Computational Geometry Reference

Convex Hull Algorithms

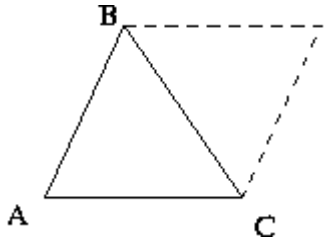
Jarvis March O(n.h), worst case O(n.n)

- Pick a point that you know to be on the convex hull.
- To determine the next point on the hull, loop through all points and find the one that forms the minimum sized anticlockwise angle off the horizontal axis from the previous point.
- Continue until you encounter the first point

Graham Scan O(n log n)

- Find a point you know to be on the convex hull. (Lower y coordinate)
- Sort all other points angularly around this point (anti clockwise), by calculating the angle that each point makes with the x axis (within the range 0 to 360 degrees)
- Add the first two points to the hull.
- For every other point except the last point
- Make it the next point in the convex hull
- Check to see if the angle it forms with the previous two points is greater than 180 degrees
    - As long as the angle formed with the last two points is greater than 180 degrees, remove the previous point
- To add the last point
    - Perform the deletion above,
    - Check to see if the angle the last point forms with the previous point and the first point is greater than 180 degrees or if the angle formed with the last point and the first two points is greater than 180 degrees.
    - If the first case is true, remove the last point, and continue checking with the next-to-last point.
    - If the second case is true, remove the first point and continue checking.
    - Stop when neither case is true.

Area of a Triangle



Heron's Formula

- Area = Sqrt(s(s-a)(s-b)(s-c)), where s = (a+b+c)/2

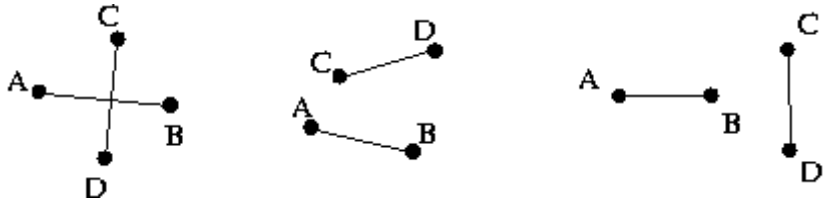
Cross Product / Determinant

To calculate the area of a triangle with vertices (A, B, C), pick a vertex (say A) and create a vector to the other two vertices (let u = B - A, and v = C - A). The area of the triangle (A, B, C) is one half the length of cross product u x v.

Area = (AxBy - AyBx + AyCx - AxCy + BxCy – CxBy)/2

This is the signed area so be sure to take the absolute value!

Line Segment Intersection



The Endpoints of Segment AB must be on opposite sides of segment CD

AND

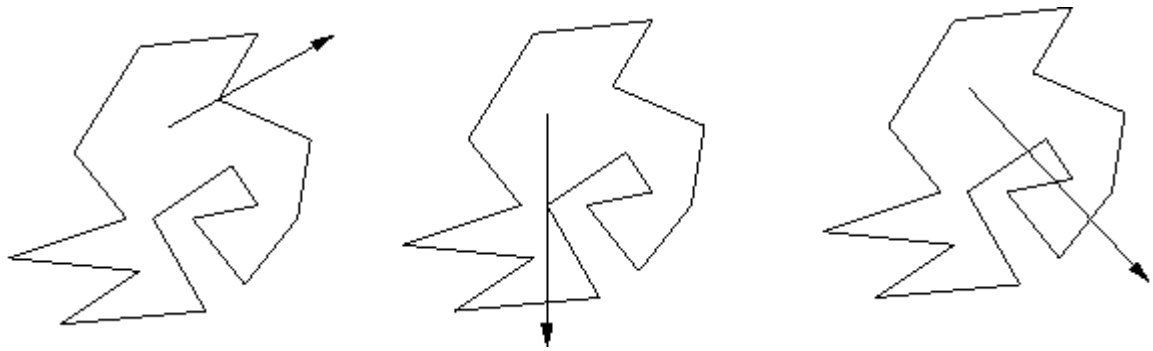
The Endpoints of Segment CD must be on opposite sides of segment AB

Use the signed area of the triangles created to check on side of the segment a point resides.

Line / Line Segment Intersection

This is easier than segment/segment intersection as you only have to test if the endpoints of the segment line on opposite side of the line.

Point in polygon.
To calculate if a point is within a non-convex polygon, make a ray from that point in a random direction and count the number of times it intersects the polygon. If the ray intersects the polygon at a vertex or along an edge, pick a new direction. Otherwise, the point is within the polygon if and only if the ray intersects the polygon an odd number of times.



- This method also extends to three dimensions (and higher), but the restriction on intersection is that it only intersects at faces and not at either a vertex or an edge.

Area of polygon
The area of a polygon with vertices (x 1, y 1), ..., (x n, y n) is equal to the determinant:

$$\begin{vmatrix} x1 & x2 & \dots & xn \\ & & & \\ y1 & y2 & \dots & yn \end{vmatrix} = 2.\text{Area}$$

This formula generalizes to compute d! times the volume of a simplex in d dimensions.

Areas and volumes are signed!

Monte Carlo

This geometric trick is based on randomness. Instead of calculating the probability that something occurs, simulate a random event and calculate the fraction of times it occurs. If enough events are simulated, the difference between these two values becomes very small. This can be helpful to determine something like the area of a figure. Instead of calculating the area directly, determine a bounding box, and throw ``darts'' at the box, and estimate what the probability of hitting the figure is. If this is calculated accurately enough, this can give a good estimate of the actual area. The problem with this method is to get a good relative error (error divided by the actual value) requires a large number of successful events. If the probability of the event occurring is very small, the method does not yield good results.

Example

Finding the area of a polygon, (better methods exist including triangulation and determinants). This is merely to illustrate an application of Monte Carlo methods.

Find the bounding box of the polygon.

Find area of bounding box (A)

Generate a large number (N) of random points inside the bounding box.

Find the number of those points (M) that lies in the polygon, see above.

Area of Polygon ~ (M*A)/N

The more points that are generated the more accurate the answer will tend to be!

Further Reading:

USACO Training Gateway.
Algorithm Design Manual, S Skiena (Available Online).
Vector Primer, www.flipcode.com, www.gamedev.net
Matrix Primer, www.flipcode.com, www.gamedev.net
Various Geometry Texts, www.gamedev.net