Path Planning Algorithms Example Notes

Andres Pulido

I. Exact Cell Decomposition

I chose the horizontal polygon decomposition (Figure 1) because I had to implement the line collision in the obstacles. I implemented it so the nodes of my graph are the centroids of each decomposed polygon. I used Dijkstra's for graph search. The final distance was 13.14 units.

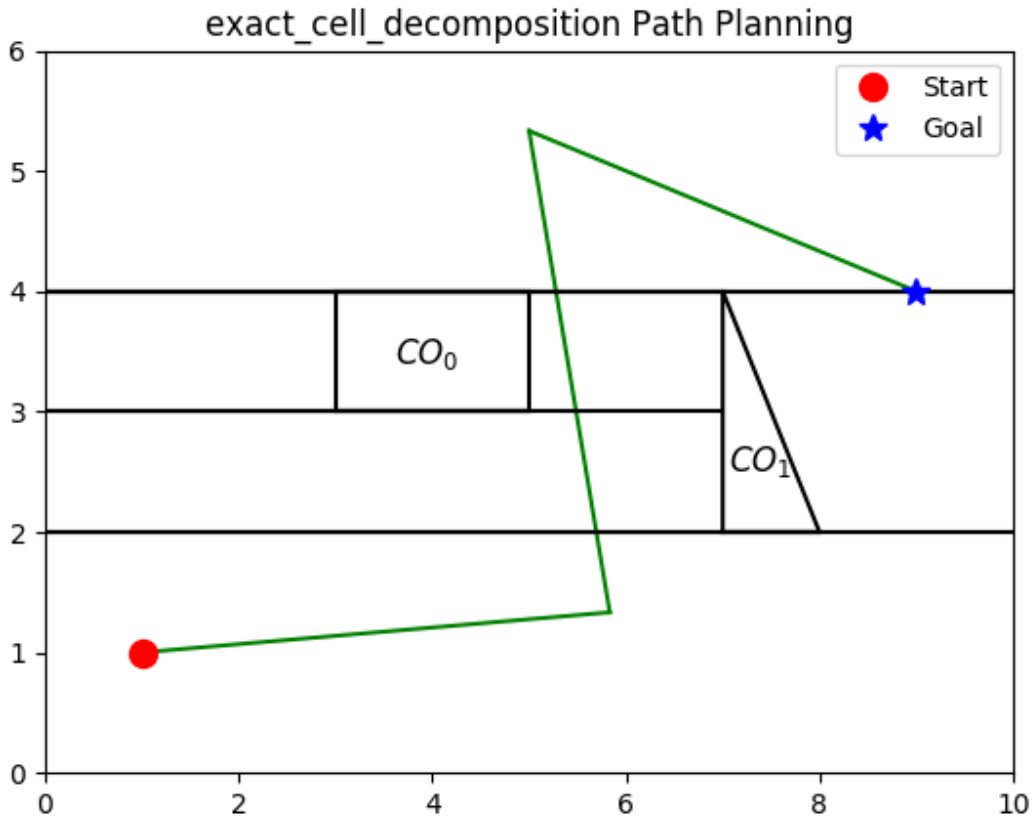

Fig. 1 Exact Cell Decomposition

II. Potential Field

a) For potential field (Figure 2) I chose the conic attractive force a repulsion over all the edges of my obstacle, and stochastic gradient descent. To overcome the local minimum at the top of the triangle I implemented a heuristic given by traveling perpendicular to the repulsive force, if there has been no progress for the last 5 steps, ie, if the total distance in 5 steps is less than some threshold.

b) Two possible changes I implemented is that there is no heuristic for checking if the robot got stuck, and the other is a quadratic attractive force instead of conic. For both of them the distance does not exist because I was not able to find a path. The reason is that if I did not have my heuristic, the robot would get stuck near the triangle and not complete the path. Then, if I had the heuristic, and had the quadratic attractive force, near the goal the robot would take a long time

(because of the small gradients) and not make progress, so the heuristic would make the robot go perpendicular and not reach the goal. The toal distance is 10.9 units.

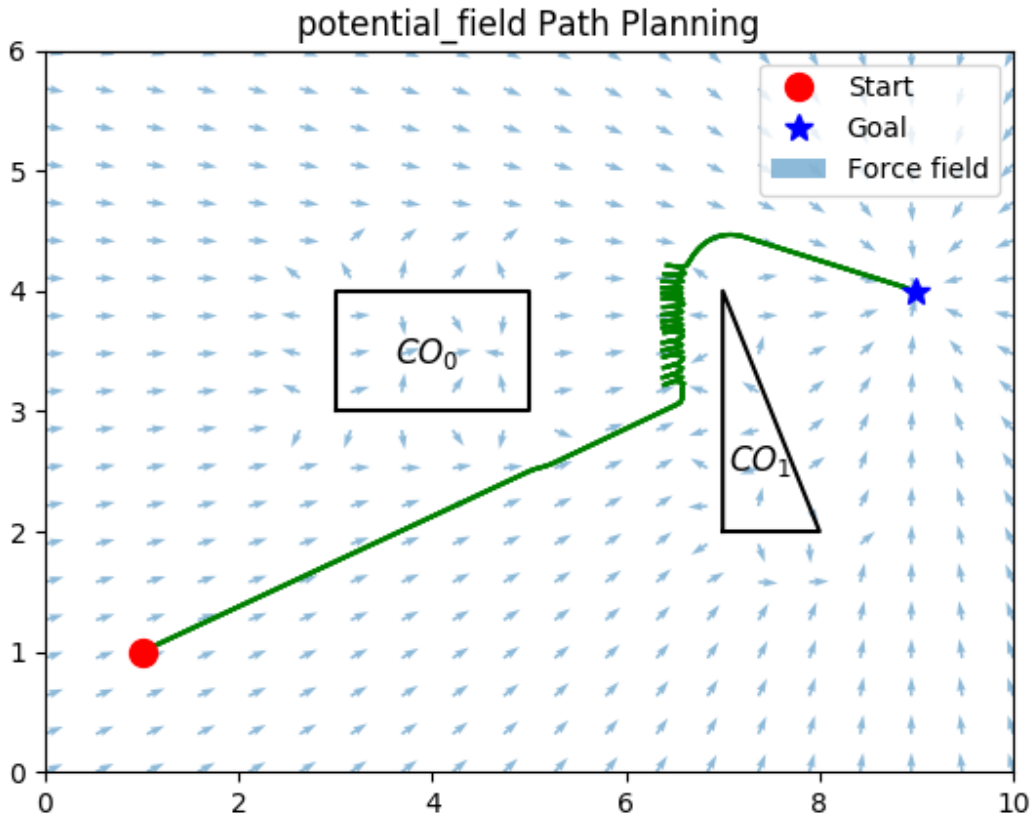

Fig. 2 Potential Field

III. Probabilistic Roadmap

a) For PRM (Figure 3) I sampled accross a uniform distribution inside the workspace of the robot. To connect the nodes, I implemented a k-Nearest Neighbors algorithm that connects the node to the kNNs based on euclidean distance. I used Dijkstra's for graph search. The total distance is 9.79.

b) The parameters I chose for this implementation is an upper bound of the max number of nodes to sample, I chose 50, as well as the k in kNN, which I chose 5. A possible limitation is that it takes much more time to produce an optimal path because of the closeness of the optimal path to the obstacles, as shown in the Figure 4 with distance 8.7 units, where I implemented the visibility graph algorithm. A positive is that it took the least amount of time to get a path.

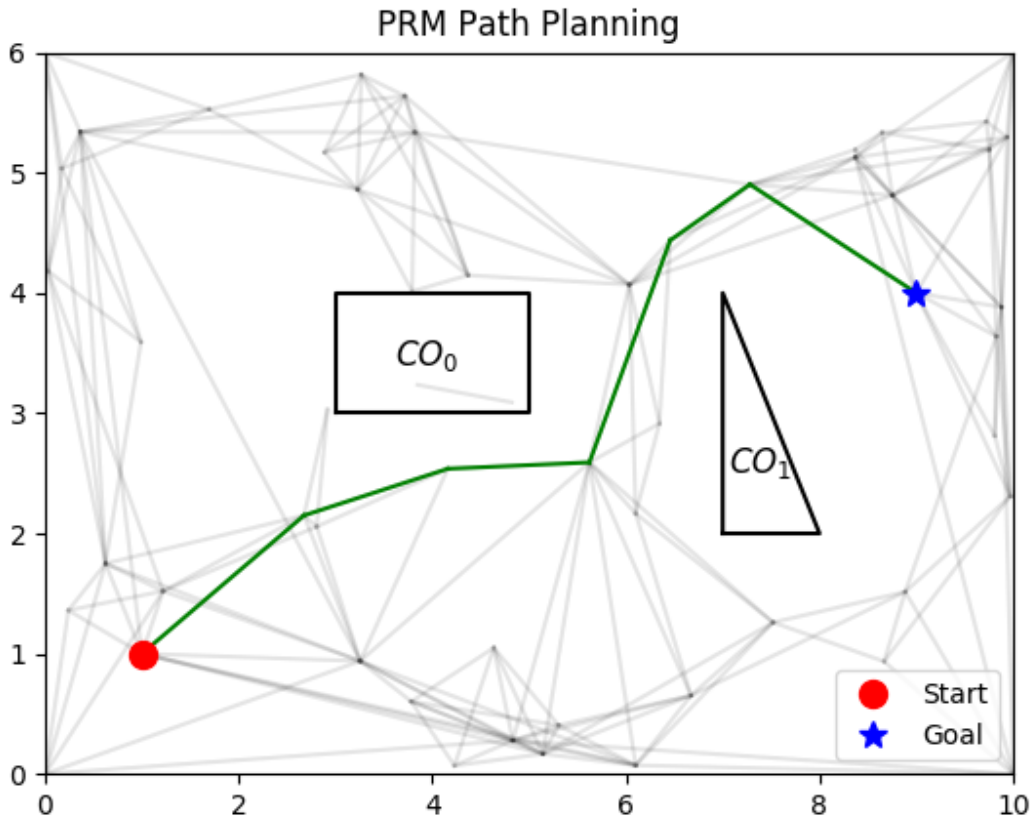

Fig. 3 PRM

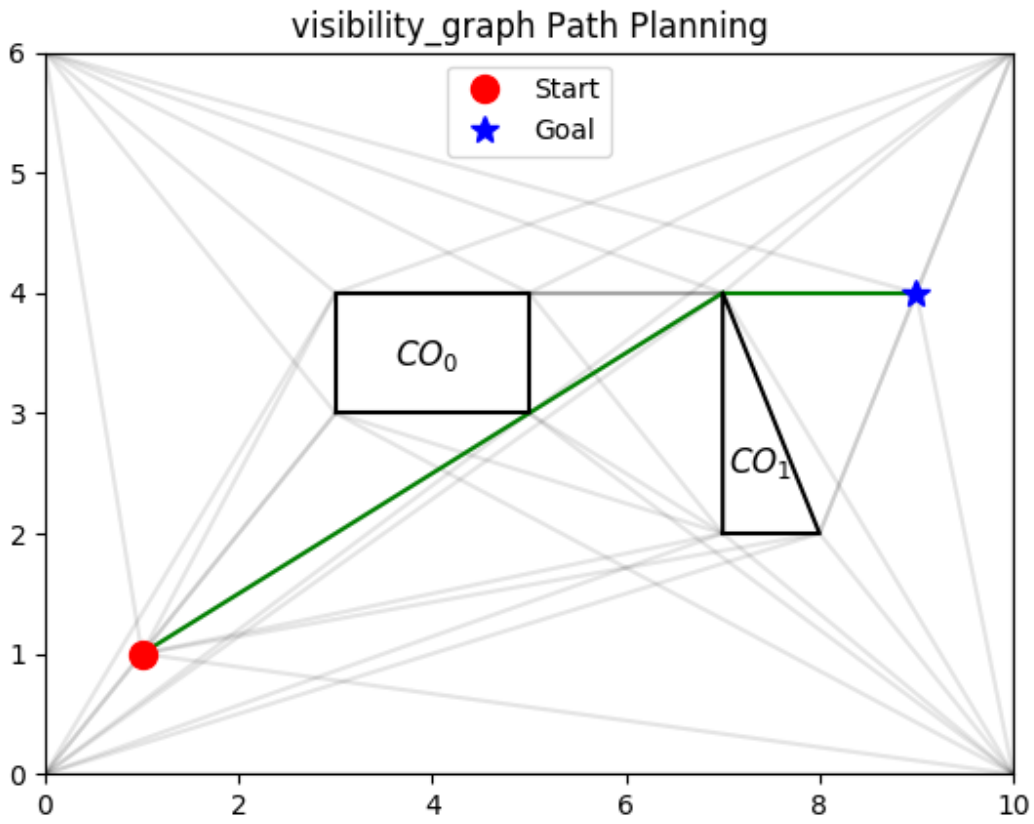

Fig. 4 Visibility Graph