Computational Geometry

By Andi Qu

What is Computational Geometry?

Computational geometry is normal geometry but computational.

Commonly tested techniques include:

- Finding the area of a polygon
- Finding the convex hull
- Line sweep
- Finding the closest/farthest pair of points
- Distance queries over a range
- And much more that will take something like 10 slides to list

6 FACTS about Computational Geometry that will SHOCK you

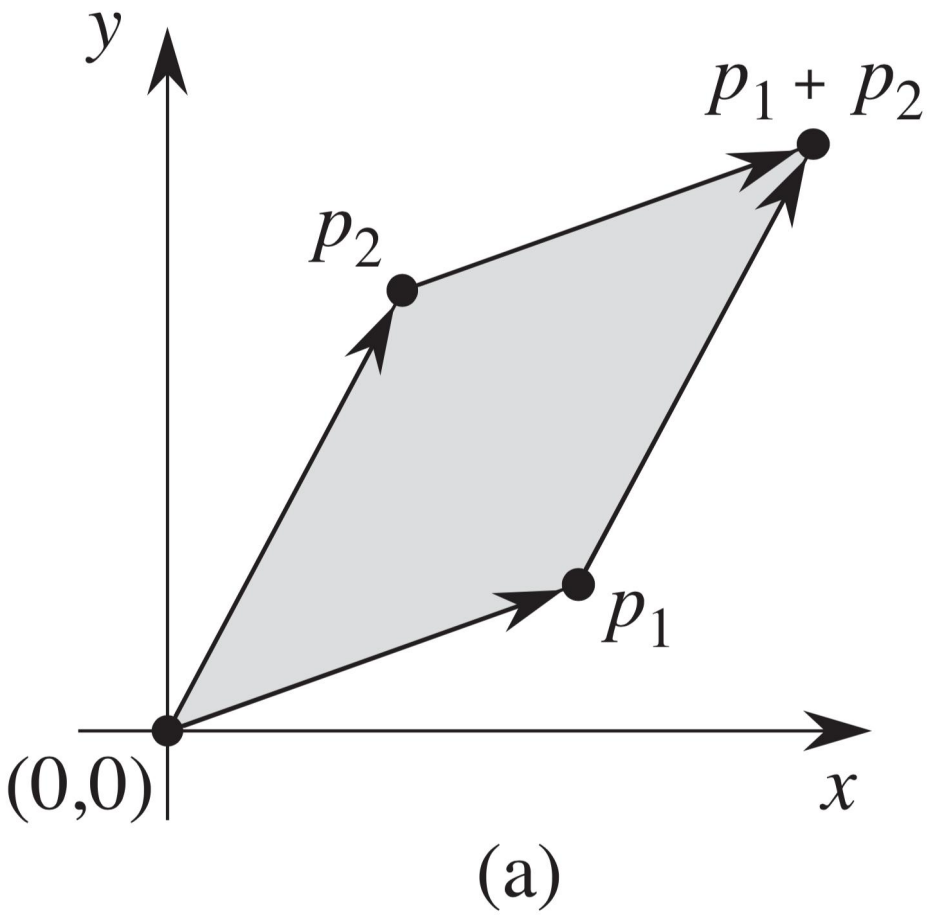
- Computational geometry is literally the worst.
- Computational geometry either voted for Hillary Clinton or Donald Trump, whomever you liked less.
- Computational geometry has no friends.
- Computational geometry prefers T-Series.
- Computational geometry is hideous.
- Computational geometry is the reason why Taariq only got 30/1000 in the 2018 December USACO.

Basic Concepts

Cross Product

If we have 2 vectors p_1 and p_2 with heads (x_1, y_1) and (x_2, y_y) and tails being the origin (0, 0), then he cross product $p_1 \times p_2$ is given as:

$$x_1 \cdot y_2 - x_2 \cdot y_1$$



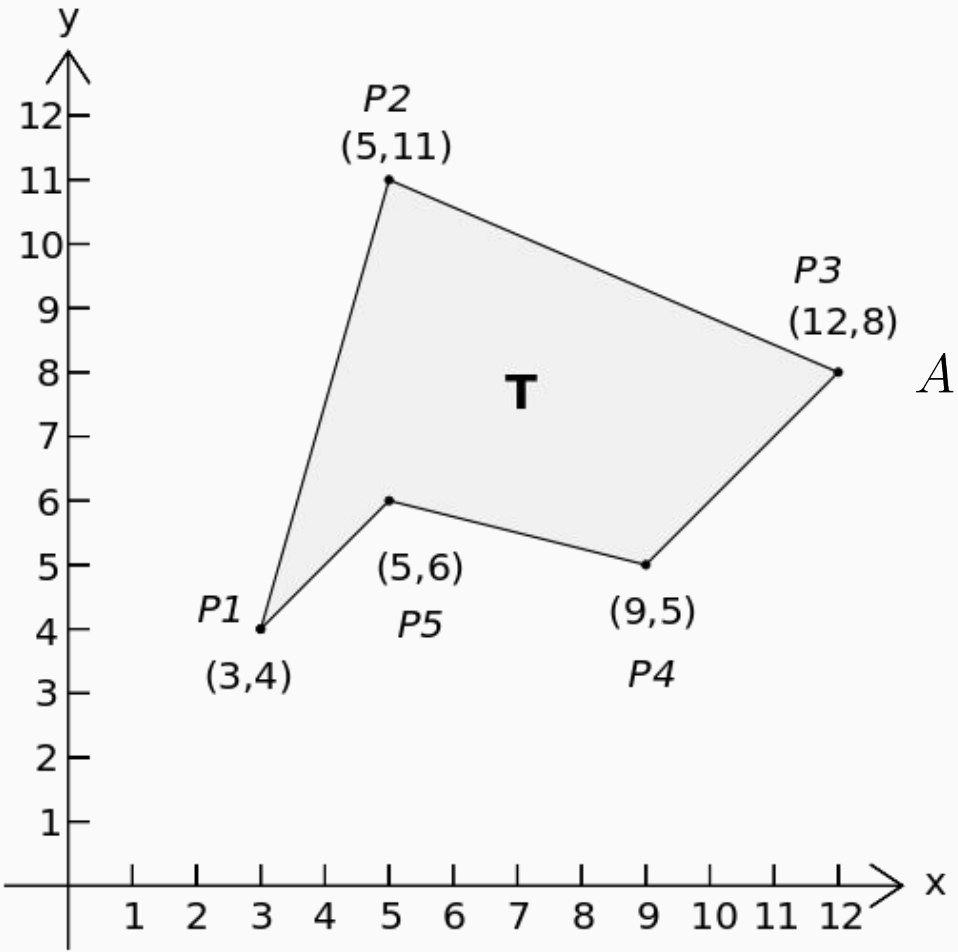
The cross product of 2 vectors gives the directed area of the parallelogram with 2 edges equal to those vectors

Shoestring Formula

An algorithm to determine the area of any simple polygon in O(n) time.

Given a polygon with coordinates (x_i, y_i), the area of the polygon can be calculated from the formula:

$$A = \frac{1}{2}\left|\sum_{i=1}^{n-1} x_i y_{i+1} + x_n y_1 - \sum_{i=1}^{n-1} x_{i+1} y_i + x_1 y_n\right|$$



$$A = \frac{1}{2}|3 \cdot 11 + 5 \cdot 8 + 12 \cdot 5 + 9 \cdot 6 + 5 \cdot 4 - 4 \cdot 5 - 11 \cdot 12 - 8 \cdot 9 - 5 \cdot 5 - 6 \cdot 3|$$

$$= \frac{60}{2} = 30$$

Example Problem

Kattis convexpolygonarea:

Given the vertices of a convex polygon, determine the area.

Solution:

Apply shoestring formula.

Check if Clockwise or Counterclockwise

Notice that if you use the shoestring formula to determine area without the absolute value, you get directed area (because of how cross product works).

If the directed area is **negative**, the points were given in **clockwise** order. Otherwise, they were given in **counterclockwise** order.

Example Problem

Kattis polygonarea:

Given the vertices of a polygon, determine its area and whether the points were given in clockwise order or not.

Solution:

Apply shoestring formula.

Check for Direction of Turn

Similar to checking whether or not the points of a polygon were given in clockwise order, to check whether you turn clockwise or counterclockwise from point *A* to point *B* with origin *C*:

- Calculate the cross product $A \times B$ with origin *C*.
- If the cross product is positive, then *ACB* is a counterclockwise turn.
- Otherwise, it is a clockwise turn.

Check if 2 Segments Intersect

Given segments *AB* and *CD*:

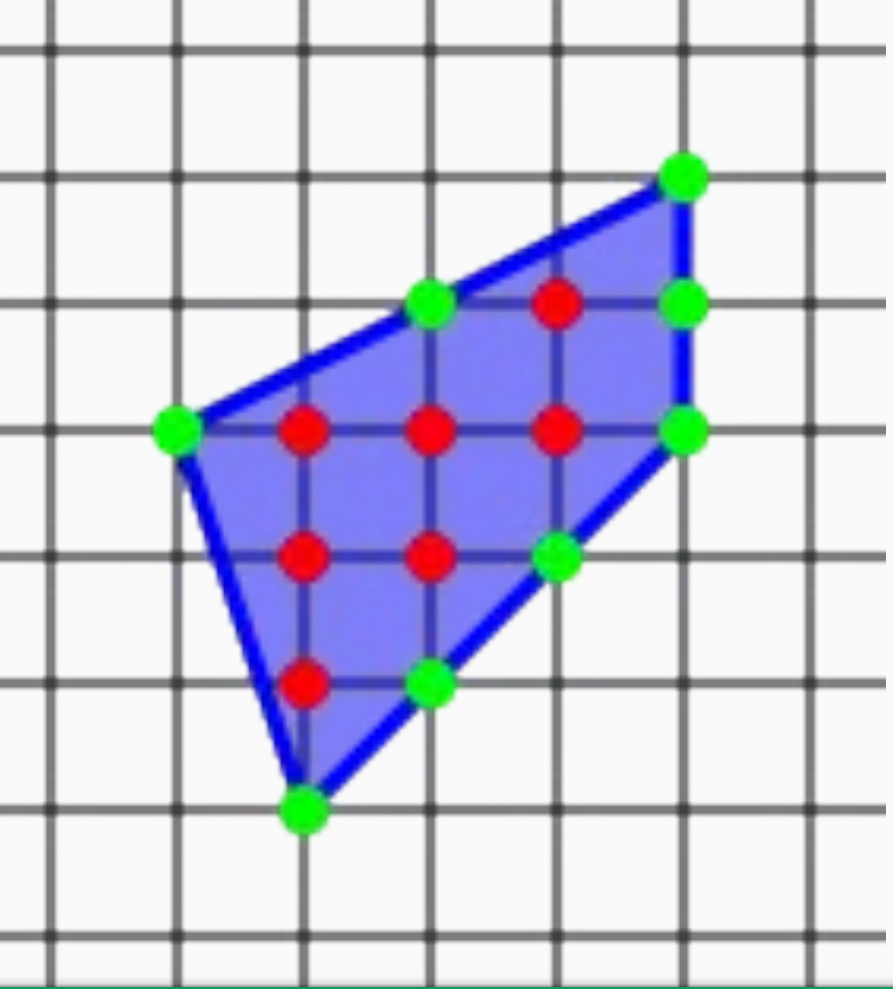
- First, find the directions of the turns of *AB* to *C* and *D* and *CD* to *A* and *B*.
- If the directions for both pairs are opposite, they intersect.
- Else, if the any point of 1 segment is collinear with the other segment and that point lies between the endpoints of the other segment, they intersect.
  - E.g. A_x lies between C_x and D_x, and A_y lies between C_y and D_y.
- Else, they don't intersect.

Pick’s Theorem

Another formula to determine the area of any simple polygon with vertices on grid points on a given plane (e.g. the Cartesian plane).

If *A* is the area, *i* is the number of interior grid points, and *b* is the number of grid points on the boundary (including vertices), then we have:

$$A = i + \frac{b}{2} - 1$$



$$i = 7, b = 8$$

$$\therefore A = i + \frac{b}{2} - 1 = 10$$

Also magic, but not as much as the Shoestring Formula

Example Problem

Polygonal Laser:

Given the vertices of a simple polygon, determine the number of interior points.

Solution:

Left as an exercise to the reader.

ALWAYS CHECK FOR INTEGER OVERFLOW!!

Convex Hulls

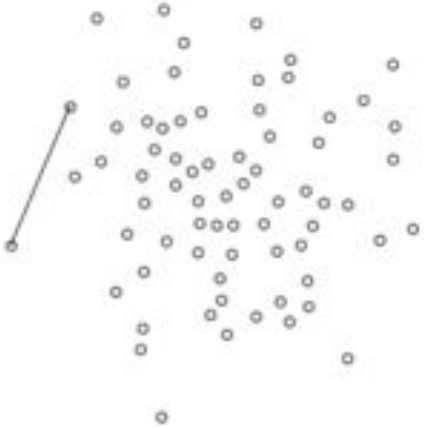
Convex Hull

A convex hull of *P* points is the convex polygon with minimal area that contains all of the points.

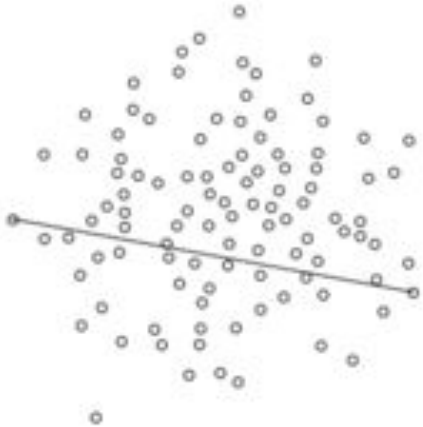
Common algorithms to find the convex hull (ranked in order of superiority):

- Chan's Algorithm (O(n log h))
- Graham Scan (O(n log n))
- Monotone Chain (O(n log n))
- Kirkpatrick–Seidel Algorithm (O(n log h))
- Quickhull (Basically quicksort - average O(n log n); $O(n^2)$ worst case)
- Jarvis' March (O(nh))

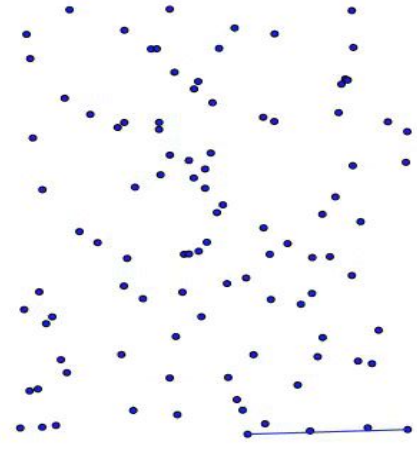
Monotone Chain



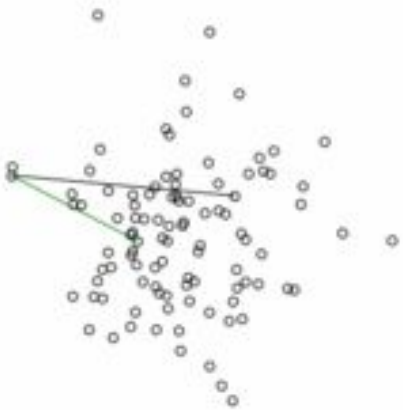
Quickhull



Graham Scan



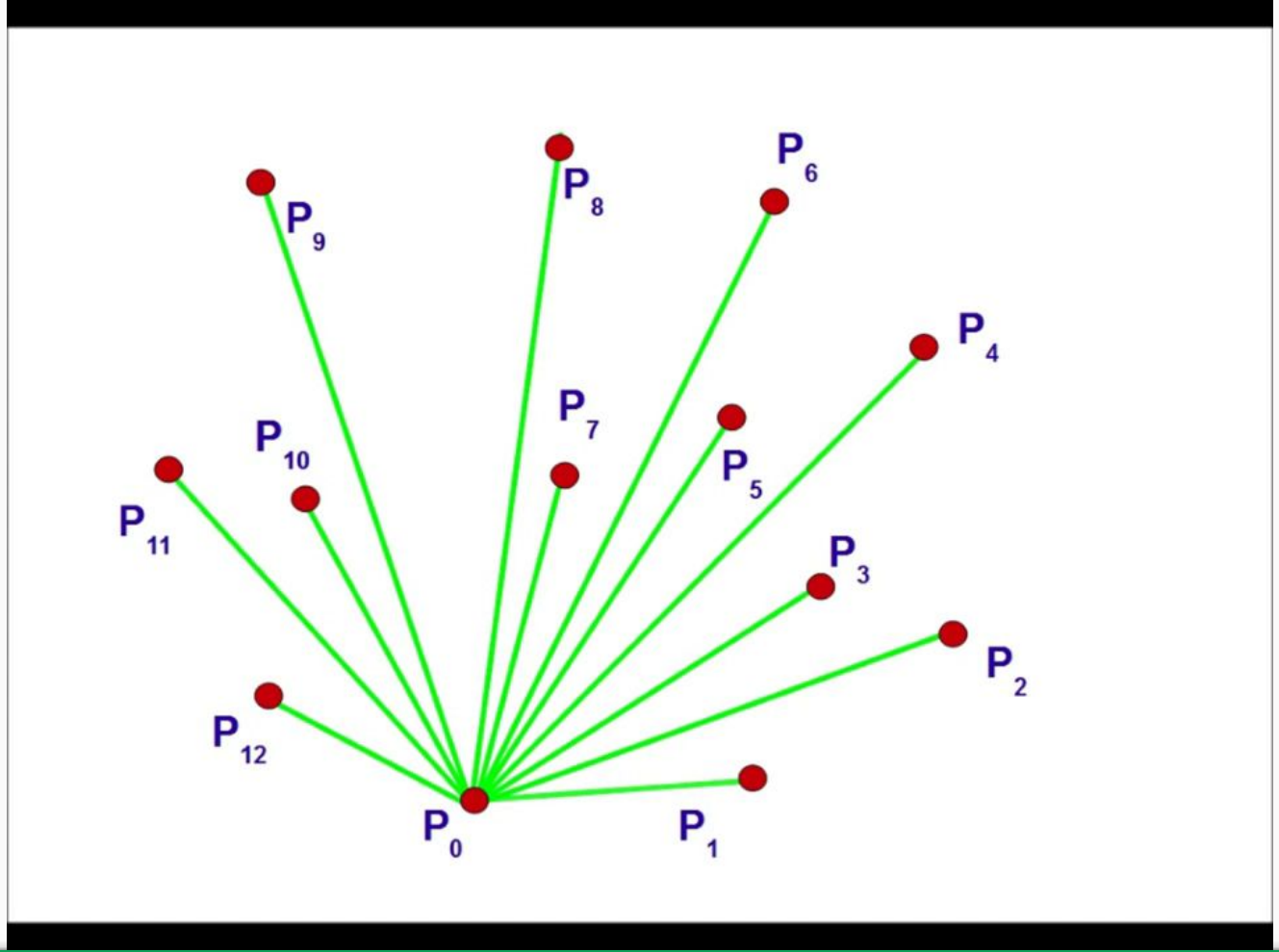
Jarvis' March



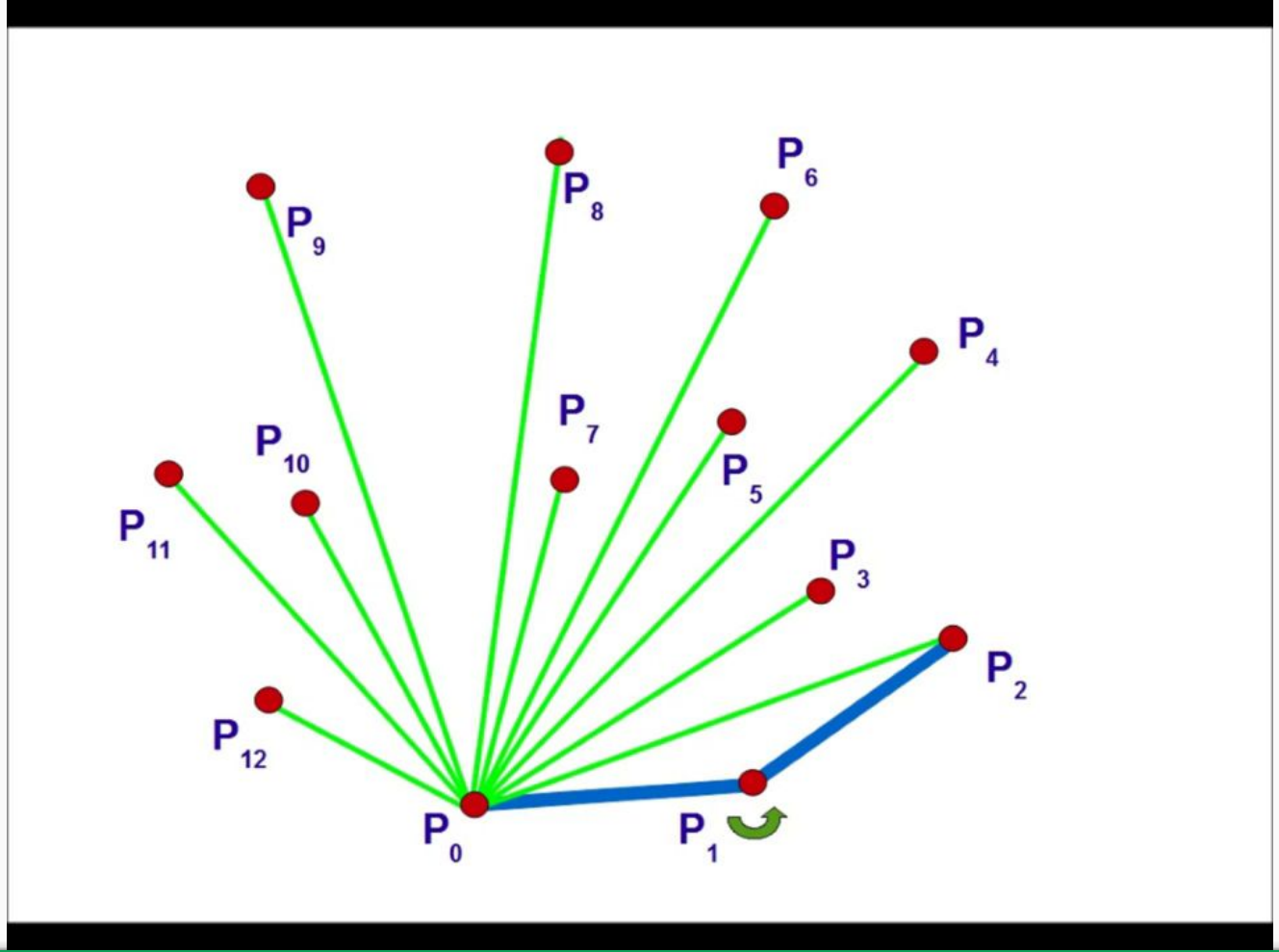
Visualisation of the 4 common convex hull algorithms

Graham Scan

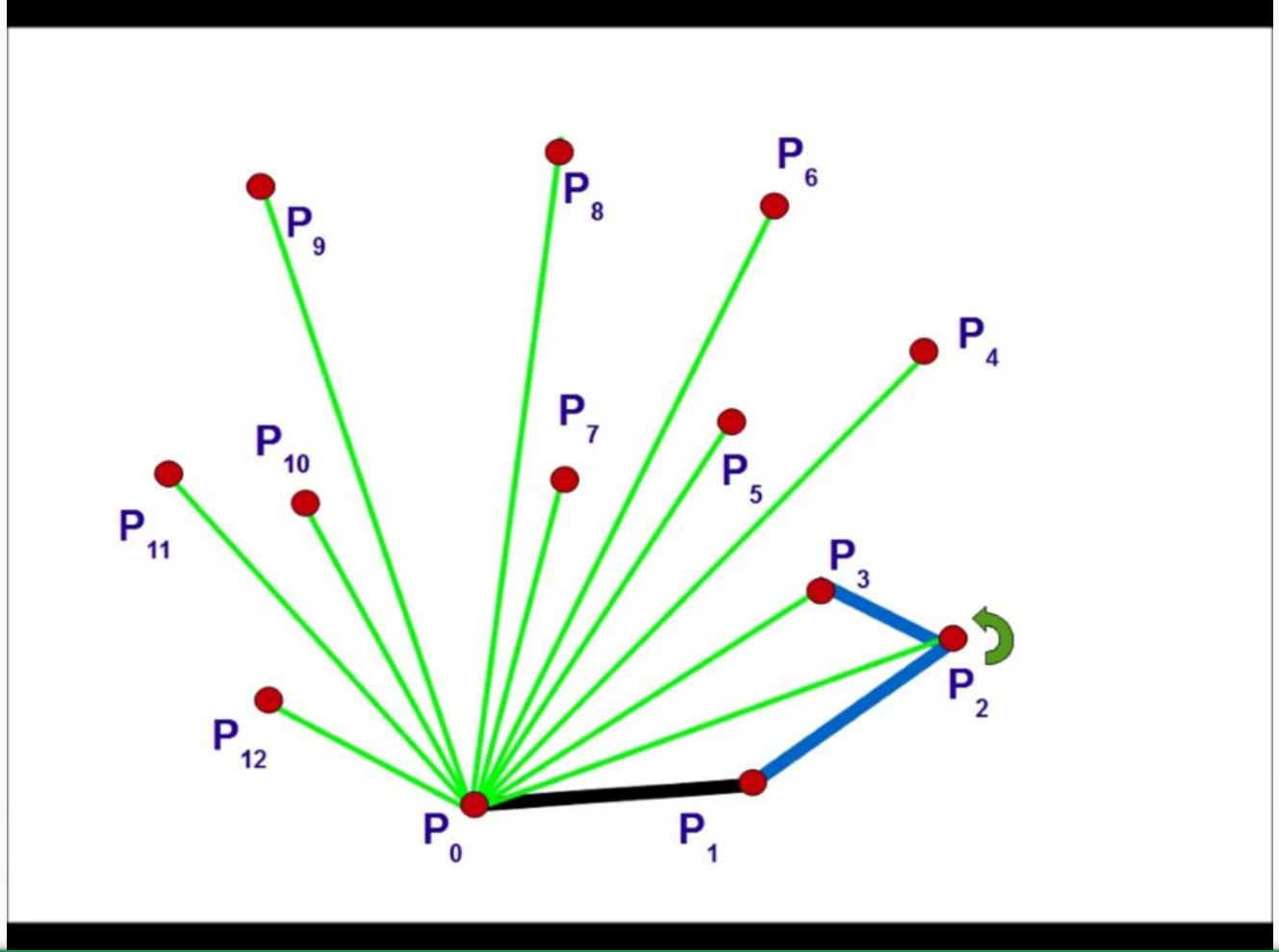
- Initialise an empty stack *S*.
  - This will store the points of the convex hull in counterclockwise order.
- Find the lowest y-coordinate point *P*.
  - If there are multiple, take the leftmost one.
- Sort every other point by the angle to the horizontal that they make with *P*.
  - If there are multiple points with the same angle, keep the farthest one only.
- For each point *Q* in this sorted list:
  - Repeatedly pop the top of the stack until the directed angle *S[-2]S[-1]Q* is clockwise.
  - Add the point to the top of the stack.



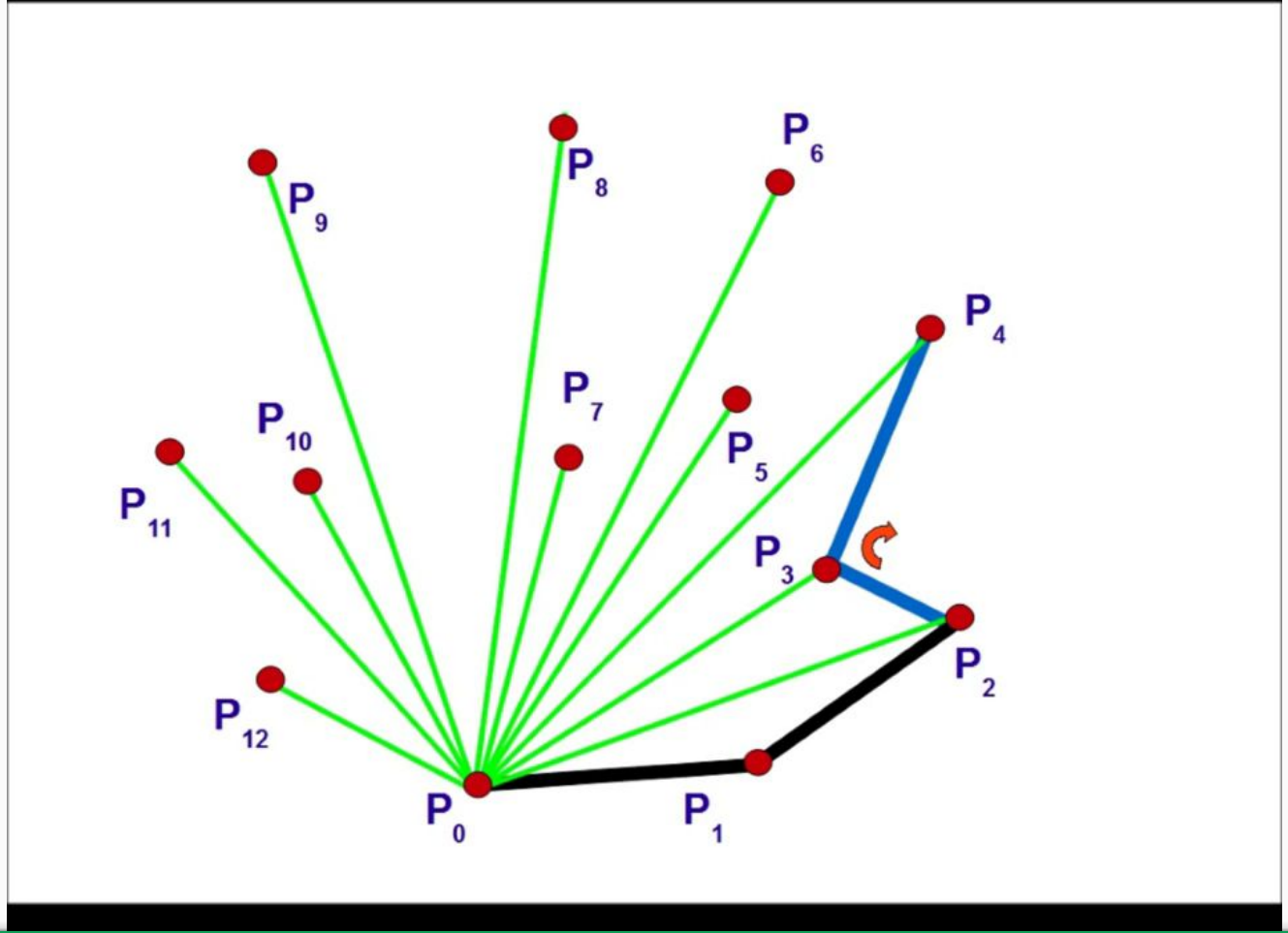
Totally not stolen from Robin



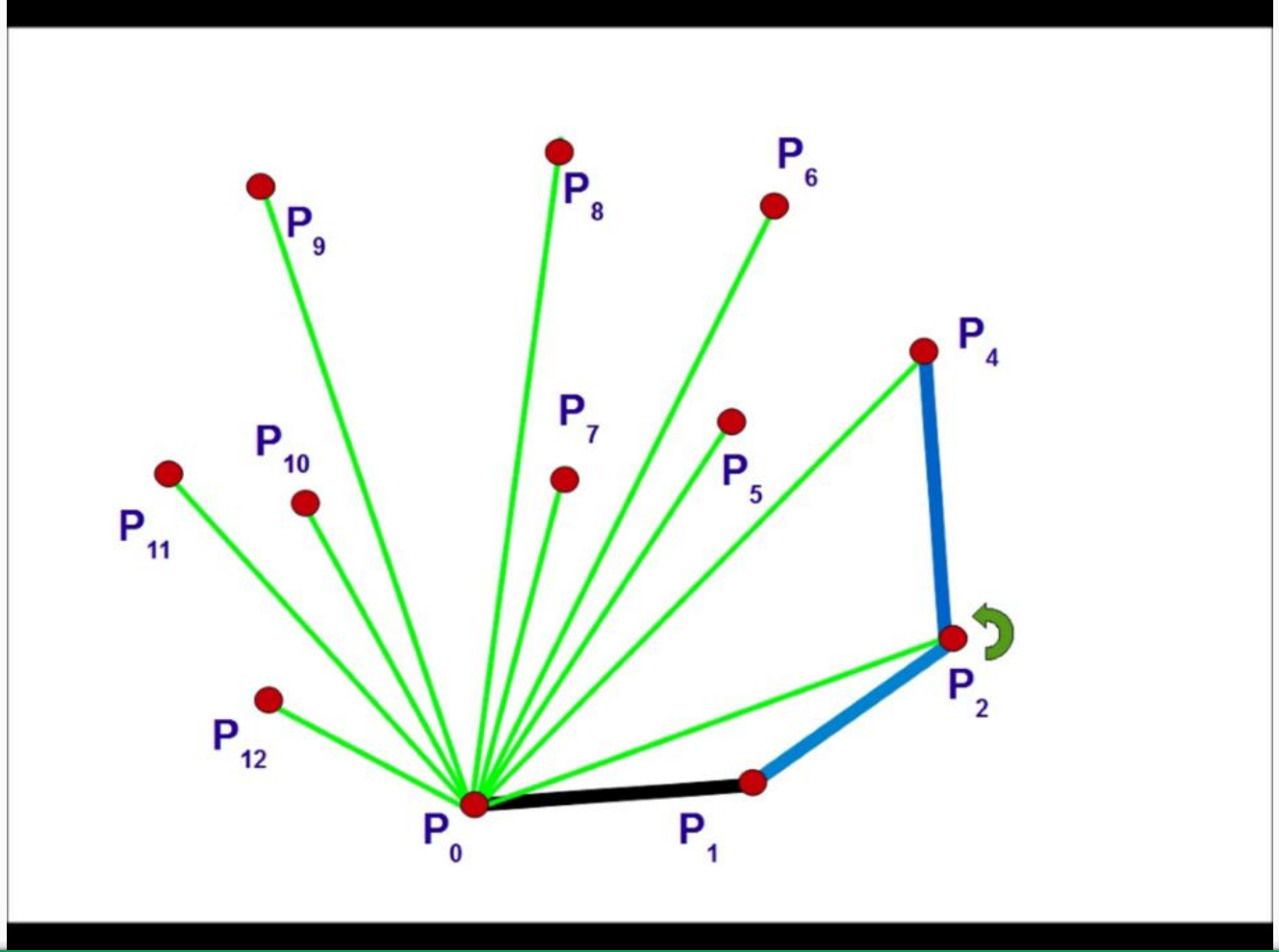
Totally not stolen from Robin



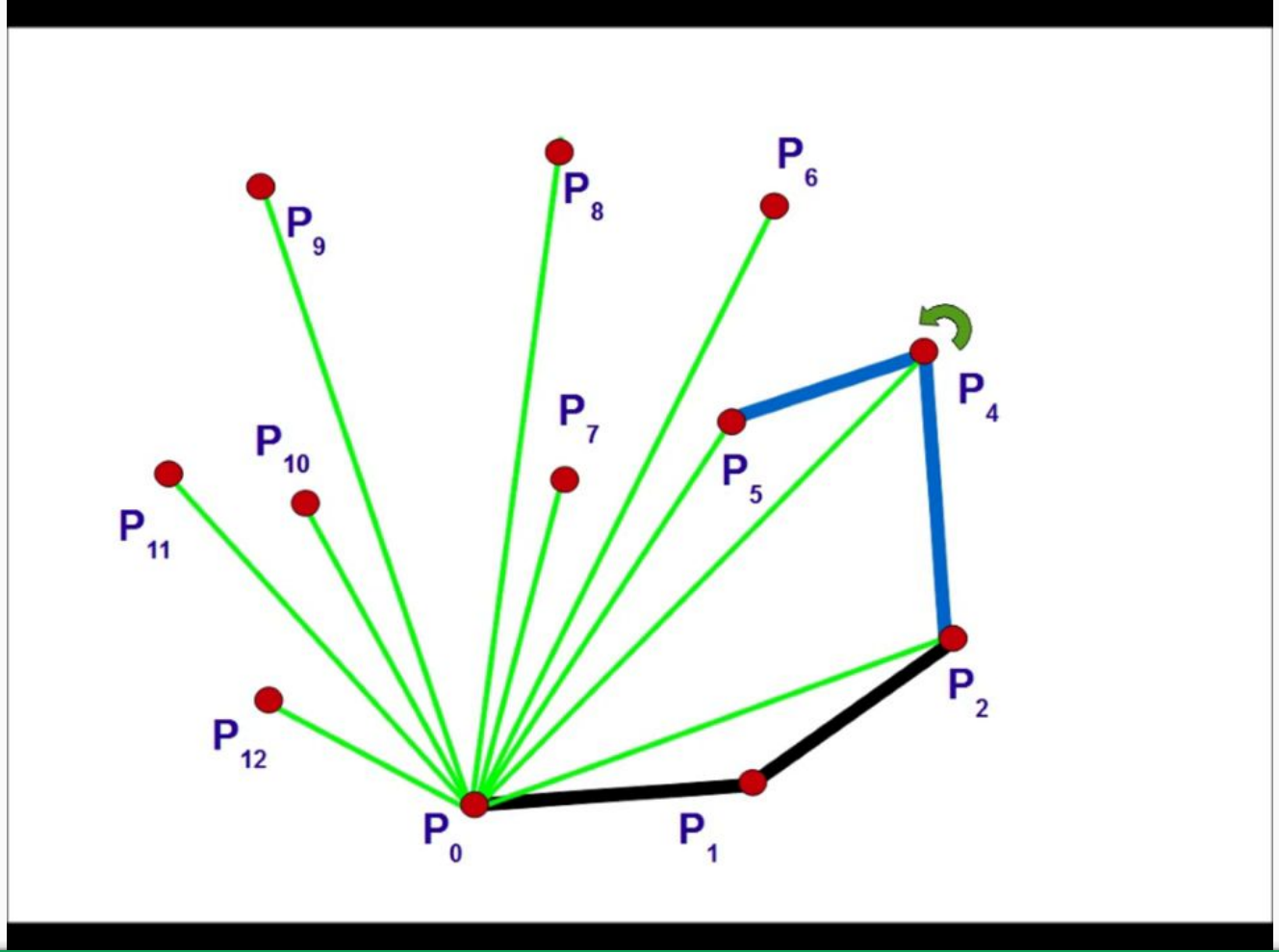
Totally not stolen from Robin



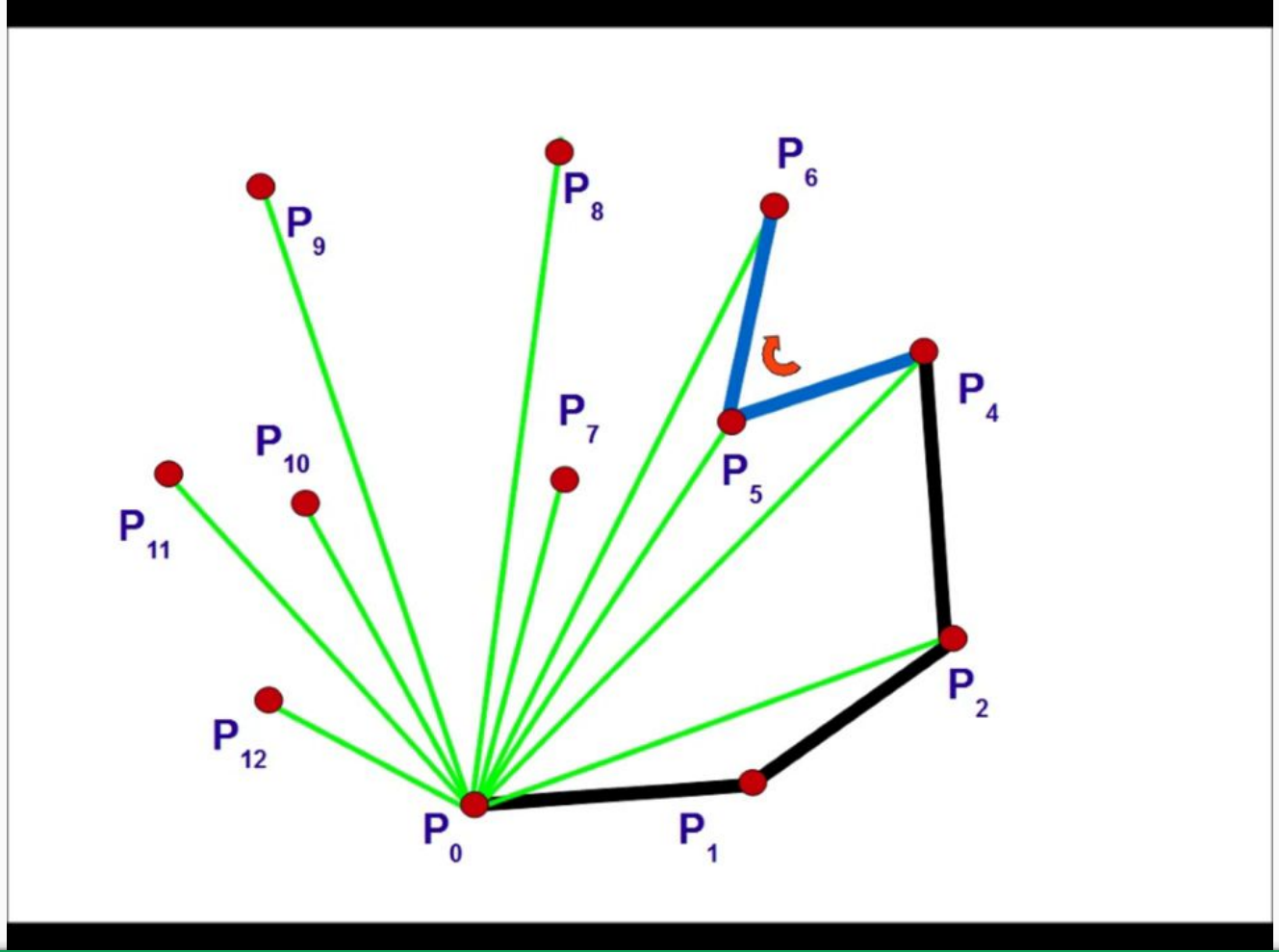
Totally not stolen from Robin



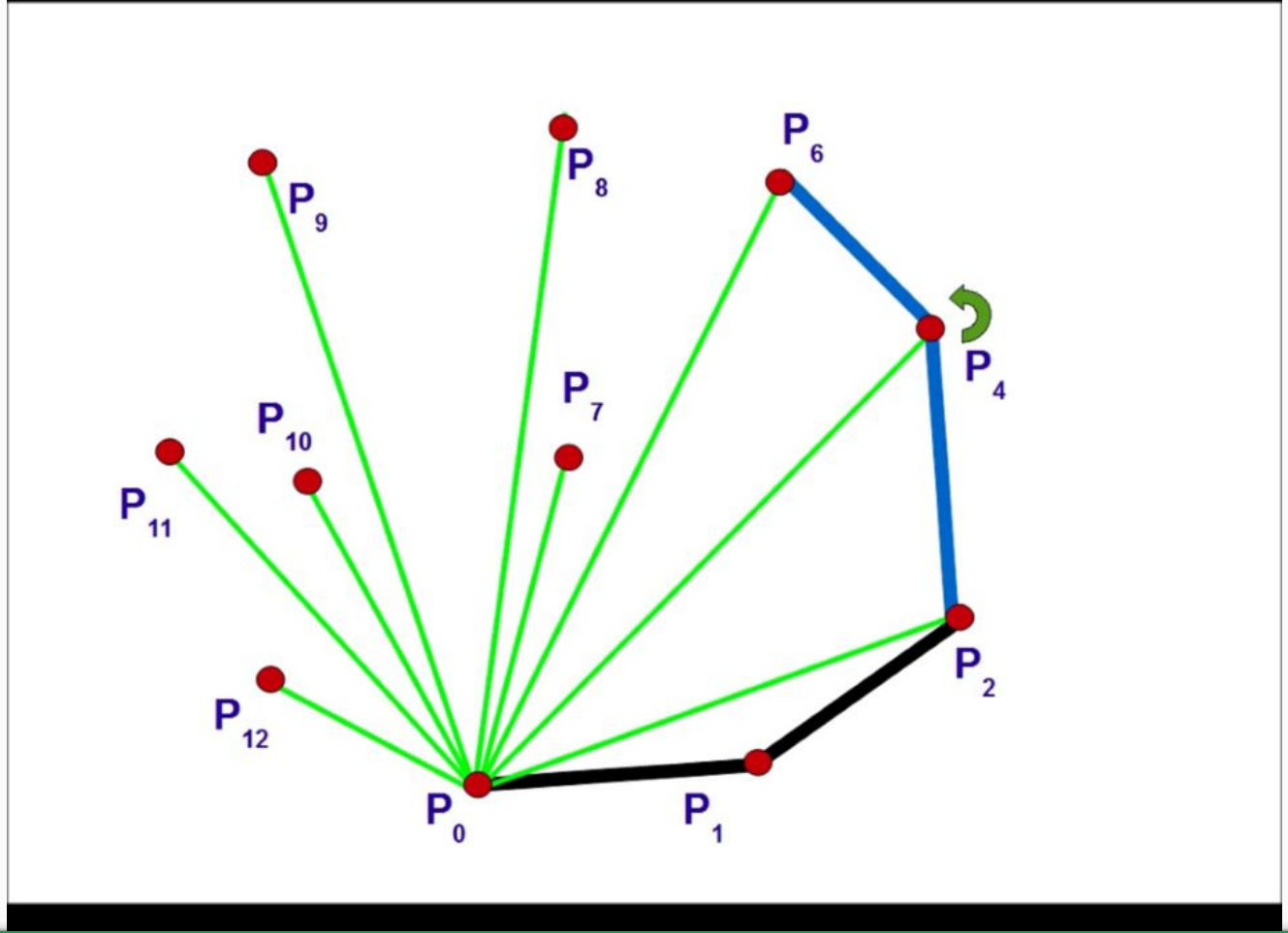
Totally not stolen from Robin



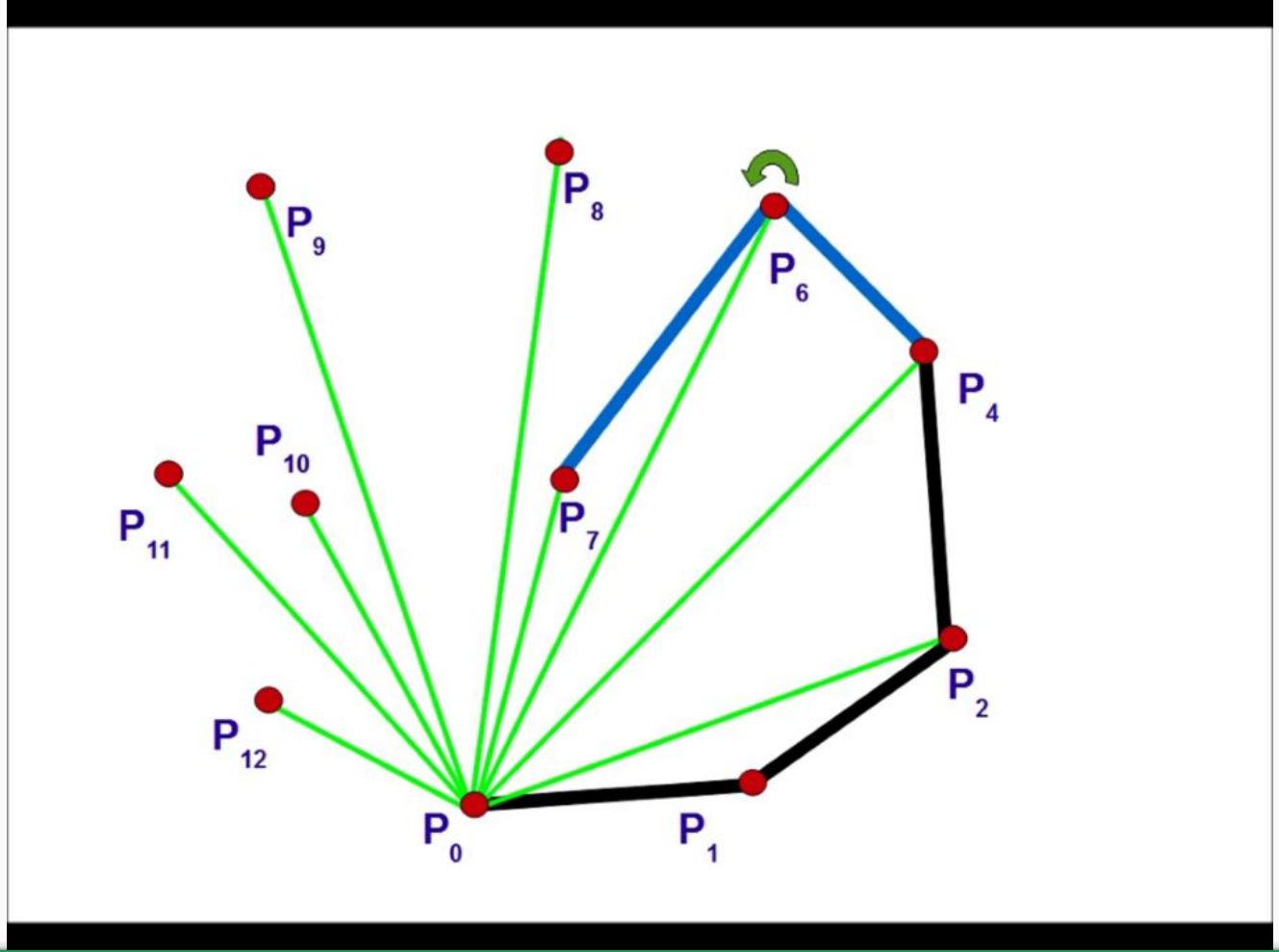
Totally not stolen from Robin



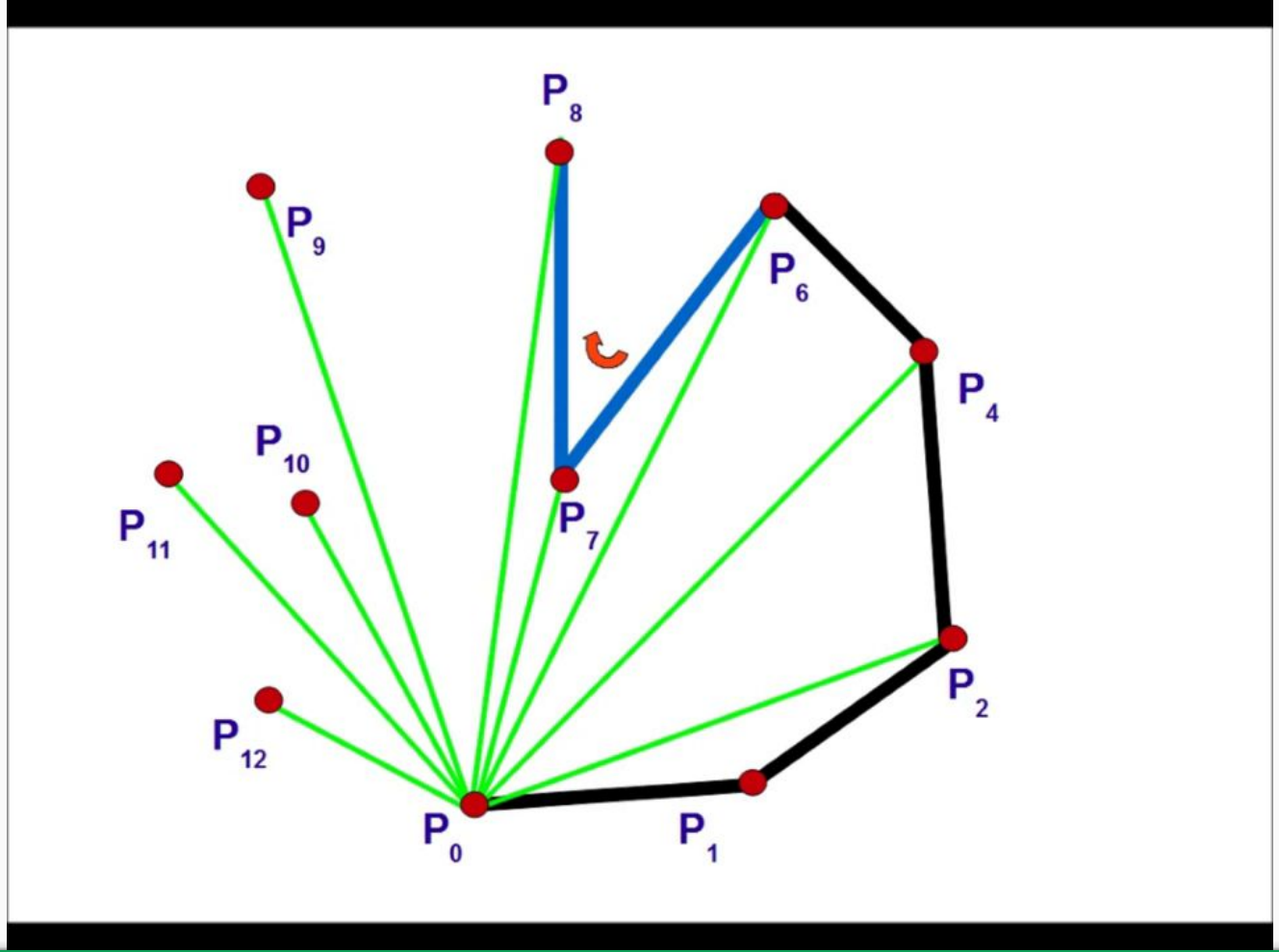
Totally not stolen from Robin



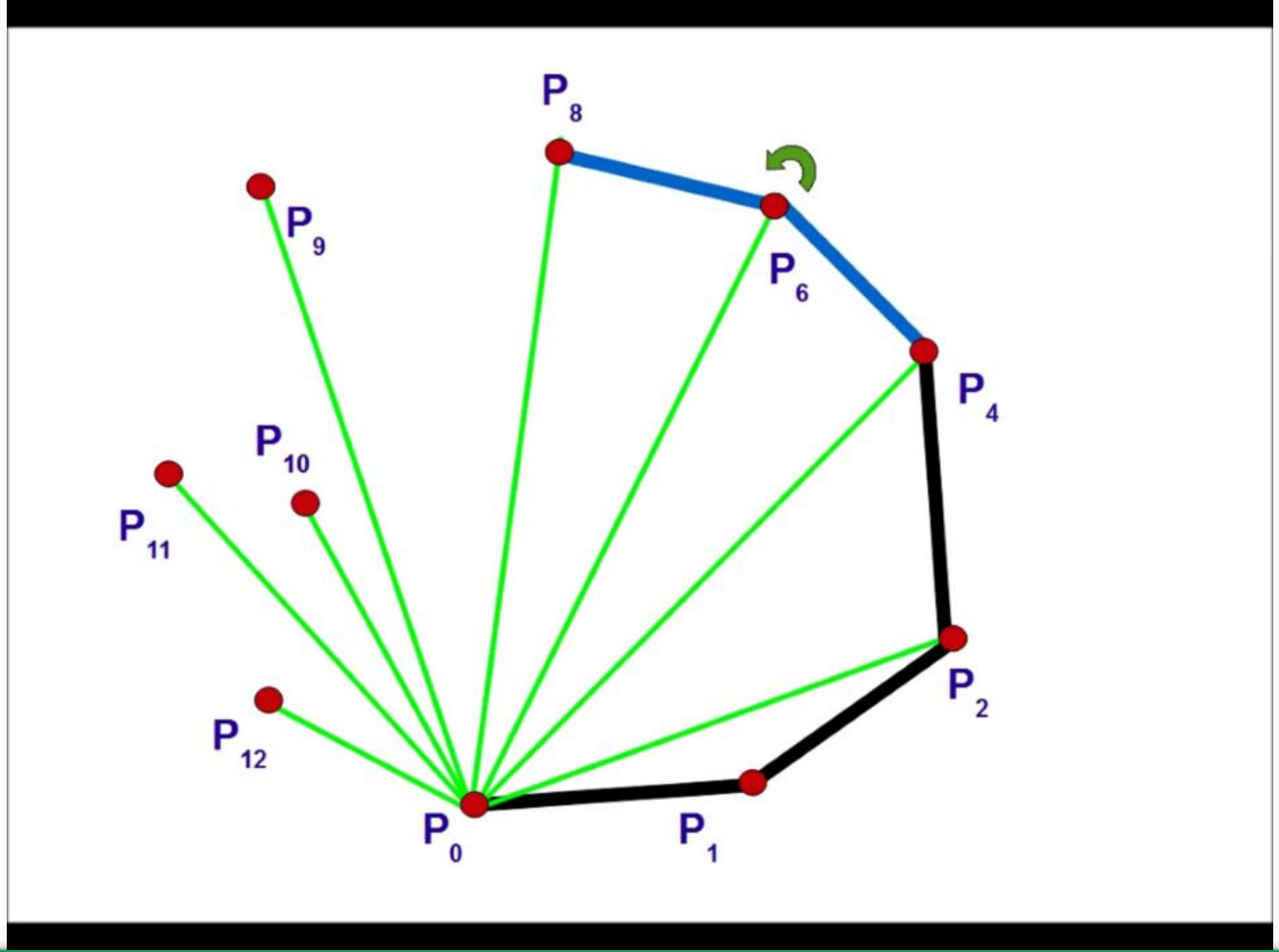
Totally not stolen from Robin



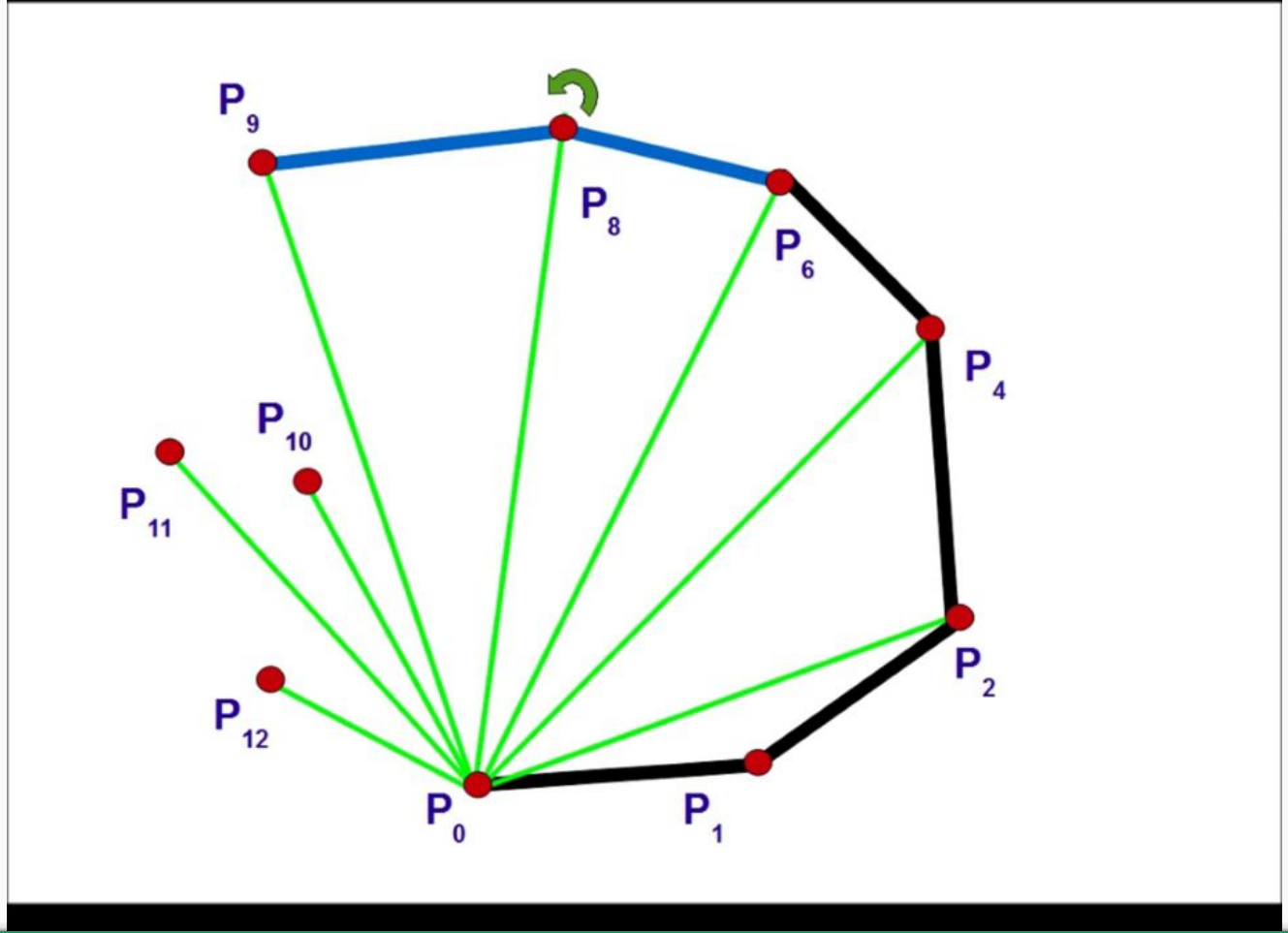
Totally not stolen from Robin



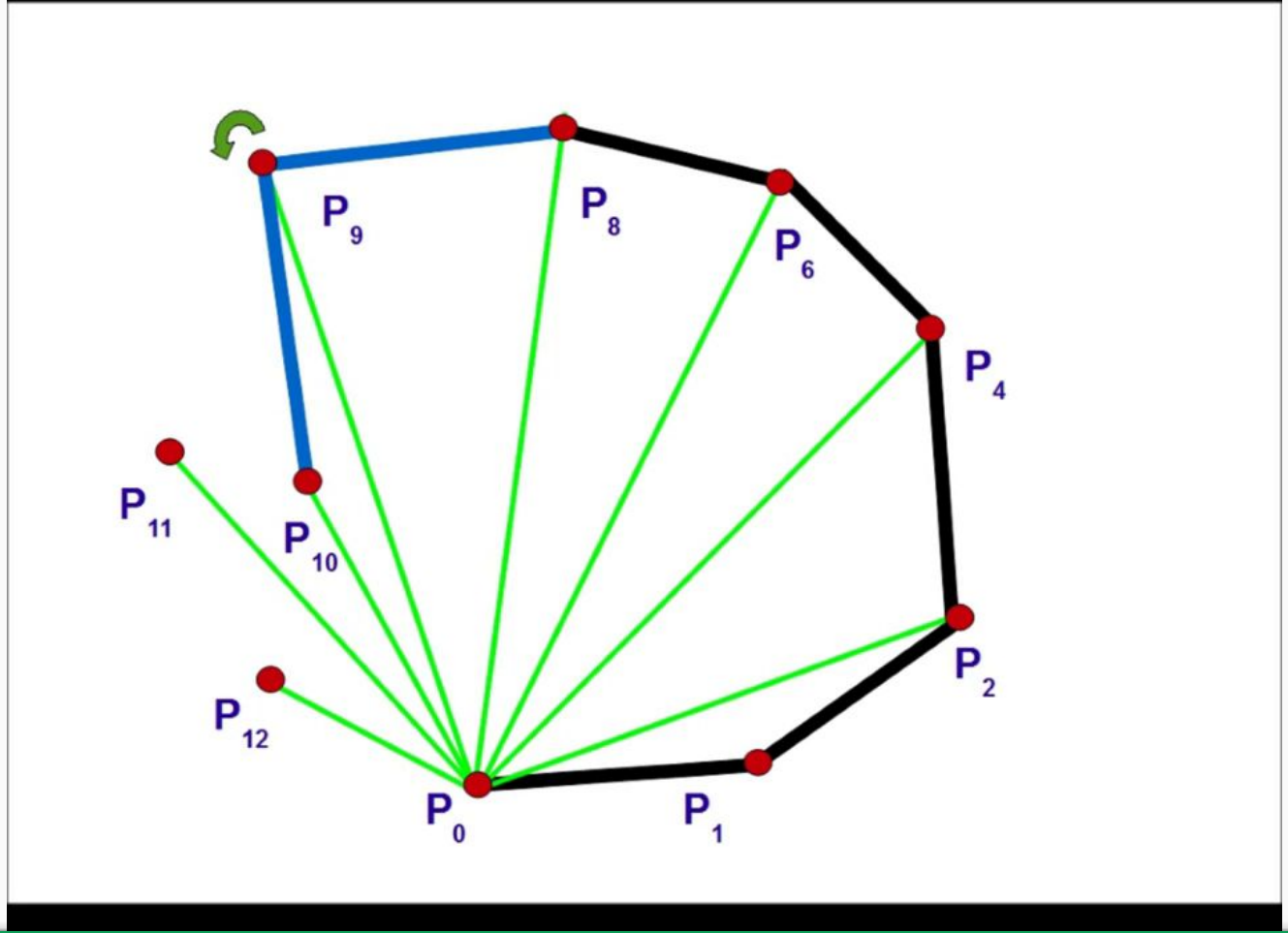
Totally not stolen from Robin



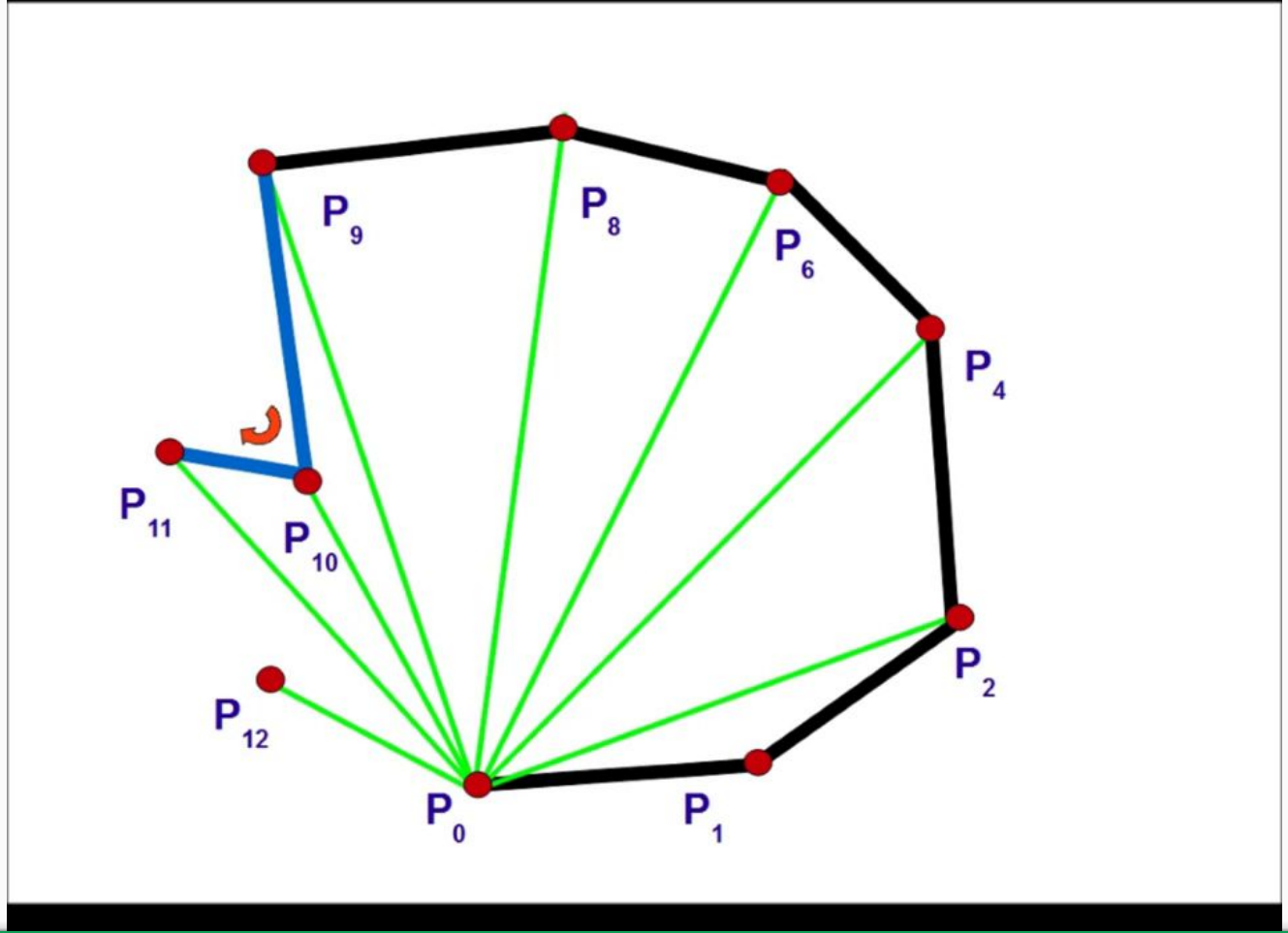
Totally not stolen from Robin



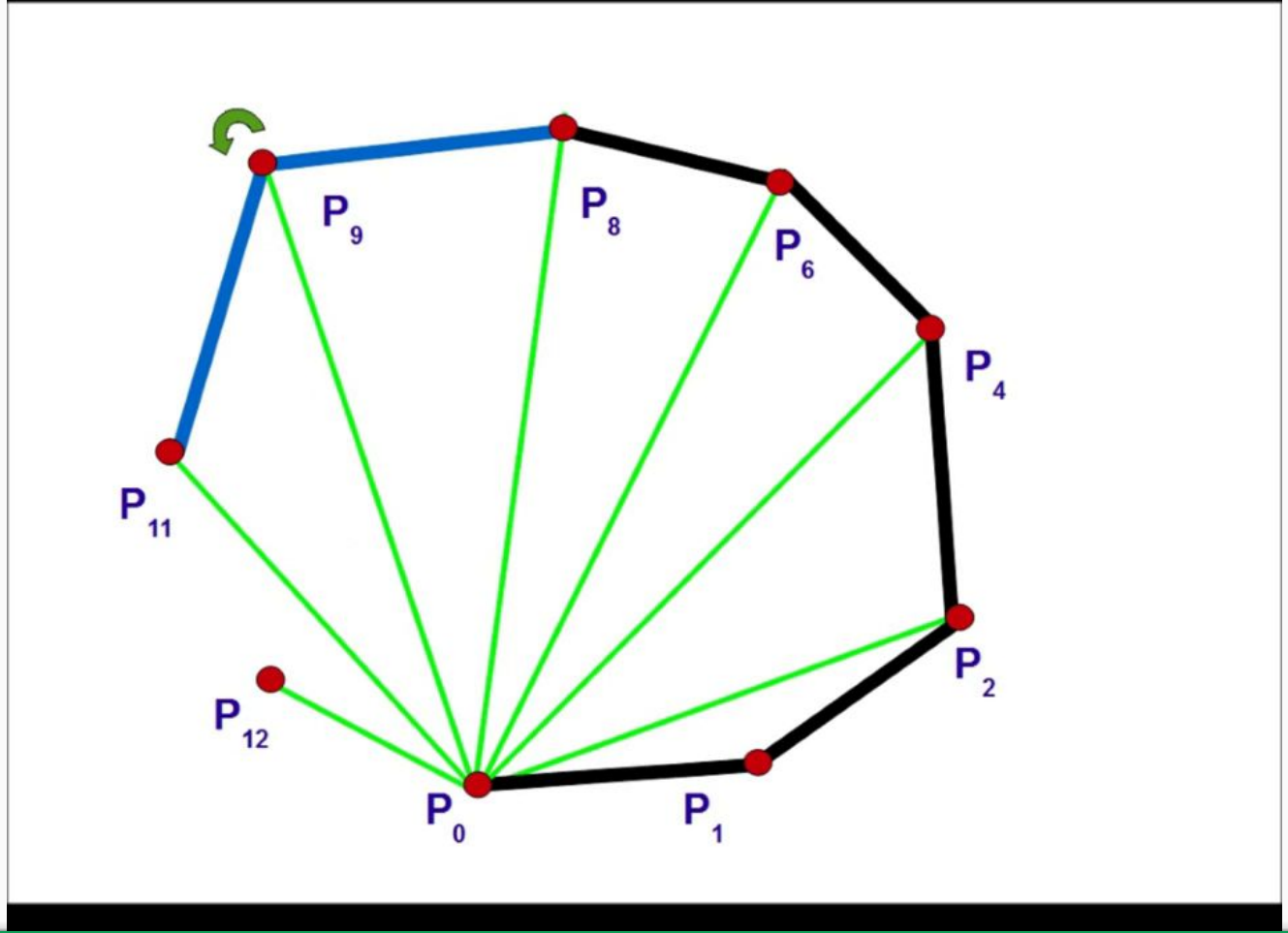
Totally not stolen from Robin



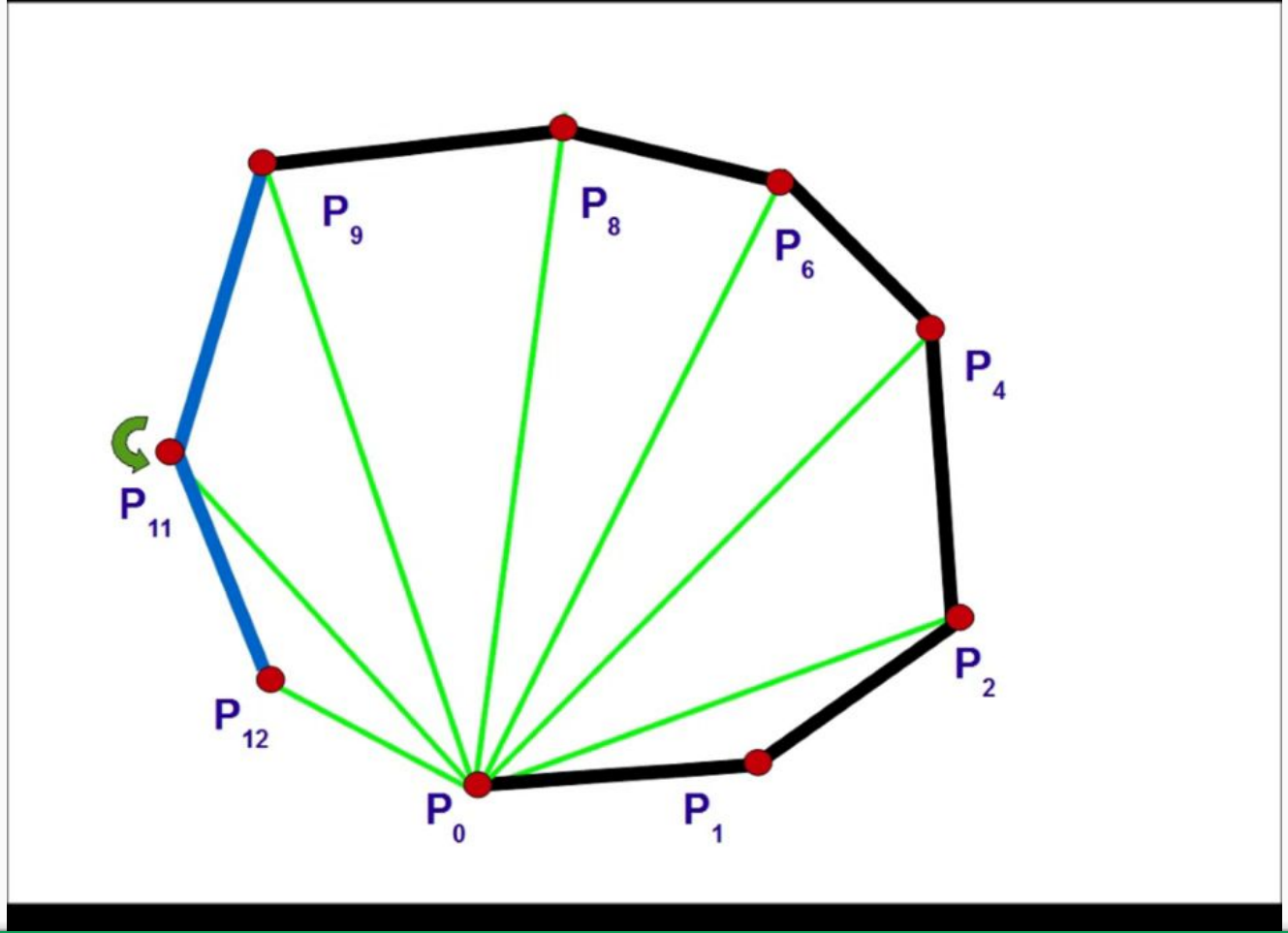
Totally not stolen from Robin



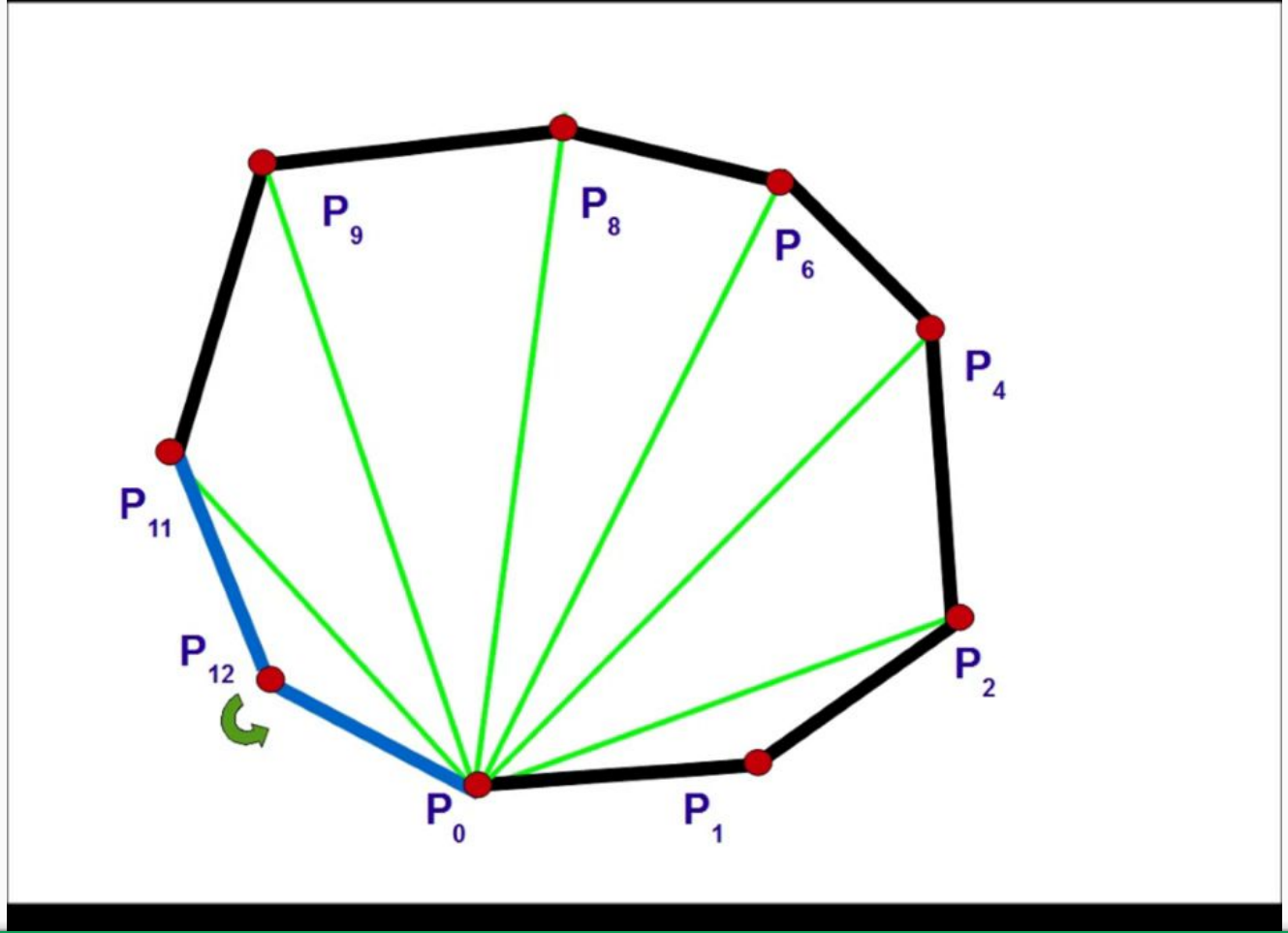
Totally not stolen from Robin



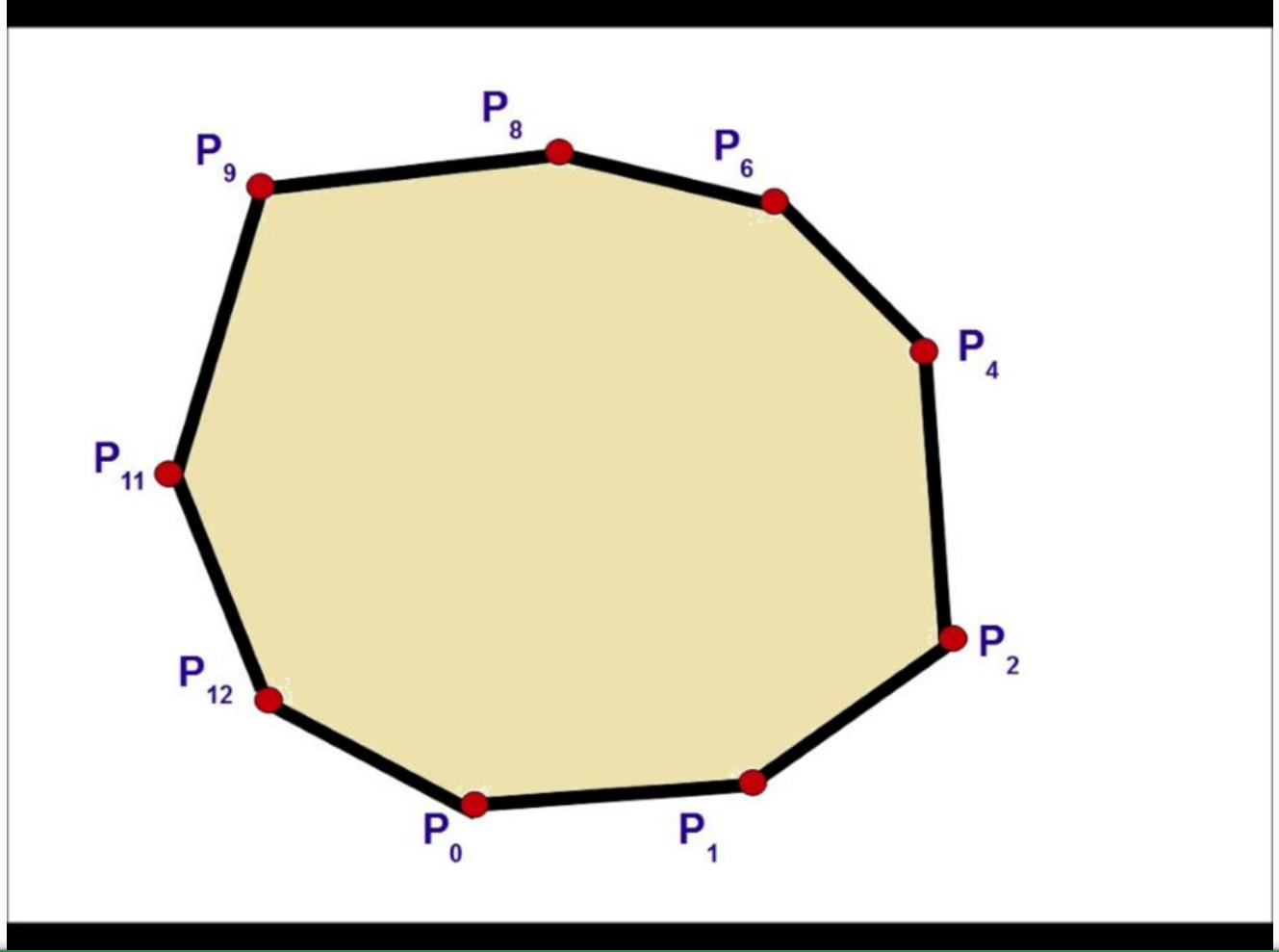
Totally not stolen from Robin



Totally not stolen from Robin



Totally not stolen from Robin

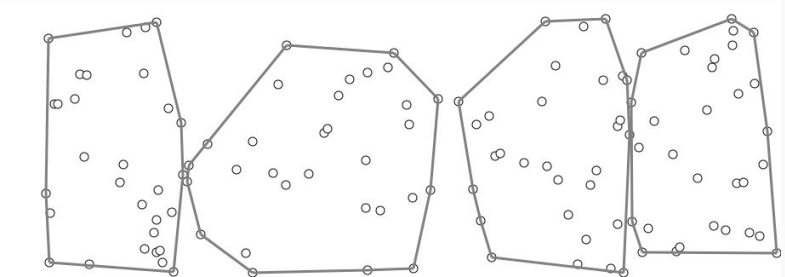


Totally not stolen from Robin

Chan’s Algorithm

An algorithm that combines the Graham Scan with Jarvis’ March to find the convex hull of many smaller convex hulls.

- Partition the points into *K* subsets.
- Find the convex hull of each of those subsets with the Graham Scan.
- Find the convex hull of those convex hulls with Jarvis’ March.



Example Problem 1

Codeforces Round 166 Div 2B:

Given polygons *A* and *B* where *A* is convex, determine whether or not *B* lies inside *A*.

Solution:

Take the convex hull of *A* and *B*, and then check for any points of *B* on the convex hull.

Example Problem 2

USACO Platinum, December 2018 - Balancing Beam:

In order to save money for a new stall in her barn, Bessie the cow has started performing in the local circus, demonstrating her remarkable sense of balance as she carefully walks back and forth on an elevated balance beam!

The amount of money Bessie earns in her performance is related to where she manages to ultimately jump off the beam. The beam has positions labeled $0, 1, \ldots, N+1$ from left to right. If Bessie ever reaches 0 or $N+1$ she falls off one of the ends of the beam and sadly gets no payment.

If Bessie is at a given position k, she can do either of the following:

1. Flip a coin. If she sees tails, she goes to position $k-1$, and if she sees heads, she goes to position $k+1$ (i.e. $\frac{1}{2}$ probability of either occurrence).

2. Jump off the beam and receive payment of $f(k)$ $(0 \le f(k) \le 10^9)$.

Bessie realizes that she may not be able to guarantee any particular payment outcome, since her movement is governed by random coin flips. However, based on the location where she starts, she wants to determine what her expected payment will be if she makes an optimal sequence of decisions ("optimal" meaning that the decisions lead to the highest possible expected payment). For example, if her strategy earns her payment of 10 with probability $1/2$, 8 with probability $1/4$, or 0 with probability $1/4$, then her expected payment will be the weighted average $10(1/2) + 8(1/4) + 0(1/4) = 7$.

Solution:

Plot each pair *(k, f(k))* on the coordinate plane, take the upper convex hull, and the answer for each *k* is simply the *y* value of the convex hull at *k*.

Example Problem 3

Some Australian guy's problem:

You have *n* objects. The *i*-th object has values t_i and p_i. The profit of a subset of size *m* is calculated as follows: you sort the elements in the subset by their *t* values and add up $(t_{s_{j+1}} - t_{s_{j-1}}) \cdot p_{s_j}$ over all *j* from 1 to *m*, where s_j is the *j*-th object of the subset (1-indexed). Assume $t_{s_0} = 0$ and $t_{s_{m+1}} = T$ for some *T*. Find the maximum profit for any subset if all t_i are distinct.

Solution:

The answer is twice the area of the convex hull after plotting each object.

Dynamic Convex Hull (Insertions Only)

To add a point:

- First check whether or not the point is in the convex hull.
  - This is done by finding the centroid (or any other point inside) of the convex hull. If the additional point and the centroid lie on the same side of every edge of the convex hull, then it is inside.
- If not, find the upper and lower tangents from the point to the convex hull.
  - This is done by choosing the point closest to the point being added, and then moving clockwise/counterclockwise until you reach a tangent.

3D Convex Hull

A convex hull in 3D. The most common algorithm for this is Chan’s Algorithm (still O(n log h)). (You don’t really ever need to know this)

Example Problem 1

Some Past GCJ Problem:

OMG FIND THE 3D CONVEX HULL OF THESE POINTS!!1!!!11!!1!

Solution:

Just do it.

Example Problem 2

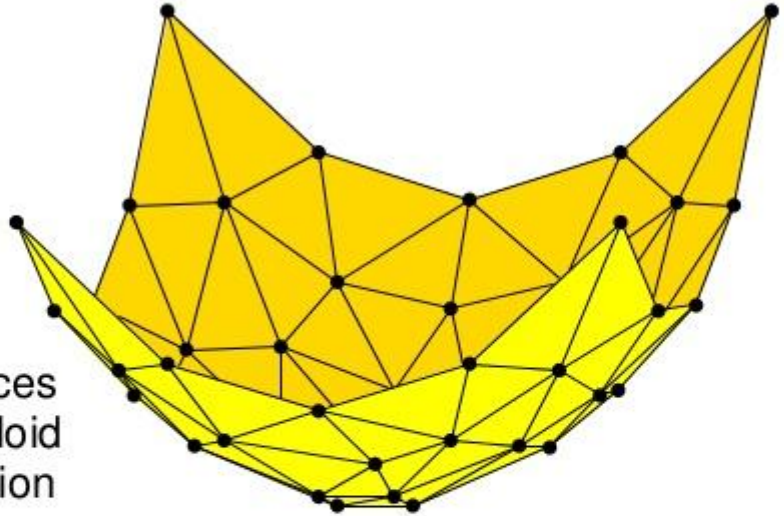
Delaunay Triangulation (used in 3D modelling and Euclidean MSTs):

Given some points in the plane, triangulate them such that the circumcircle of every triangle made is empty.

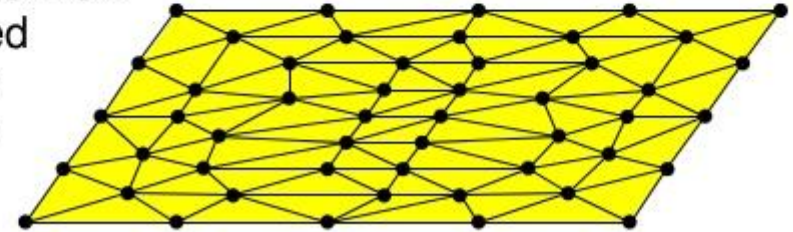
Solution:

Apply a parabolic lifting map (i.e. project onto a 3D paraboloid) on each point, and find the 3D convex hull of those projected points i.e. map each point (x, y) to the point in space $(x, y, x^2 + y^2)$.

Parabolic Lifting Map



"Lift" the vertices onto a paraboloid in one dimension higher. The convex hull of the lifted vertices gives the Delaunay triangulation.



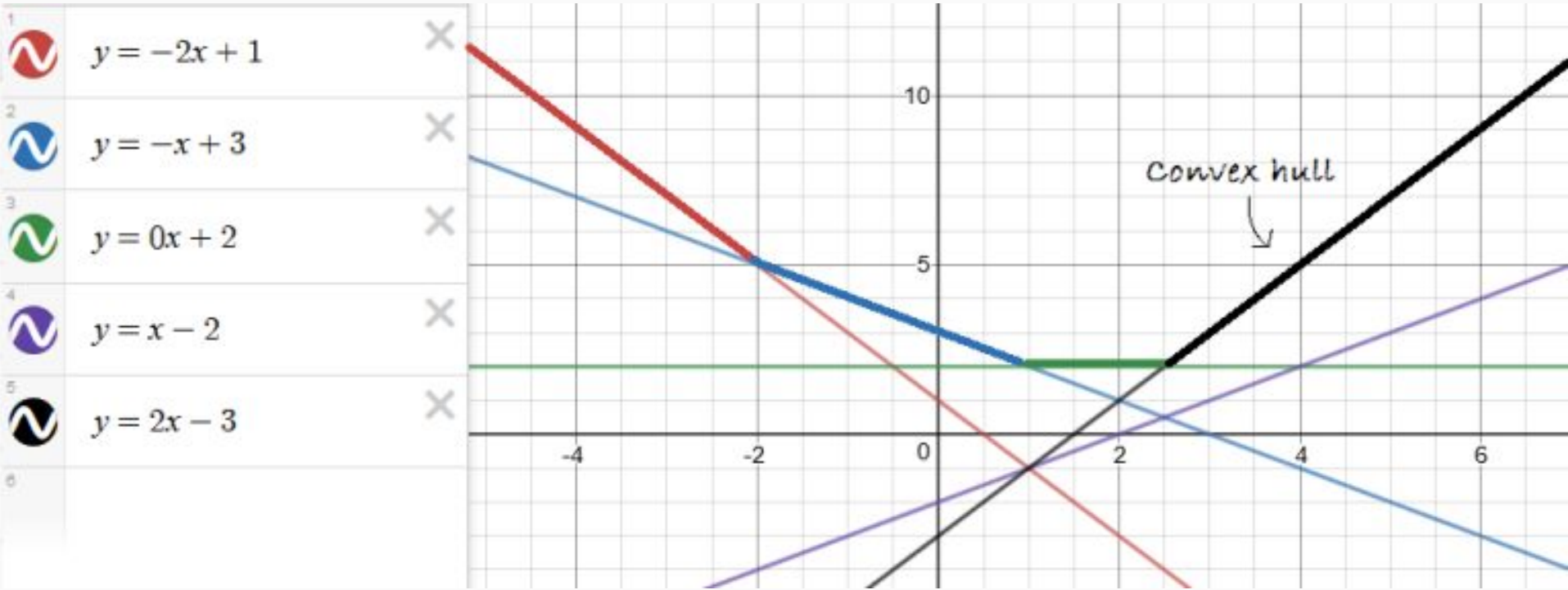
Wow! Who knew 3D convex hulls could actually be useful?

Convex Hull Trick

Given a set of functions $f_i(x) = A_i x + B_i$ and then given queries where we need to answer 2 types of queries, add a function or get max/min at a point, we can do the following:

- Store the upper or lower convex hull of the lines of these functions.
- Discard every other line.
- Store the intersections of the remaining lines.

This allows us to answer these queries efficiently.



In this case, we are querying maximum, so we can discard the purple line.

Example Problem 1

Codeforces Round 526 Div 1E:

Given a set of non-nested rectangles, each with cost A_i, find the maximum union of rectangles - sum of those rectangles' A_i.

Solution:

Sort each rectangle by their x value and apply CHT.

Example Problem 2

Ralph and Unreadability:

Given arrays *A* and *B* where $A[i] \geq A[i+1]$ for every *i*, find the minimum *dp[n]* where $dp[i] = min(\{dp[j] + A[i] \cdot B[j] : j < i\})$

Solution:

Left as an exercise to the reader.

Line Sweep

Line Sweep

“Sweep” a vertical line through a set of points/segments and find cool stuff, usually using a BBST to help with relationships.

Stuff you can do with a line sweep:

- Find Delaunay Triangulation and Voronoi Diagrams
- Find intersections between a set of line segments
- Find the closest pairs of points in a given set of points
- And more

Example Problem

SACO 2018 Day 1 Problem 3:

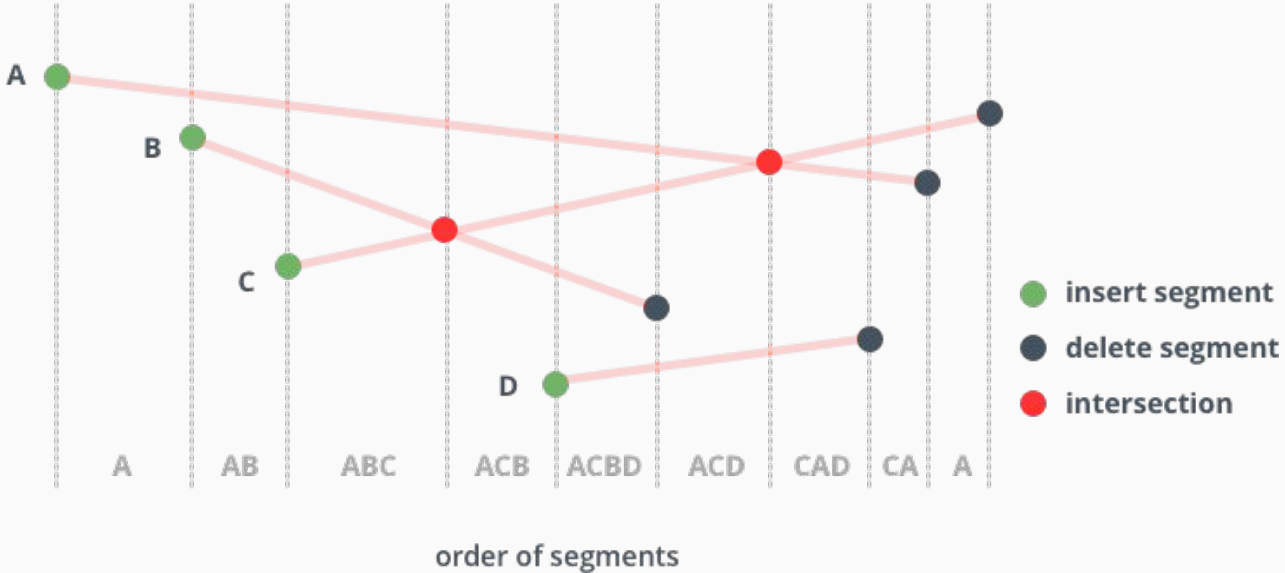
Find the largest square with no obstructions in a grid with given obstructions.

Solution:

Do a line sweep and magical stuff happens.

Bentley-Ottmann Algorithm

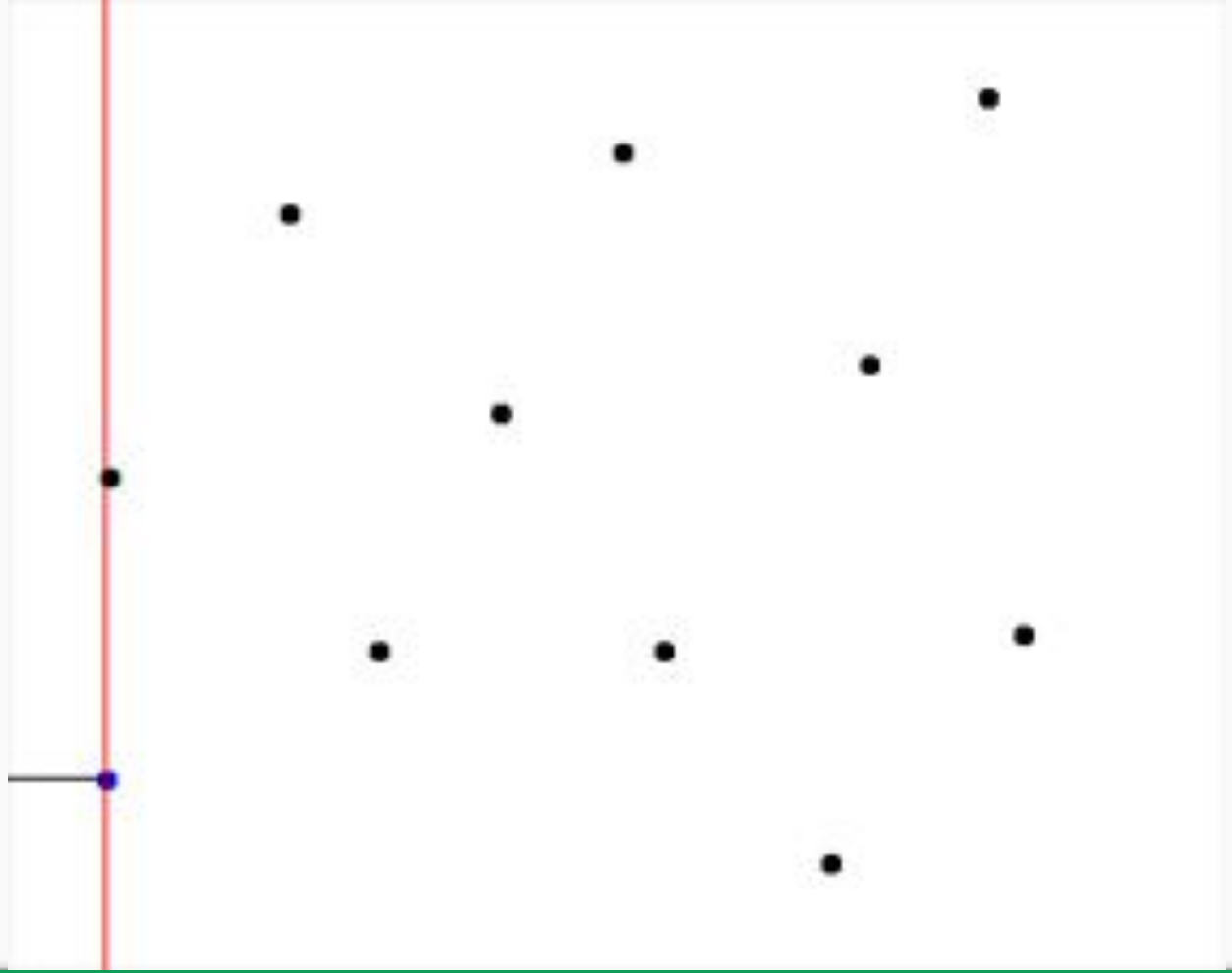
An algorithm to find all intersections in a set of line segments in O((n + i) log n) time:



Fortune’s Algorithm

An algorithm that finds the Voronoi diagram of a set of points in O(n log n) time and O(n) space:

Sweep a line through a set of points and then for each point, make a parabola corresponding to each point as the focus and the sweep line as the directrix.



Fun Fact: baking some dough balls in a dish gives you the same effect

Miscellaneous

Rotation

Rotation by 90 degrees (clockwise): $(x, y) \rightarrow (y, -x)$. Scale stays the same.

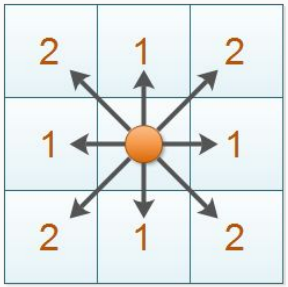
Rotation by 180 degrees: $(x, y) \rightarrow (-x, -y)$. Scale stays the same.

Rotation by 45 degrees (clockwise): $(x, y) \rightarrow (y + x, y - x)$. Scale increases by $\sqrt{2}$.

Manhattan and Chebyshev Distance

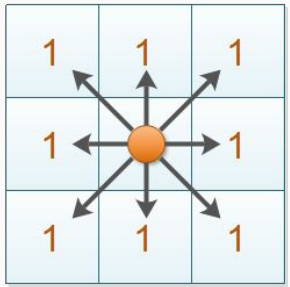
Given 2 points *P* = (*x1*, *y1*) and *Q* = (*x2*, *y2*), the Manhattan distance between *P* and *Q* is given as |*x1* - *x2*| + |*y1* - *y2*|, while the Chebyshev distance is given as max(|*x1* - *x2*|, |*y1* - *y2*|).

Manhattan Distance



$$|x_1 - x_2| + |y_1 - y_2|$$

Chebyshev Distance



$$\max(|x_1 - x_2|, |y_1 - y_2|)$$

Example Problem

SACO 2017 Day 1 Problem 2:

Given a list of stars in the order that they appear each day, output the size of the constellation that each appear in, if a constellation has maximum distance between connected stars less than a specified Manhattan distance.

Solution:

Rotate the grid 45 degrees and use a 2D segtree to query the rectangle bound by the Manhattan distance given. Manhattan distance now becomes Chebyshev distance, and a simple DSU can be used to join constellations.

Reflection

To reflect a point (x, y) around a line with equation $y = mx + c$,

Let $d = \frac{x+(y-c)\cdot m}{1+m^2}$

Then we have (x, y) $\rightarrow$ ($2d - x$, $2dm - y + 2c$)

(Do not question how I got this result)

Example Problem

Law of Reflection:

Given a point object *P*, an observer at a point *O*, and a mirror *AB*, determine whether *P* is visible from *O* from looking in the mirror.

Solution:

Reflect *P* across *AB* to *P'*, and determine whether *OP'* intersects *AB*.

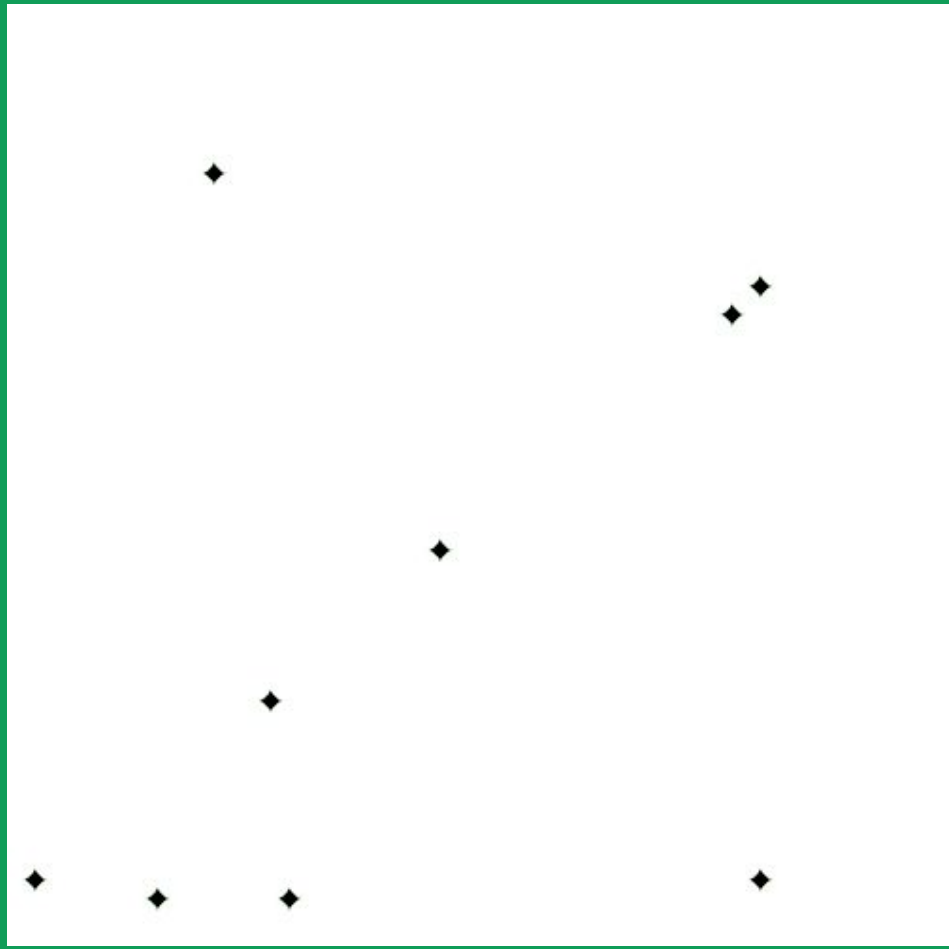
Rotating Calipers

Place the polygon in the jaws of an imaginary caliper (a distance-measuring tool), and then rotate it. This is an elegant technique to find:

- All antipodal pairs of a polygon (i.e. vertices such that the line between them is collinear with one of the edges)
- Diameter/width of a polygon
- Max/min distance between 2 polygons
- A lot of triangulations
- Union/convex hull/common tangent/intersection of 2 polygons

Creating the Rotating Caliper Graph

Questions?



Minkowski Distance Animation