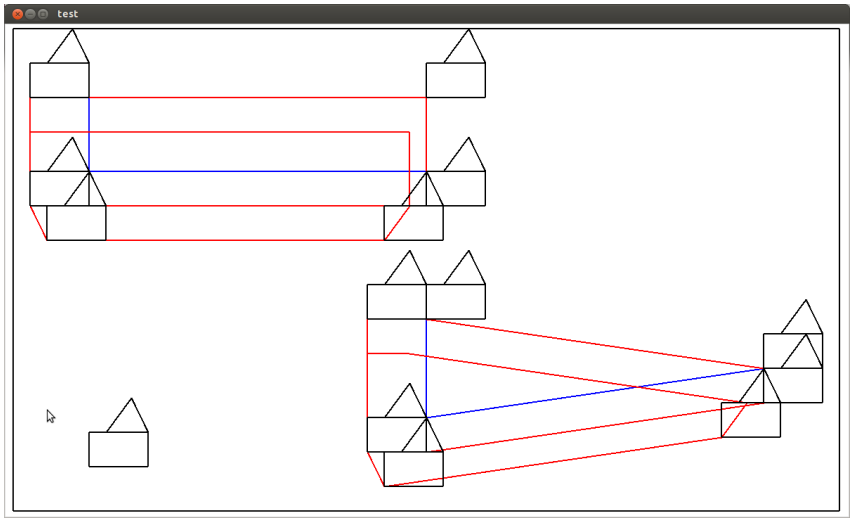
Jorge Perez
6.S078
Problem set #2

Part 0. Implement configuration space obstacles for convex polygons

To implement configuration space I used the star algorithm. First, I wrote a method that took in a vector and side of the vector. The method calculated the vector perpendicular to it on the side pointed to by the parameter. It would then calculate the angle between the calculated vector and the x axis. The vectors would then be sorted by angle and connected to form the desired polygon.

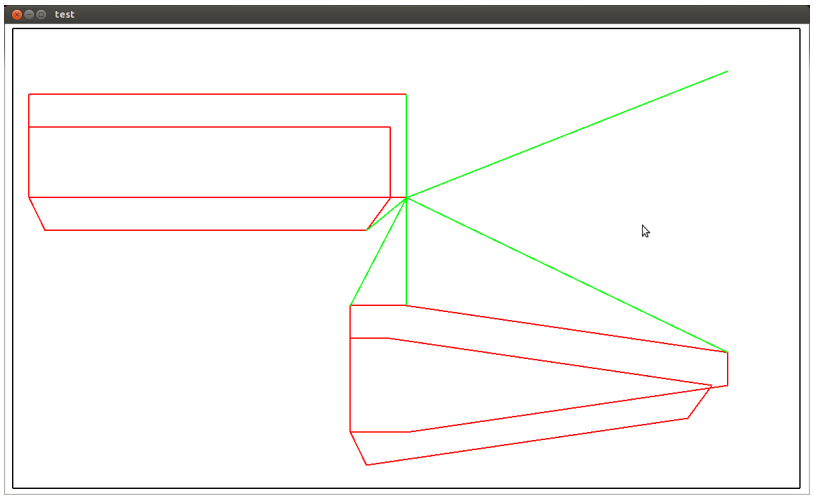
The robot is made out of several polygons so the configuration space needs to take that into account. The configuration space for the robot was calculated by first calculating the configuration space of each of the polygons for the robot. Then, then the separated configurations spaces are superposed and shifted so that the referenced the same origin in the robot.



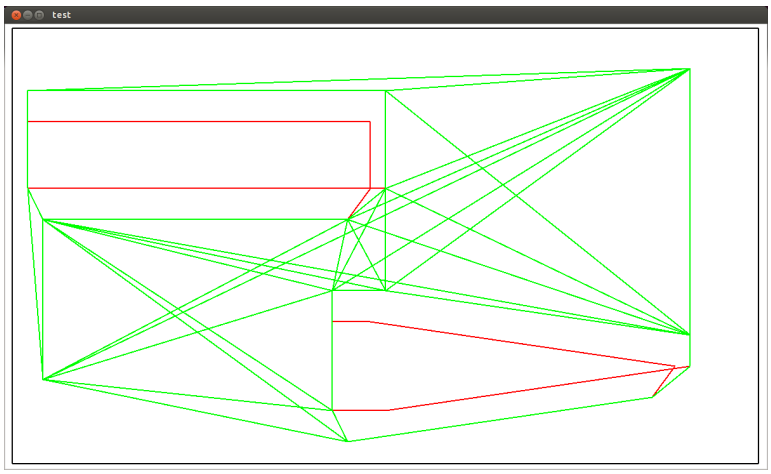
In the screenshot, the blue lines are part of the obstacles, the red lines are the configuration space polygons for the rectangle and triangle and in black is the robot. The robot has been drawn in the outer vertices of the configuration space to illustrate correctness of the configuration space.

Part 1.Implement a visibility graph just for translations

The visibility graph was implemented using the a naive algorithm. The algorithm iterated through all the vertices in the configuration space, the start goal and the end goal connecting them to all other possibilities and checking is the segment created was legal. To check is the segment is legal the algorithm checked for intersections and checked that the midpoint of the segment was not inside a polygon. This avoided connecting vertices in the polygon that were not consecutive.

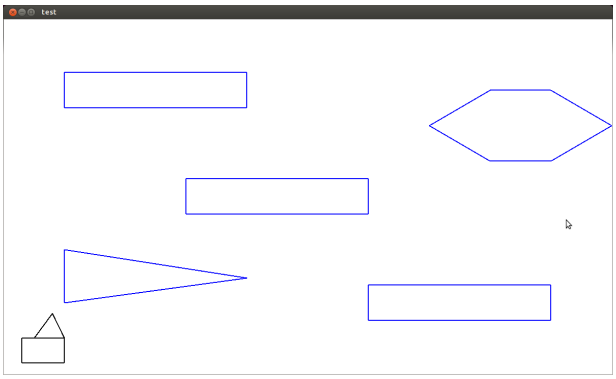


This is an example of one of the points being sampled. The green lines coming out of the points are the children of that node in the graph. The goal node is in the upper right and the start node is in the lower left.

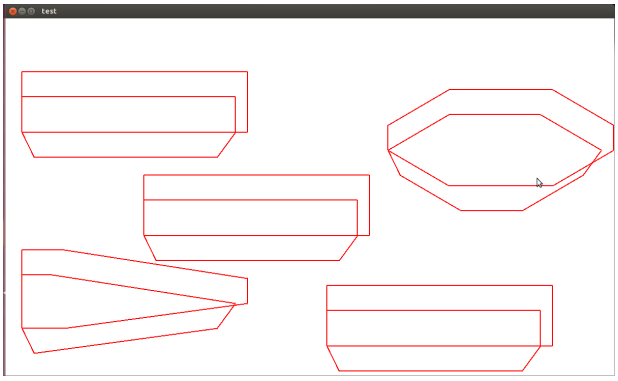


This is the graph once all the points have been sample. Note again that the goal is at the upper right and the start in the lower left.

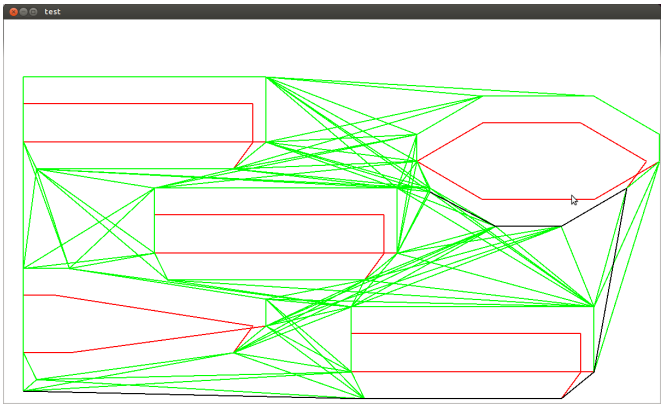
Once the graph was built I ran depth first search, breadth first search, uniform cost search and A star with an admissible heuristic and with an heuristic that was not admissible. The cost of the path was the sum of the length of the segments leading to the goal. The heuristic used for A star was the euclidean distance to the goal. For the non admissible case, the heuristic was multiplied times a constant alpha.



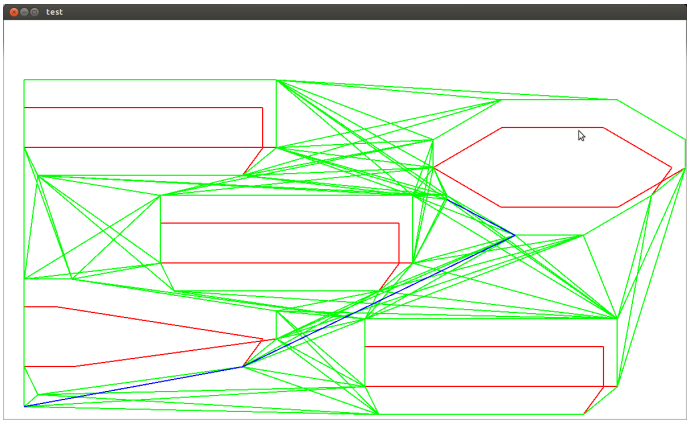
obstacles(blue), robot(black)



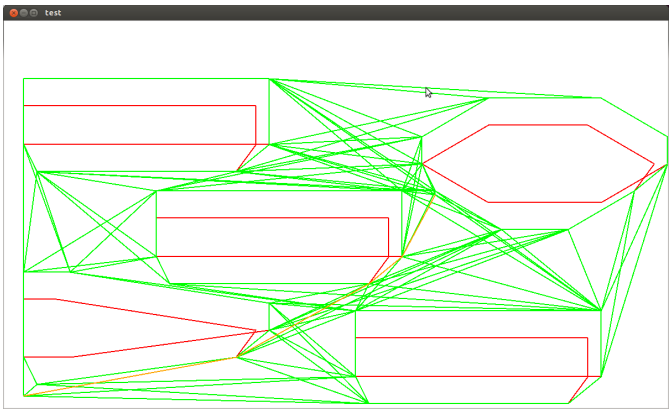
configuration space obstacles



visibility graph (green), DFS path (black)



BFS path(blue)



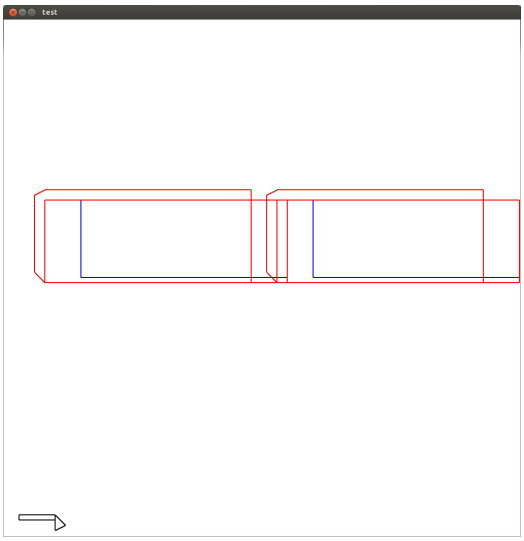
UCS and A star path(orange)

Algorithm	# of segments in path	path cost	# of explored paths
DFS	7	176.49	7
BFS	3	98.83	34
UCS	4	86.47	27
A star (admissible)	4	86.47	27
A star (not admissible)	4	86.47	25

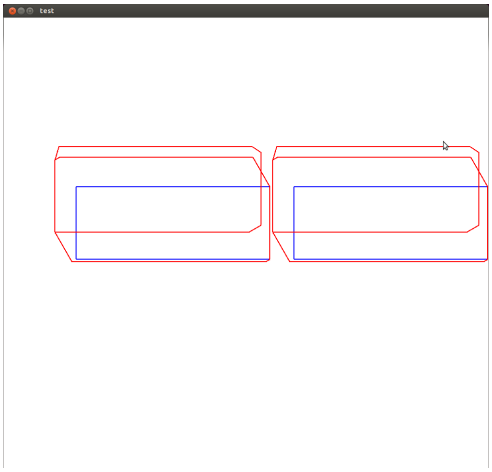
DFS finds a path that is not optimal in any way but it explores less nodes. BFS finds the optimal path in terms of fewest possible segments. UCS and A star both find the optimal path. A star with a non admissible heuristic finds the optimal path in this case but looks at less nodes than the admissible heuristic.

Part 2. Extend it to a closed set of closely spaced orientations of the robot.

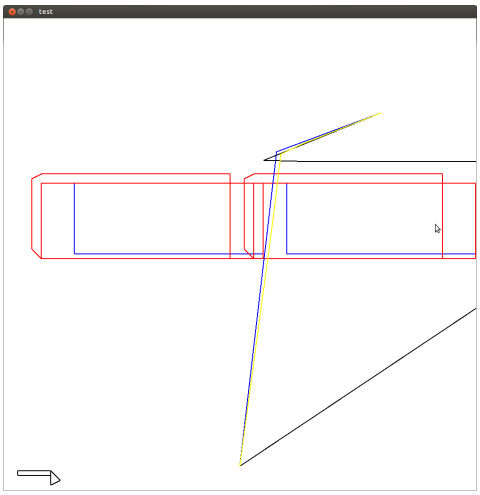
In order to allow rotations, the configuration space for various angles of the robot is calculated. I decided to let the robot go from 0 to -60 degrees in steps of 1 degree. In my setup the can only rotate once it reaches a node. A given node can be transferred to a different layer of rotation if the current position is in that configuration space and in all the layers of rotations below it. For example, if you want to connect from 0 to 60 degrees the node also has to in the 1,2,3,...,59 configuration spaces. Rotations did not affect the cost of the path.



Without rotations the robot did not fit in between obstacles



With a rotation of 60 degrees the robot clearly fits



yellow(Astar and UCS path), blue(BFS), black(DFS)

Algorithm	# of segments in path	path cost	# of explored paths
DFS	17	170.896223793	10
BFS	2	90.8685	829
UCS	5	89.8192	7346
A star (admissible)	5	89.8192	7751
A star (not admissible)	5	89.8192	8538