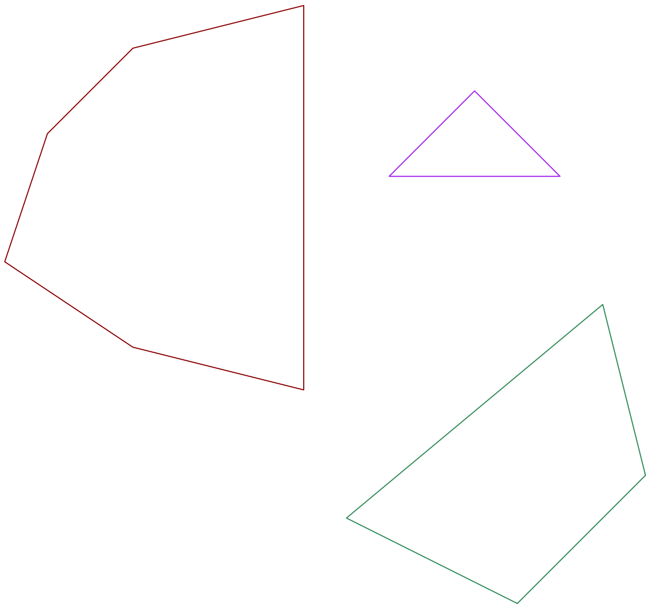
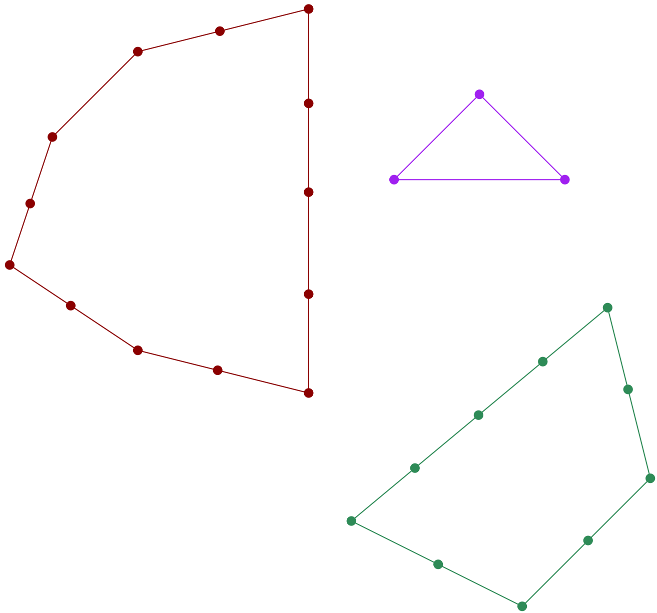
Basic Algorithm

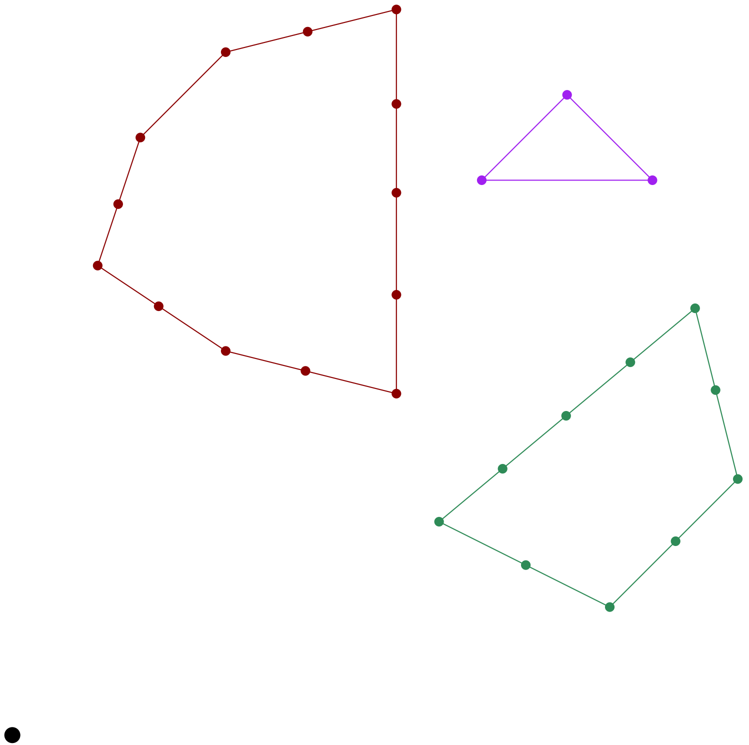
Start with polygons



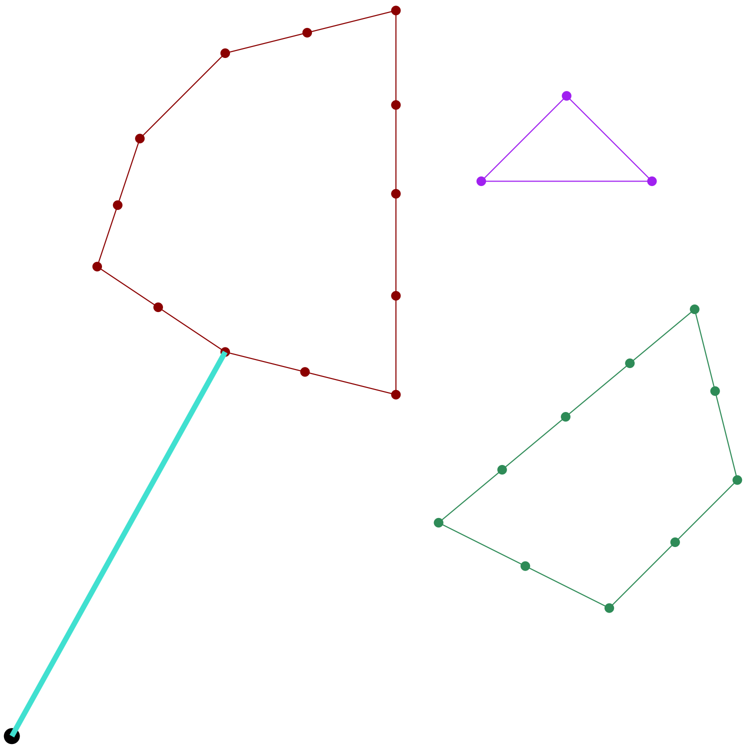
Add extra vertices to long edges



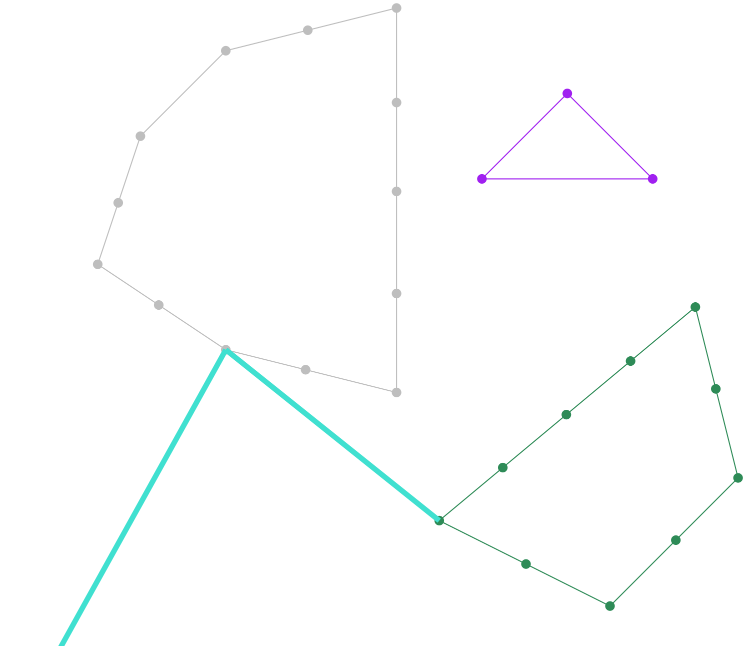
Choose a starting point (in this case the cutter always starts from the 'home' position)



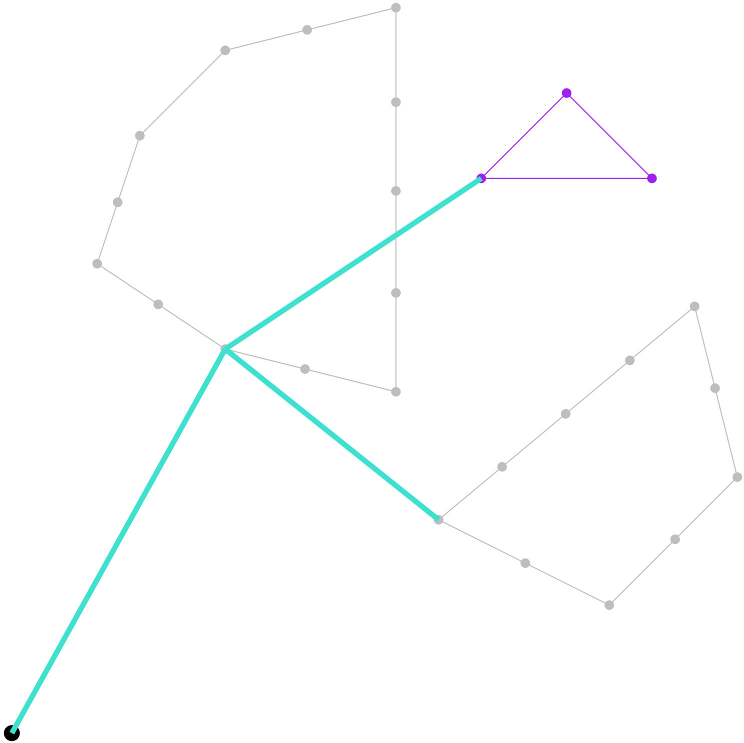
Find the nearest point and draw a line to it to start a tree



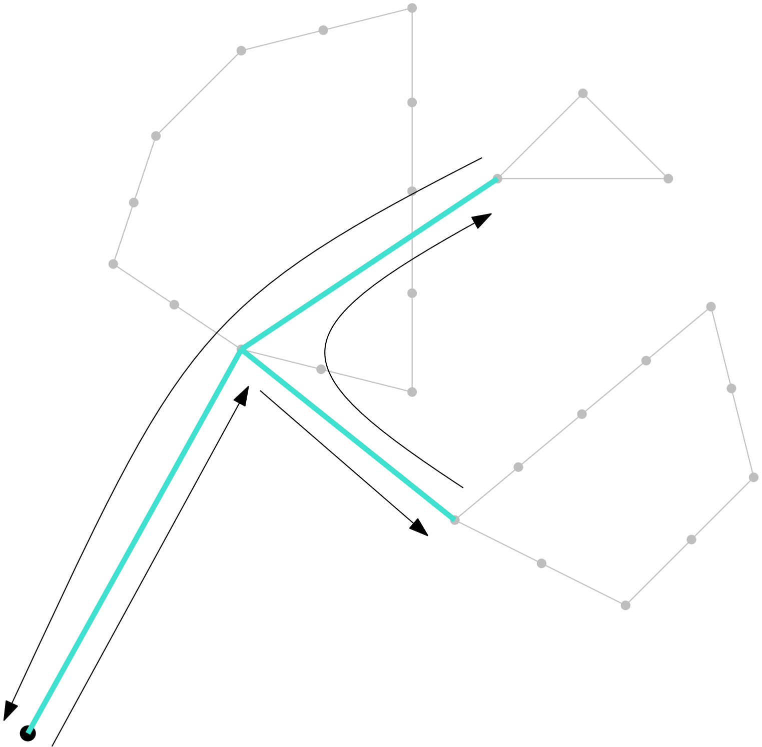
Find the nearest point from any unvisited polygon to any point in the tree



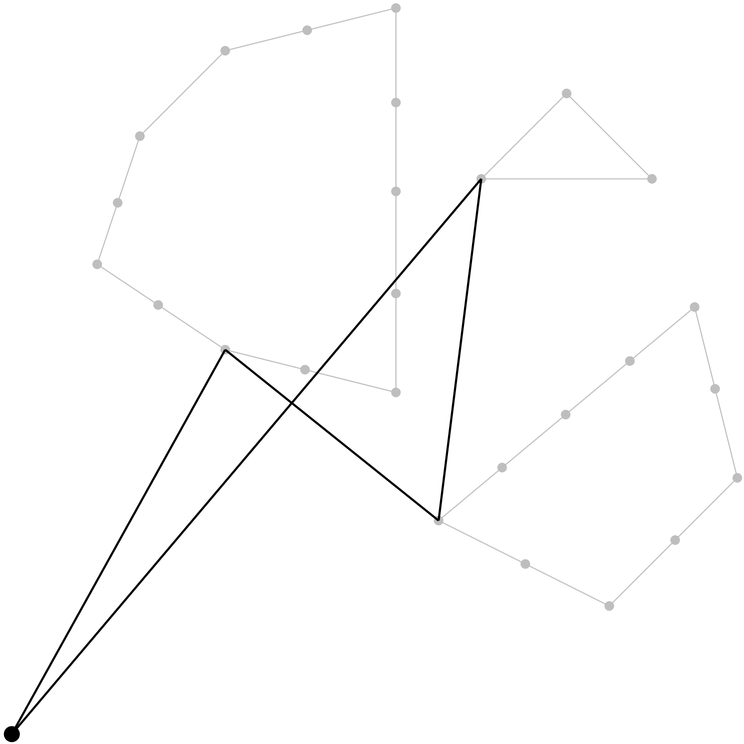
Repeat



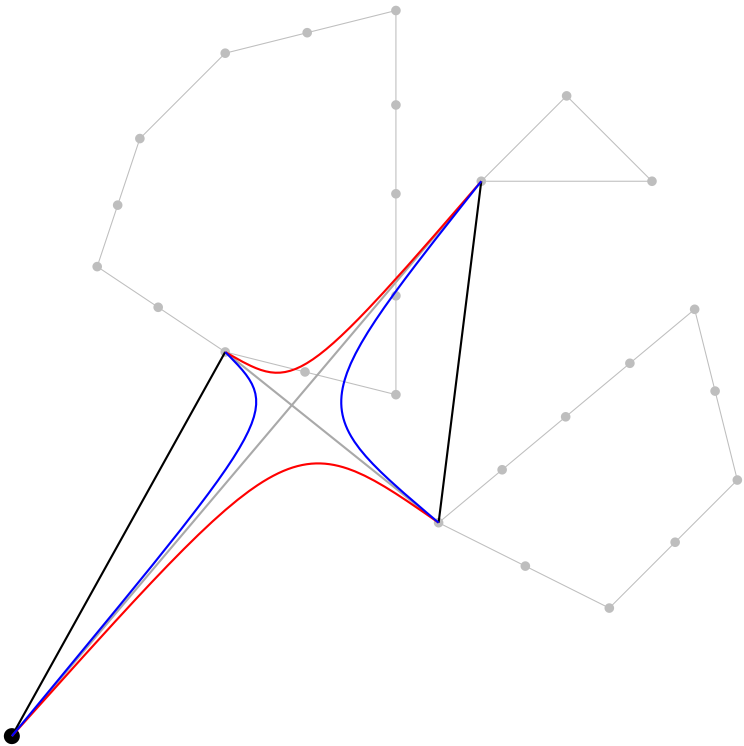
Walk around the "outside" of the tree, skipping already-visited nodes



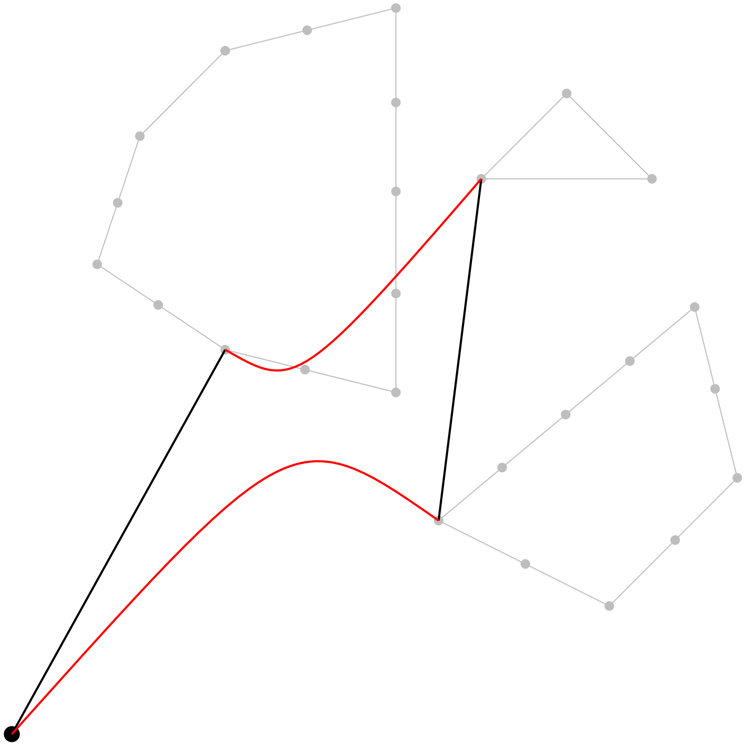
This makes a cycle



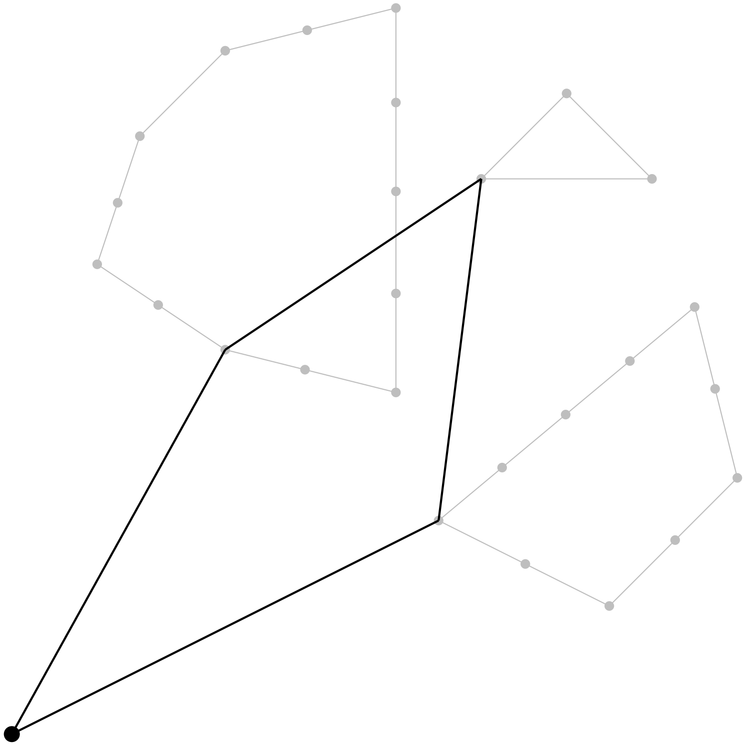
Uncross intersections. There are two choices, red or blue:



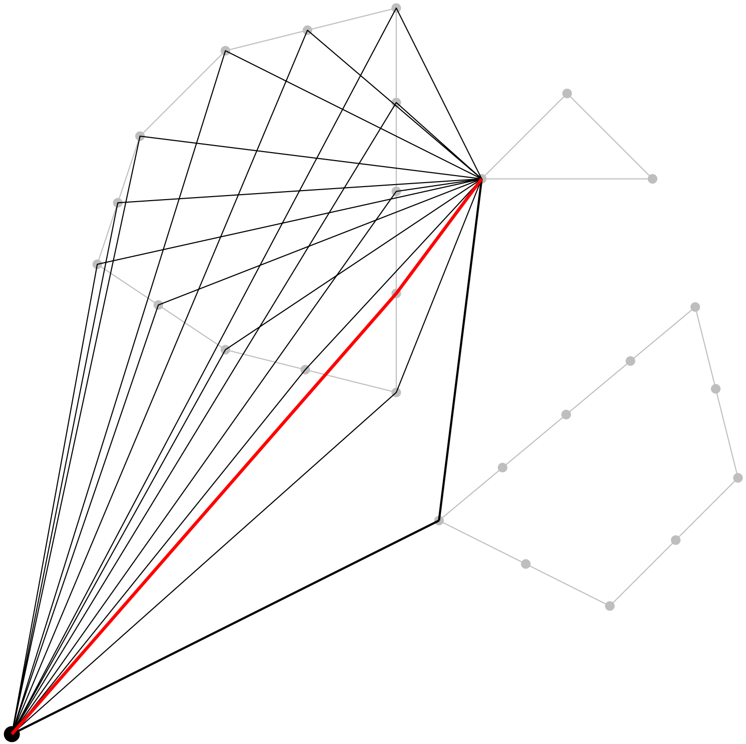
Only red keeps it a cycle



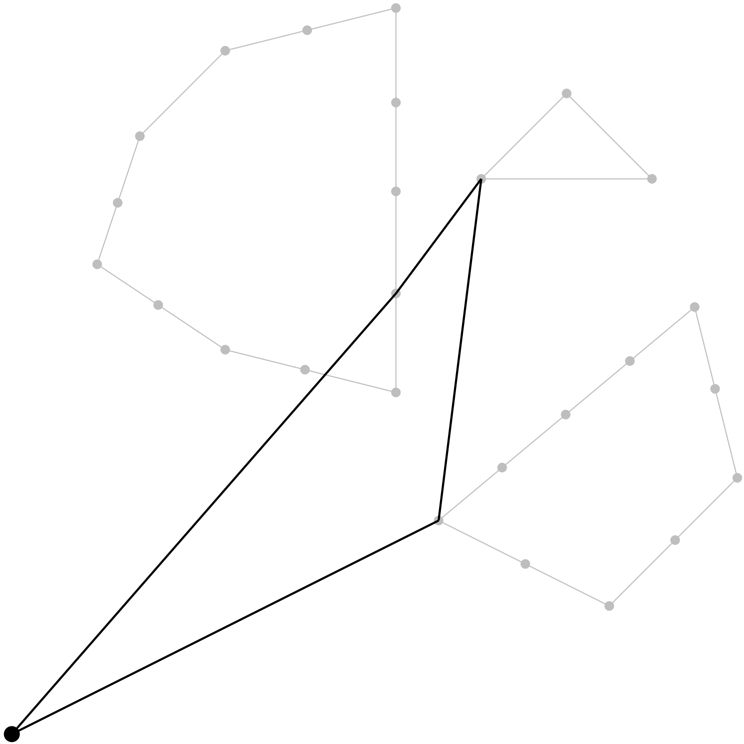
Now improve the choice of polygon vertex



Try every vertex in a polygon to see which makes the path shortest

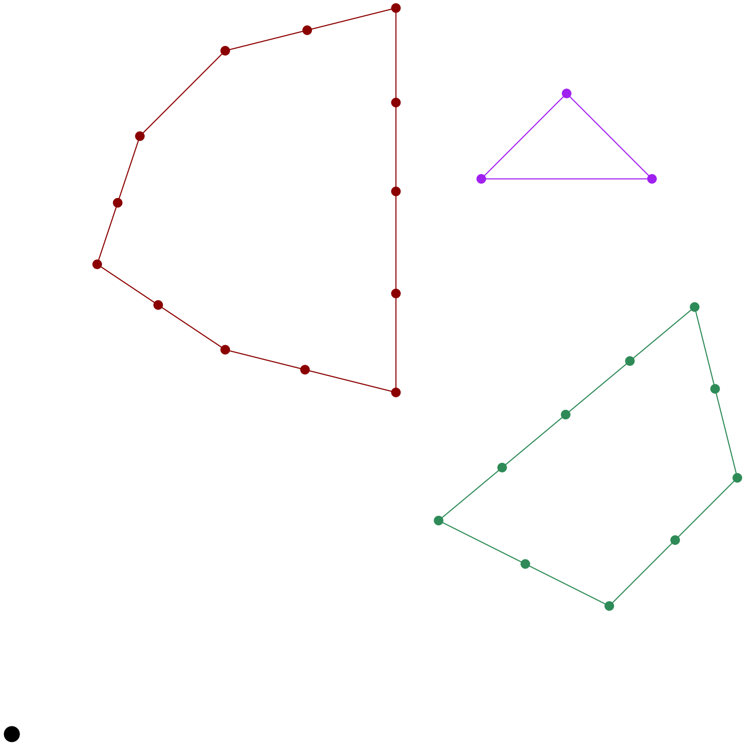


Repeat for every polygon and we're done!

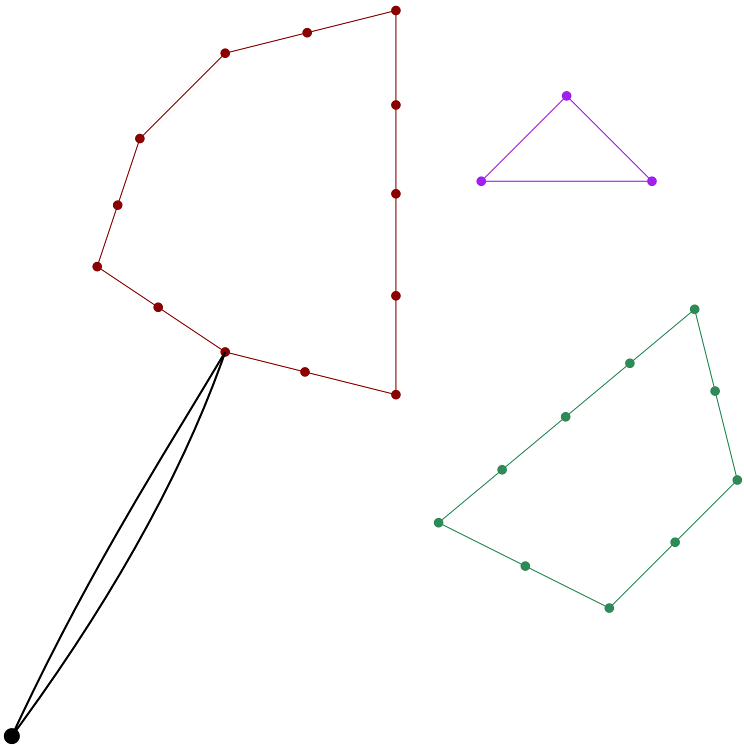


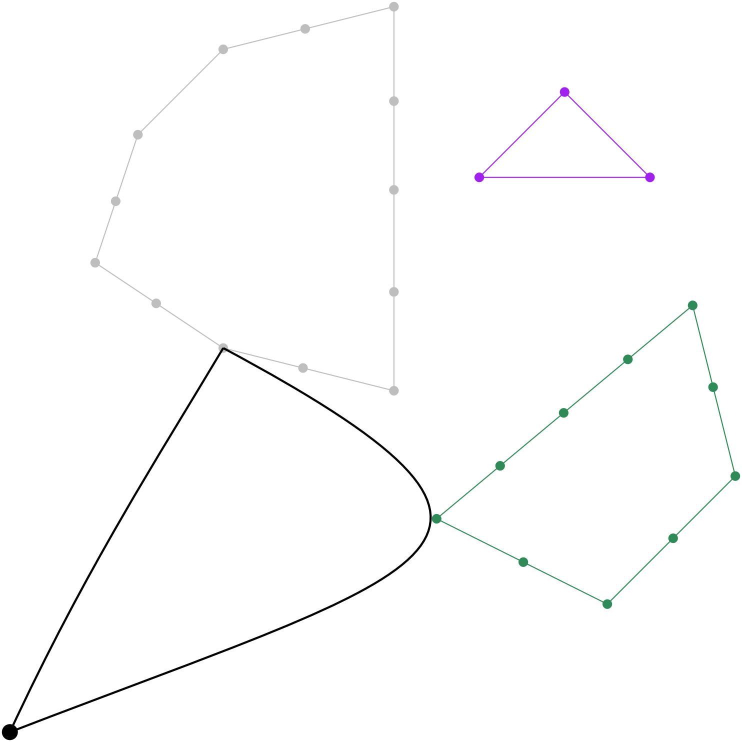
Alternative to walking around a tree

Instead of a tree, gradually expand a cycle

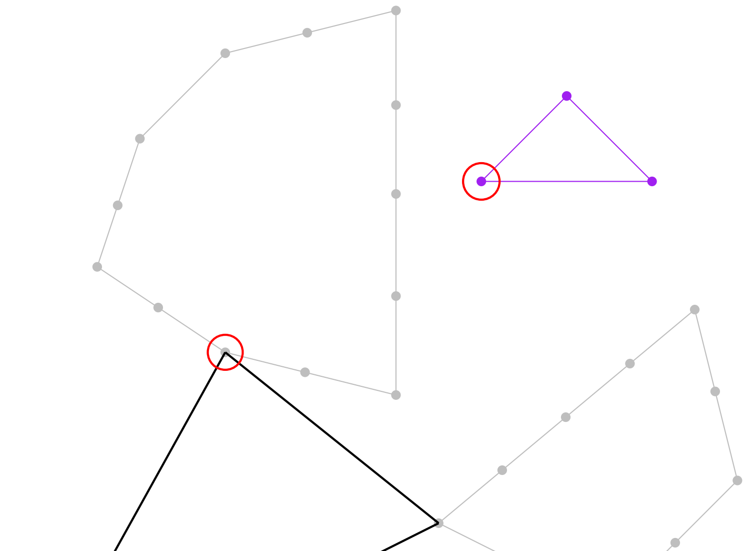


Find the shortest detour to a new polygon

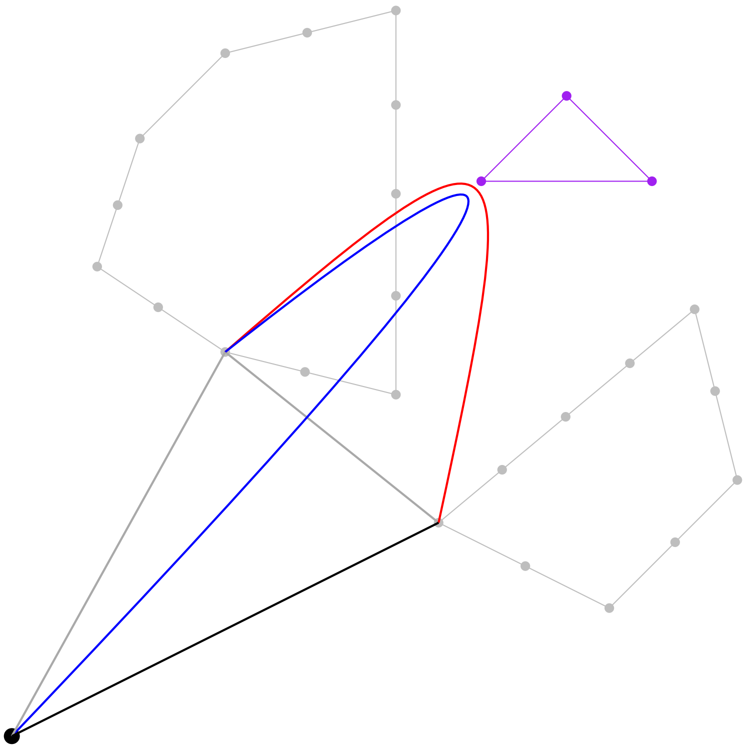




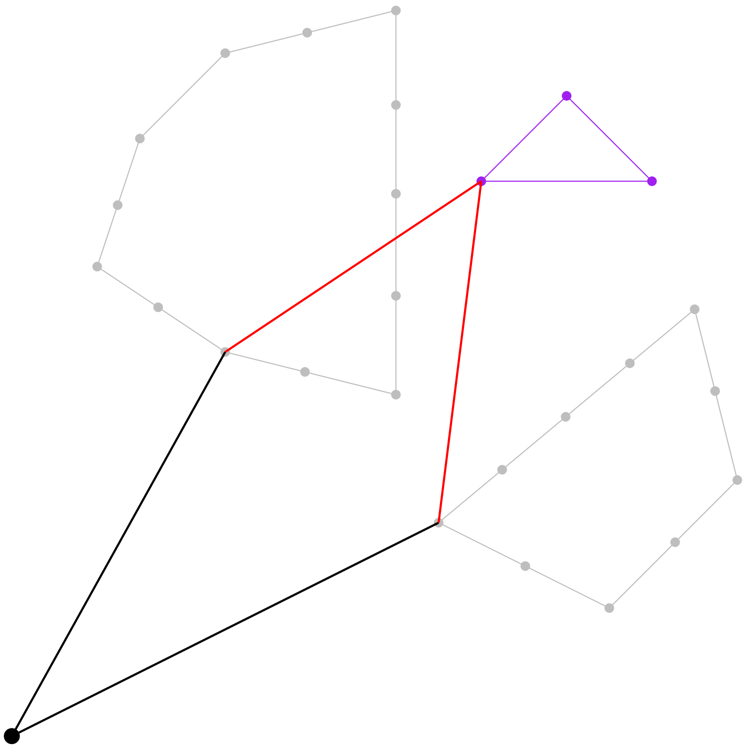
To do so choose the closest vertex on the cycle to any unvisited point



And consider bending the two adjacent edges to the new polygon



Red is the shortest detour so choose that



Details / Questions

Uncrossing an intersection (red) can make an extra intersection appear (green).

Do we have to do multiple loops checking for intersections? Is convergence guaranteed?