Homework 1

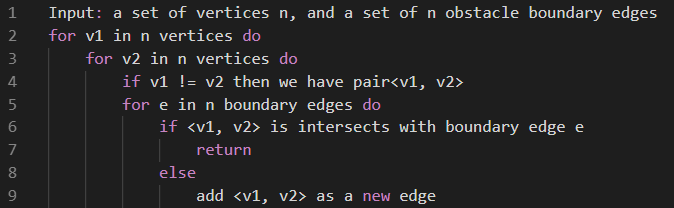
Haoran Liang

hl74

Sep 11, 2019

1. **Describe two trade-offs between the Bug 1 and Bug 2 algorithms.**
   1. **Time:** Time is one trade-off between Bug 1 and Bug 2 algorithm. For Bug 1, the robot will travel along the perimeter of the obstacle and record the minimum distance on the obstacle’s perimeter from the target point. Thus, robot needs to circle around the obstacle to know the shortest path. The more obstacles, the more distance a robot needs to travel. Different from the Bug 1 algorithm, Bug 2 algorithm will go along the line-m between the start point and target point. The robot will leave the obstacle once it back to the line-m and able to go straight to the target point. Therefore, under most cases, Bug 2 will use less time to reach the target point than Bug 1 algorithm.
   2. **Stability:** We know that Bug 1 will circle around an obstacle and find the shortest path from the obstacle towards target point. Thus Bug 1 is an exhaustive search algorithm which will look all choices and find the optimal one. This algorithm may sacrifice time but will have higher stability. We know that Bug 2 will travel along the intersection line-m towards target point. If the obstacle has multiple intersection points with line-m, then Bug 2 will create several inner loops when finding the shortest path. Therefore, the time spent will increase exponentially. However for Bug 1 algorithm, it will always circle around the obstacle no matter obstacle’s shape. Thus, Bug 1 has better stability and safer than Bug 2 algorithm.
2. **Visibility Graph**
   1. The upper-bound time complexity should be O(n3). Suppose we have n vertices, including start point and goal point. Thus, we could compose all vertices pairs such as <v1, v2> using O(n2) times. Then for each distinct vertices pair, we need to check if it is intersect with all obstacle boundary edges. For n vertices, there should be also n edges, and the intersection check function costs constant time. Therefore, the total time complexity should be O(n3).

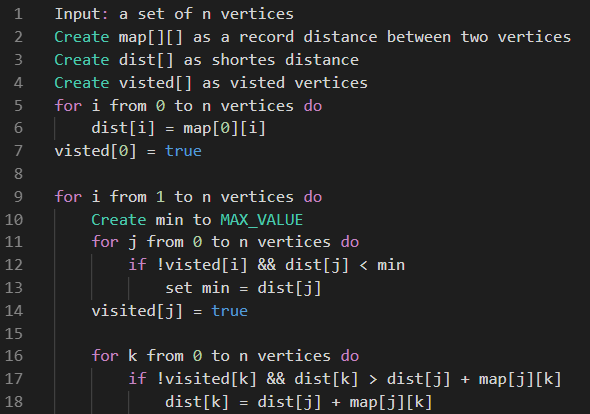
**Pseudocode:**



* 1. Yes, we can use **Dijkstra algorithm** to find the shortest path from start to goal.

For each vertices we will look for its shortest path to all other connect edges, and the upper bound of time complexity should be O(n2)

**Pseudocode:**



1. **Algebraic primitive**

We have single algebraic primitive *H* = { (x, y) | x2 + y2 <= 1}. Thus, we know that this primitive is a circle where the center is the origin. We assume there is a point *A* inside of the circle, where *A’* is the point after rotation.

Suppose: point *A* = (x , y) and angle between x-axis is *a*, and the distance from A to origin is *d*

point *A’* = (x’, y’) and angle of rotation is *b*, and the distance from A’ to origin is also *d*

Thus we have:

x’ = d \* cos(a + b) and y’ = d \* sin(a + b)

d = x / cos(a) = y / sin(a) so that x = d \* cos(a), y = d \* sin(a)

By trigonometric function we know that:

cos(a + b) = cos(a) \* cos(b) – sin(a) \* sin(b)

sin(a + b) = sin(a) \* cos(b) + cos(a) \* sin(b)

So that:

x’ = d \* (cos(a) \* cos(b) – sin(a) \* sin(b)) = x \* cos(b) – y \* sin(b)

y’ = d \* (sin(a) \* cos(b) + cos(a) \* sin(a)) = y \* cos(b) + x \* sin(b)

Now we can test if x’2 + y’2 <= 1:

x’2 + y’2 = x2cos(b) – 2\*cos(b)\*sin(b) + y2sin(b) + y2cos(b) + 2\*cos(b)\*sin(b) + x2sin(b)

= x2sin(b) + x2cos(b) + y2sin(b) + y2cos(b)

= x2 \* (sin2(b) + cos2(b)) + y2 \* (sin2(b) + cos2(b))

= x2 \* 1 + y2 \* 1 = x2 + y2

Thus, we proved that point A’ = (x’, y’) is inside of the circle after rotation.

1. **Two line intersection**

**Pseudocode**:

