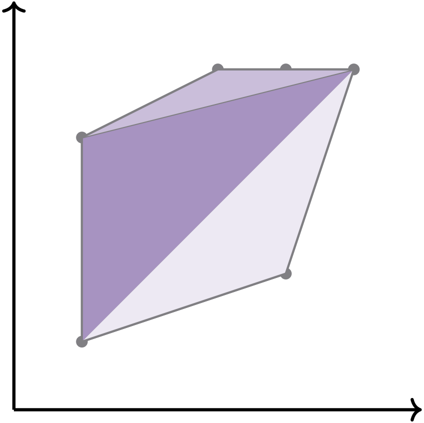
1. Find the point min-x, and another point max-x.
2. Join these two points with a line L that divides the shape into two parts.
3. Find the point P outwards with the maximum distance from L.
4. Join P with min-x and max-x, ignore all other points inside the triangle (convex hull).
5. Repeat step 3 and step 4 on the two lines formed by the two new sides of the triangle until all points are inside the convex hull.



1. Find the point min-x, and another point max-x.
2. Join these two points with a line L that divides the shape into two parts.
3. Find the point P outwards with the maximum distance from L.
4. Join P with min-x and max-x, ignore all other points inside the triangle (convex hull).
5. Repeat step 3 and step 4 on the two lines formed by the two new sides of the triangle until all points are inside the convex hull.



1. Find the point min-x, and another point max-x.
2. Join these two points with a line L that divides the shape into two parts.
3. Find the point P outwards with the maximum distance from L.
4. Join P with min-x and max-x, ignore all other points inside the triangle (convex hull).
5. Repeat step 3 and step 4 on the two lines formed by the two new sides of the triangle until all points are inside the convex hull.



A nice example of QuickHull algorithm in Python, which randomly generates a set of points and finds the convex hull can be found at AnantJoshiCZ/QuickHull.git.

A convex polytope is a convex hull of a finite set of points in $\mathbb{R}^n$.

A convex polytope is a convex hull of a finite set of points in $\mathbb{R}^n$.
A hyperplane is a subspace of co-dimension 1.

$$a_1x_1 + a_2x_2 + \ldots + a_nx_n = b$$

A convex polytope is a convex hull of a finite set of points in $\mathbb{R}^n$.
A hyperplane is a subspace of co-dimension 1.

$$a_1x_1 + a_2x_2 + \ldots + a_nx_n = b$$

A face is an intersection of convex polytope with supporting hyperplane.

A convex polytope is a convex hull of a finite set of points in $\mathbb{R}^n$.
A hyperplane is a subspace of co-dimension 1.

$$a_1x_1 + a_2x_2 + \ldots + a_nx_n = b$$

A face is an intersection of convex polytope with supporting hyperplane.
An $(n-1)$-dimensional face of a polytope is called a facet and $(n-2)$-dimensional face is called a ridge.

The main geometric operation used in the quickhull algorithm is the signed distance from a point $\vec{p} \in \mathbb{R}^n$ to a hyperplane:

$$\frac{\vec{n} \cdot \vec{p} - b}{||\vec{n}||}$$

The main geometric operation used in the quickhull algorithm is the signed distance from a point $\vec{p} \in \mathbb{R}^n$ to a hyperplane:

$$\frac{\vec{n} \cdot \vec{p} - b}{||\vec{n}||}$$

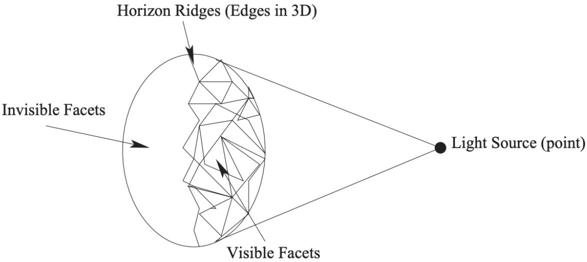
Normal vector $\vec{n} = (a_1, \ldots, a_n)$. Let, $\vec{x_0}$ be a point in the hyperplane then, $\vec{q} = \vec{p} - \vec{x_0}$. Hence, $proj_{\vec{n}}\vec{q} = \frac{\vec{n} \cdot \vec{q}}{||\vec{n}||} = \frac{\vec{n} \cdot (\vec{p} - \vec{x_0})}{||\vec{n}||} = \frac{\vec{n} \cdot \vec{p} - \vec{n} \cdot \vec{x_0}}{||\vec{n}||} = \frac{\vec{n} \cdot \vec{p} - b}{||\vec{n}||}$, as $\vec{n} \cdot (\vec{x} - \vec{x_0}) = 0 \iff \vec{n} \cdot x = \vec{n} \cdot \vec{x_0} = b$.

The main geometric operation used in the quickhull algorithm is the signed distance from a point $\vec{p} \in \mathbb{R}^n$ to a hyperplane:

$$\frac{\vec{n} \cdot \vec{p} - b}{||\vec{n}||}$$

Normal vector $\vec{n} = (a_1, \ldots, a_n)$. Let, $\vec{x_0}$ be a point in the hyperplane then, $\vec{q} = \vec{p} - \vec{x_0}$. Hence, $proj_{\vec{n}}\vec{q} = \frac{\vec{n} \cdot \vec{q}}{||\vec{n}||} = \frac{\vec{n} \cdot (\vec{p} - \vec{x_0})}{||\vec{n}||} = \frac{\vec{n} \cdot \vec{p} - \vec{n} \cdot \vec{x_0}}{||\vec{n}||} = \frac{\vec{n} \cdot \vec{p} - b}{||\vec{n}||}$, as $\vec{n} \cdot (\vec{x} - \vec{x_0}) = 0 \iff \vec{n} \cdot x = \vec{n} \cdot \vec{x_0} = b$.

If the distance is positive, we say the point is above the hyperplane, else the point is below the hyperplane.

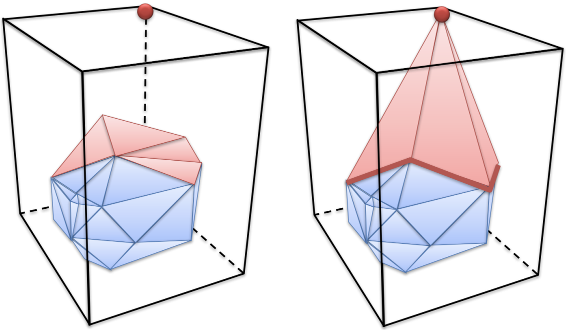


2005 - Agarwal

Theorem (Simplified Beneath-Beyond, 1961 - Grünbaum)

If S is a convex hull in $\mathbb{R}^n$ and P be a point in $\mathbb{R}^n \setminus S$, then F is a facet of $\textbf{conv}(P \cup S)$ *if and only if*:

- *F is a facet of S, and P is below F, or*
- *F is not a facet of S, and its vertices are P and the vertices of the ridges of S with one facet below P and other facet above P.*



2010 - Farag

1. Create initial convex hull Hull by choosing $(n+1)$-points which do not share a plane or a hyperplane.
2. For each facet F in Hull, find all unassigned points above it and add them to F's outside set F^c.
3. For each F with non-empty F^c,
   1. Find the point P with maximum distance from F and add it to Fac.
   2. Create a visible set V and initialize it to F.
   3. The boundary of V froms the set of horizon ridges H.
   4. Add the facets created from P and H to Fac.
   5. Delete all the points of F^c which are created from facets of V.
   6. For each Fac\Hull go to step 3.2
   7. Delete the internal facets from Hull and add Fac\Hull to Hull.
4. Repeat step 2 and step 3 until all points are points are inside the convex hull.

A nice visual implementation of QuickHull algorithm in 3-dimension can be found at carolhmj/quickhull-3d.git.

Lemma

Every extreme point of the input is added to the convex hull irrespective of which outside set it has been assigned to.

Lemma

Every extreme point of the input is added to the convex hull irrespective of which outside set it has been assigned to.

Prove by contradiction: assume an extreme point P which does not belong to any outisde set and hence do not belong to Hull. Since, P is an extreme point it must belong to atleast one outside set for F in the initial Hull. By assumption, there is a point Q with P in its visible outside sets but not in its new outside sets. By definition P is above a visible facet and below all new visible facets for Q, which implies that P is in Hull and hence not an extreme point.

Theorem

The quickhull algorithm produces a convex hull of a set of points in $\mathbb{R}^n$.

Theorem

The quickhull algorithm produces a convex hull of a set of points in $\mathbb{R}^n$.

The algorithm begins with a convex hull of $(n+1)$-points and partitions points into outside sets and chooses the point with maximum distance. By last lemma and the Simplified Beneath-Beyond theorem quickhull algorithm produces convex hull of processed points.

Grahm Scan	$\mathcal{O}(nlogn)$
sorted points	$\mathcal{O}(n)$
Jarvis March or Gift wrapping	$\mathcal{O}(nh)$
all points on convex hull	$\mathcal{O}(n^2)$
Quickhull for dimension ≤ 3	$\mathcal{O}(nlogh)$
for dimension $d > 3$ [1996 - Klee]	$\mathcal{O}(\frac{n}{h}(\mathcal{O}(h^{\lfloor d/2 \rfloor}/\lfloor d/2 \rfloor!)))$
Kirkpatrick–Seidel algorithm	$\mathcal{O}(nlogh)$
Chan's algorithm	$\mathcal{O}(nlogh)$

- Image Processing: Quickhull and Chan's algorithm because they can handle large point sets, allowing for the determination of object boundaries or region of interests in images.

- Image Processing: Quickhull and Chan's algorithm because they can handle large point sets, allowing for the determination of object boundaries or region of interests in images.
- Robotics and Motion Planning: The Quickhull algorithm, due to its efficiency and ability to handle high-dimensional point sets, is commonly employed in path planning and collision avoidance.

- Image Processing: Quickhull and Chan's algorithm because they can handle large point sets, allowing for the determination of object boundaries or region of interests in images.
- Robotics and Motion Planning: The Quickhull algorithm, due to its efficiency and ability to handle high-dimensional point sets, is commonly employed in path planning and collision avoidance.
- Geographic Information Systems (GIS): Graham's scan or Quickhull to compute the convex hull of geographic point data.

- Image Processing: Quickhull and Chan's algorithm because they can handle large point sets, allowing for the determination of object boundaries or region of interests in images.
- Robotics and Motion Planning: The Quickhull algorithm, due to its efficiency and ability to handle high-dimensional point sets, is commonly employed in path planning and collision avoidance.
- Geographic Information Systems (GIS): Graham's scan or Quickhull to compute the convex hull of geographic point data.
- Collision Detection: Both the Quickhull algorithm and the Jarvis march algorithm are commonly used due to their efficiency and ability to handle 3D point sets.

- Image Processing: Quickhull and Chan's algorithm because they can handle large point sets, allowing for the determination of object boundaries or region of interests in images.
- Robotics and Motion Planning: The Quickhull algorithm, due to its efficiency and ability to handle high-dimensional point sets, is commonly employed in path planning and collision avoidance.
- Geographic Information Systems (GIS): Graham's scan or Quickhull to compute the convex hull of geographic point data.
- Collision Detection: Both the Quickhull algorithm and the Jarvis march algorithm are commonly used due to their efficiency and ability to handle 3D point sets.
- Data Visualization: Graham's scan, Quickhull, and Chan's algorithm are popular choices due to their efficiency and ease of implementation.