W4733 ROBOTICS HW4

Jing Qian (jq2282)

Q1.

a) According to the authors, the main difficulties for C-space graph search methods are:

- C-space normally has six dimensions while only three or four dimentional spaces are tractable. It is very hard to search the six-dimensional C-space.
- Even if the resolution is small, the graph for the free space tends to have super many nodes.

b) In this paper, only 2-d plane with two degrees of translation freedom and one degree of rotation freedom is discussed, so they do not need to worry about the 6-d C-space searching difficulty. Also, the hierarchical cell decomposition techinique could alleviate the difficulty. This techinique decompose the C-space of the mobile robot into cells at successive levels of approximations and at each level of decomposition, the algorithm searches for a sequence of adjoining empty cells that consist a path from start point to target.

c) The hexagonal decomposition is an improvement on the original hierarchical cell decomposition, so it doesn't have similar drawbacks that graph search methods have. It is efficient and fast. It only visits part of the cells, calculates potentials of part of the cells and has flexible search step size, which means it only decomposes when necessary.

Q2

Implementation of the enlarge procedure:

1) Doing Step 1-3 in the "Find Collision Free Path Algorithm" on page 612.

2) Suppose the center is O, calculate the maximum distance from the obstacle vertices to the center O and call it d_{maxOb}.

3) Search path in the current region of influence using step 4-7 in the "Find Collision Free Path Algorithm". If one path is found, return it.

4) If no path is found, call the radius of current region of influence as R. if $R = d_{maxOb} + t$ or the four boundaries of map are reached, returns "no path". Because there is a path on the outside ring with width t. If the robot could reach this ring, it will find a path. If it could not, no matter how large the region of influence is, it will never find a path. Also, if the robot has

reached the whole map and found no path, there is no path then.

5) If no path is found and $R < d_{maxOb} + t$, compare following two values: $2R$ and $d_{maxOb} + t$ where t is the minimal width for a path.

If $2R < d_{maxOb} + t$, we enlarge the radius of region of influence as $2R$; otherwise, we enlarge the radius of region of influence as $d_{maxOb} + t$.

Go to part 3) and do search.

Q3

The task is to test whether an obstacle vertex falls into the region bounded by the dashed lines show in Figure 8b. The region bounded by the dashed lines are composed of a semicircle centered in the center of Hexagon B and with radius equlas to $(t/2)$ where $t = \sqrt{l^2 + w^2}$ and a rectangle whose lengths are the two parallel line segments in Fig 8a.

We could solve this tasks in two steps: check whether an obstacle vertex is in the circle centered in the center of Hexagon B and with radius equlas to $(t/2)$; check whether an obstacle vertex is in the rectangle whose lengths are the two parallel line segments in Fig 8a. The overlapping semicircle region in the two steps is for computational efficiency.

Implementation: (Suppose the vertices on the dashed lines also cause the hexagon to be IMPASSABLE)

a) Check whether there is a vertex inside the circle: using loops to check the distance between each obstacle vertex and the center of Hexagon B. If any distance is less than or equal to $(t/2)$, the vertex is inside the region and the hexagon is IMPASSABLE.

b) If there is no vertex inside the circle, check whether there is a vertex inside the rectangle using the crossing number algorithm. Using loops to test how many times a ray, starting from each obstacle vertex and going to the right direction, intersects the edges of the rectangle. If the right-heading ray from any vertex intersects the edges of the rectangle once, the vertex is inside the region and the hexagon is IMPASSABLE.

If all the obstacle vertices pass the two steps of checking, the hexagon is PASSABLE.

Q4

No, this strategy would not lead to an incorrect outcome or a worse overall path.

If we just search without marking already-visited PASSABLE hexagons as IMPASSABLE, we may get trapped in local minima. Like the Figure 10b, following the search strategy, the robot moves from P1->P2->P3->P4->P5. At P5, without setting P1 as IMPASSABLE, the robot may still find P1 as the neighbour with the lowest potential and goes back to P1 again. Following the same search strategy, it will repeat the movement from P1->P2->P3->P4->P5->P1, get trapped in local minima and never find a path, not to mention a better path.

On the other hand, when at P5, if the robot find a better neighbour, let's say P6, which has lower potential than P1. It will go to P6. Setting P1 as IMPASSABLE won't prevent robot from going to P6. Setting P1 as IMPASSABLE only prevents robot from local minimal.

To sum up, the strategy of marking already-visited hexagons as IMPASSABLE won't lead to an incorrect or worse outcome.

Q5

a) Because the target is in the left side of the region while obstacles are distributed on both left and right side, the potential function would lead the robot towards the dead end on the left-hand side.

b) From the path shown in Fig 12, we could see that the horizontal dark shaded obstacle is the reason that the robot trapped in a local minima while the second half of the path works quite well. It suggests that if we modify the potential function properly for the first half path and keep the original potential function for the second half path, we may get a better path without being trapped in an initial local minima. Our basic idea is for the location below the horizontal dark shaded obstacle j, the potential has an increased repelling potential from obstacle j; for the location above the horizontal dark shaded obstacle j, we keep the original potential function.

Suppose the dark shaded obstacle is horizontal and parallel to the x-axis (even if it doesn't, we could always rotate our function to do the following modifications) and the y coordinate of the lower edge is y_j, we modify the potential function as:

$$\text{Potential}((x,y)) = \begin{cases} \sum_{i=1, i\neq j}^{n} k_1 \frac{1}{r_i^2} - C(1-k_2\alpha) + k_3 \frac{1}{r_j^2}, & y < y_j \\ \sum_{i=1}^{n} k_1 \frac{1}{r_i^2} - C(1-k_2\alpha). & y \geq y_j \end{cases}$$

where $k_3 > k_1$ and should be tuned to get a proper value.