A Subnet Convex Algorithm for Hull Aggregation

PROJECT TITLE: 3D Geometry Hull Calculations
SEP Group: HULL2
By: Fulin Kuang (a1639080)

Research scope: It is essential to understand the problem before designing and implementing a solution to the problem. Since it's CV related project, and there are a lot of detailed math behind this, we should figure out the concept there, because Maptek has provided the Hull calculations function, I decided to focus on the second part – Hull Aggregation at first. In week 4, I read a paper regarding one of the methods in hull aggregation, and below is the idea of this algorithm. Since it is not a formal report or paper, I didn't use Latex for the format to save time.

Paper reading: Subnet convex hull merging algorithm for reconstructing digital terrain models
Reference: Jitao, Z. H. E. N. G., et al. "Subnet convex hull merging algorithm for reconstructing digital terrain models." *Journal of Tsinghua University (Science and Technology)* 55.8 (2015): 895-899.

<u>Pseudocode of this algorithm: refer to last page, but you need to go through the basic concept to have a brief idea of this problem to understand the algorithm</u>

Basic Concept:
Divide and conquer method to reconstruct digital terrain. First, the complete terrain point cloud must be divided into a number of smaller point cloud subsets, and then a sub-triangular network is established for the divided point cloud subsets. The outer boundary of the sub-triangular network is formed A polygon is a convex polygon, and this convex polygon is called the convex hull of the sub-triangulation. For example, the convex polygon formed by two thick solid lines in Fig. 1 is the convex hull of two subnets. Usually, a convex hull can be represented by an ordered sequence of the set of vertices constituting the convex hull (without loss of generality, this article takes counterclockwise order as an example). Therefore, the two convex hulls in Figure 1 can be represented as two sets $\mathrm{A} = \{\mathrm{a}_1, \mathrm{a}_2, \ldots, \mathrm{a}_7\}$ and $\mathrm{B} = \{\mathrm{b}_1, \mathrm{b}_2, \ldots, \mathrm{b}_7\}$. The direction of each side of the convex hull is also pointing to the counterclockwise direction of the convex hull. The convex hull edge vector $\underset{\mathrm{a}_7\mathrm{a}_1}{\longrightarrow}$ and $\underset{\mathrm{b}_7\mathrm{b}_1}{\longrightarrow}$ before merging in Figure 1, and the merged edge vector $\underset{\mathrm{a}_6\mathrm{b}_6}{\longrightarrow}$. For any vertex a_i in the convex hull, the edge vector with a_i as the tail is called the front edge of a_i, and the edge vector with a_i as the head is called the back edge of a_i.

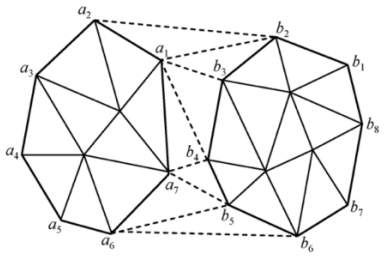


Figure 1: Schematic diagram of subnet convex hull and convex hull merge

If a convex hull does not have any more than 3 vertices collinear, it is called a standard convex hull, otherwise it is a general convex hull.

A support line of the convex hull refers to a straight line that passes through a certain vertex of the convex hull, and satisfies that the convex hull is completely on one side of the straight line. Two convex hulls that do not intersect or overlap have a common support line, which is called a common support line (or a common tangent line). The corresponding vertex is called the support point (or the common tangent point). In Figure 1, $\underset{a_6b_6}{\longrightarrow}$ and $\underset{b_2a_2}{\longrightarrow}$ are the common support lines of the two convex hulls A and B. Among them, a_2, a_6, b_2 and b_6 are called the common support points of the two convex hulls.

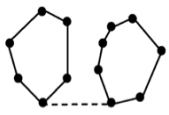
Two non-overlapping convex hulls are merged to generate a new convex hull, and the two common support lines become the new convex hull edge after merging. In figure 1, the combined convex hull of convex hulls A and B can be represented by the set $H = \{b_1, b_2, a_2, a_3, a_4, a_5, a_6, b_6, b_7, b_8\}$. Two non-overlapping convex hulls at any position can always be transformed into a left-right positional relationship through rotation. Without loss of generality, for the convenience of describing the problem, this article only considers the problem of merging the left and right convex hulls, and the result is easy to generalize to any positional relationship.

Convex hull merging needs to correctly find the common support lines and corresponding support points of the two convex hulls participating in the merging.

Convex hull Aggregation:
The steps of merging and merging include:

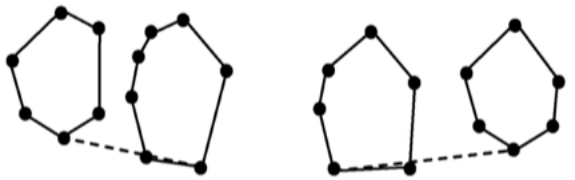
1. Find the common support lines and support points of the two convex hulls participating in the merging;
2. Correctly connect the vertices inside the two convex hulls between the common support points to generate a reasonable triangle;
3. Update the merge Convex edge

Relative position relationship between convex hulls:
After analyzing and splitting a large amount of data, the convex hulls of the two subnets participating in the merger can be summarized into six situations relative to the common support line.

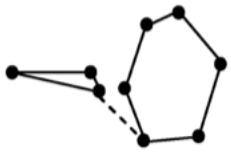

The vertex with the smallest y coordinate of the two convex hulls is the lower support point, no adjustment is required


At least one of the two convex hulls has multiple collinear points and coincides with the common support line. At this time, if multiple collinear points appear in

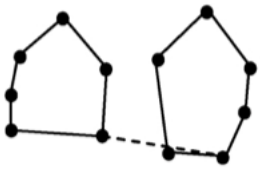
the convex hull on the left, the rightmost vertex of the collinear vertices should be taken; if multiple collinear points appear in the convex hull on the right, the leftmost vertex of the collinear vertices should be taken



Among the two convex hulls, the lowest point of one of the convex hulls is the support point, and the lowest point of the other convex hull is not the support point, and usually needs to be adjusted through judgment



The size of the two convex hulls is very different, sometimes resulting in that there is no edge that satisfies the other convex hull completely on one side of the small convex hull.



The lowest points of the two convex hulls involved in the merging are all non-support points, and usually need to be adjusted multiple times in the two convex hulls in order to finally determine the common support point and the common support line

The points with the smallest and largest x-coordinate values in the convex hull are called the leftmost and rightmost points, respectively. When there are multiple vertices with the same smallest or largest x value, the point with the smallest y value is selected. In Figure 2, a_l, b_l and a_r, b_r are used to represent the leftmost and rightmost points of the two convex hulls, respectively. The counterclockwise sequence of points from the leftmost point to the rightmost point is called the lower half chain of the convex hull, as shown in the figure The two half-chains in Figure 2 are $\{a_l, a_{l+1}, \ldots, a_{r-1}, a_r\}$ and $\{b_l, b_{l+1}, \ldots, b_{r-1}, b_r\}$ and In the same way, the upper chain can be defined.

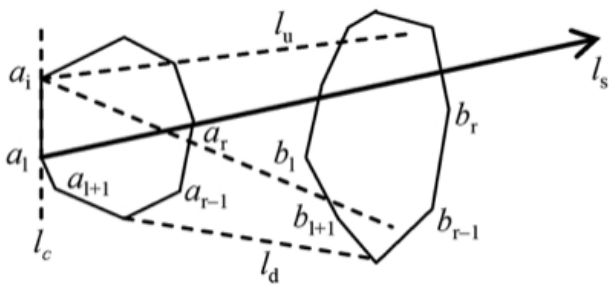


Figure 2

Without loss of generality, only the detection of the following support points as an example can easily be extended to the upper support point. For the convenience of description, $\mathrm{Prj}_{\mathrm{u}}^{\rightarrow}\mathrm{a_i}$ is used to represent the projection position of the point $\mathrm{a_i}$ on the vector u.

Two points are projected on the same vector, and the forward projection position along the vector direction is greater than the backward projection position; if the two projection positions overlap, they are defined as equal. As shown in Figure 3, it can be expressed as $Prj_{\mathrm{u}}^{\rightarrow}\mathrm{a_i} > Prj_{\mathrm{u}}^{\rightarrow}\mathrm{b_i}$

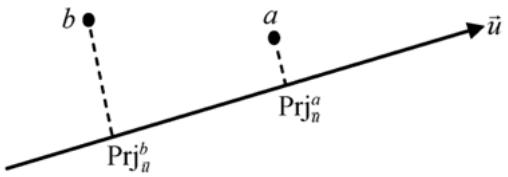


Figure 3

First, select the rightmost point in the lower half of the two convex hulls participating in the merging as the initial point, connect these two points as the lower common support line, and then alternately check whether the left and right convex hulls are both on the left side of the support line. The specific detection method and process are shown in Figure 4.

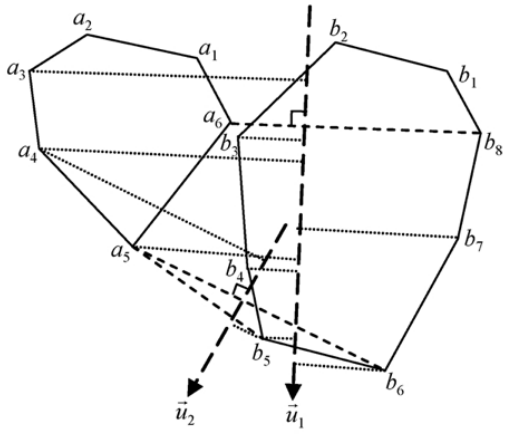


Figure 4

Select the initial support points $\mathrm{a_6}$ and $\mathrm{b_8}$, connect $\mathrm{a_6}$ and $\mathrm{b_8}$, and turn the vector $\mathrm{a_6}$ to the right by 90° to generate a vector perpendicular /overrightarrow$\mathrm{u_1}$ to it, each vertex of the lower half chain of the 2 convex hulls participating in the merge is directed / overrightarrow$\mathrm{u_1}$ Projection, find the vertices with the largest projection positions respectively, that is, satisfy:

$$\mathrm{Pr}\, j_u^{a5} = \max \mathrm{Pr}\, j_u^{a_i}, i = 3, 4, 5, 6;$$

$$\mathrm{Pr}\, j_u^{a6} = \max \mathrm{Pr}\, j_u^{b_j}, i = 3, 4, 5, 6, 7, 8.$$

Determine whether the vertex with the largest projection position in the current $\mathrm{u_1} \rightarrow$ direction in the lower half of the chain is the same as the currently selected support point. If there is a difference, it will be updated to the vertex with the largest current projection position. Repeat the above detection process until the two supporting points are the

maximum projection position points after a certain detection, and the current supporting point is the final supporting point. In the same way, the upper support point can be detected.

Obviously, for two disjoint convex hulls, there must be two upper and lower common support lines and four corresponding support points. Therefore, the common support line and support points can always be determined after a finite step detection.

Subnet merge:
The four support points of the convex hull of the two subnets are detected, and the two need to be merged into one subnet, while ensuring that the edge is still a convex hull.

Arrange the inner vertex sequence between the two convex hull support points in order, as shown in Figure 5, which are $\{a_d, a_{d+1}, \ldots, a_u\}$ and $\{b_d, b_{d-1}, \ldots, b_u\}$, and then from any common The support line begins, and a new triangle is gradually formed until another common support line. Every time a new triangle is generated, its rationality needs to be judged, that is, it should be ensured that the triangles cannot overlap or intersect. Usually, the rationality can be judged based on whether the newly generated triangle contains other vertices.

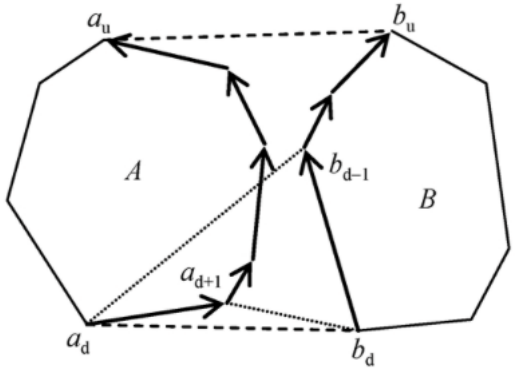


Figure 5

Suppose the current side is $\overline{a_d b_d}$The adjacent point a_{d+1} of the convex hull on the left can be taken to form $\Delta a_d b_d a_{d+1}$. At this time, the adjacent point b_{d-1} of the convex hull on the right needs to be judged. Whether it is in the triangle; It is also possible to take the adjacent point b_{d-1} of the convex hull on the right to form$\Delta a_d b_d a_{d-1}$, and it is also necessary to determine whether the adjacent point a_{d+1} of the convex hull on the left is inside. If the judgment is reasonable, just choose one if all are reasonable. In practice, when everything is reasonable, the adjacent points can be alternately taken from left and right to generate new triangles, and the emergence of long and narrow triangles can be avoided as much as possible.

The rationality judgment only needs to judge an adjacent point, which is guaranteed by the nature of the convex hull. For example, in Figure 5, the adjacent point b_{d-1} of the convex hull on the right is taken to form $\Delta a_d b_d b_{d-1}$, and only the adjacent point a_{d+1} in the convex hull on the left is judged. Here are the convex hulls A and B on the left and right sides, because $\overline{b_{d-1} b_d}$is the convex edge of B and a_d is outside of B. It is impossible to include the vertices in B in $\Delta a_d b_d b_{d-1}$; if there are vertices in A in $\Delta a_d b_d b_{d-1}$ If a_{d+1} is not in $\Delta a_d b_d b_{d-1}$ and other points are inside, then a certain vertex will inevitably be on the other side of the

convex edge $\overline{a_d a_{d+1}}$, making A not satisfied Convex hull and produce contradictions. Update the convex hull boundary after merging.

Algorithm:

Input:
The input is two subnet convex hulls $A = \{a_1, a_2, \dots, a_n\}$and $B = \{b_1, b_2, \dots, b_n\}$

Output:
The output is the merged subnet and its convex hull $C = \{c_1, c_2, \dots, c_n\}$

Step:

1. Calculate the upper and lower half chains of the two convex hulls A_d, B_d, A_u, B_drespectively;
2. Select the initial lower support points a_d and b_d;
3. The current vector $\underset{a_d b_d}{\longrightarrow}$ rotates 90° to the right to generate a new vector $\underset{u}{\rightarrow}$
4. The vertices of the lower half of the two convex hulls are projected and sorted in the $\underset{u}{\rightarrow}$ direction, and the vertex with the largest projection position is found. If it is different from the current two support points, replace the current lower support point, and go to step 3.
5. Select the initial upper support points a_u and b_u;
6. The current vector $\underset{a_u b_u}{\longrightarrow}$ rotates 90° right to generate a new vector $\underset{u}{\rightarrow}$
7. The vertices of the upper half chain of the two convex hulls are sorted in $\underset{u}{\rightarrow}$ projection. Find the vertex with the largest projection position. If it is different from the current two upper support points, replace the current upper support point, and go to step 5.
8. From the lower support line $\underset{a_d b_d}{\longrightarrow}$ to the upper support line $\underset{a_u b_u}{\longrightarrow}$, merge in turn to generate a new triangle patch
9. Update convex hull edge after merging