# Solutions for HW1

Yunhai Han

January 10, 2020

# 1 A switching Bug 0 algorithm

## 1.1 i

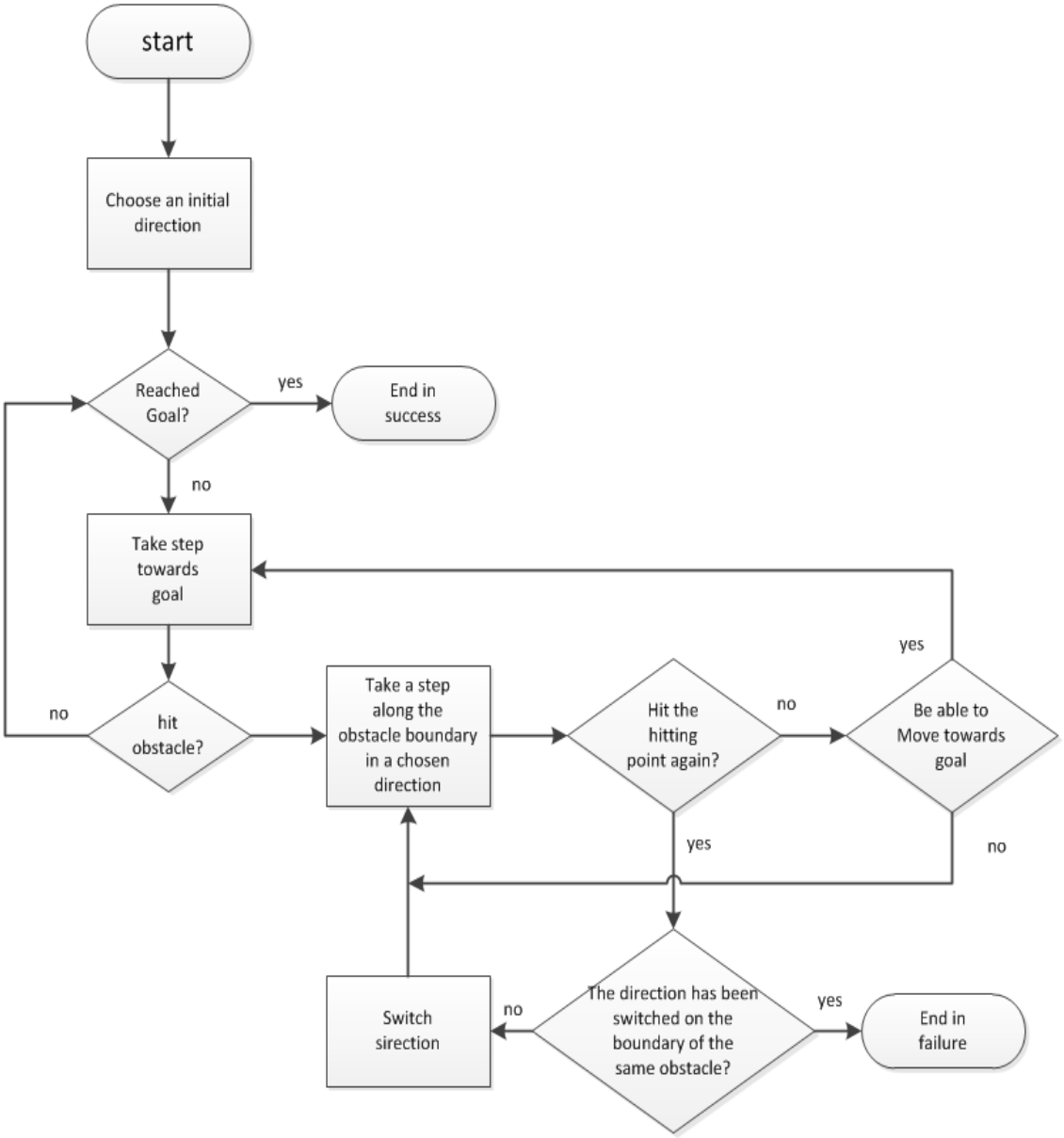


Figure 1: Flowchart

I draw the flowchart as shown in Figure 1.

## 1.2 ii

asdasd

### 1.3 iii

## 2 The Bug 2 algorithm

### 2.1 i

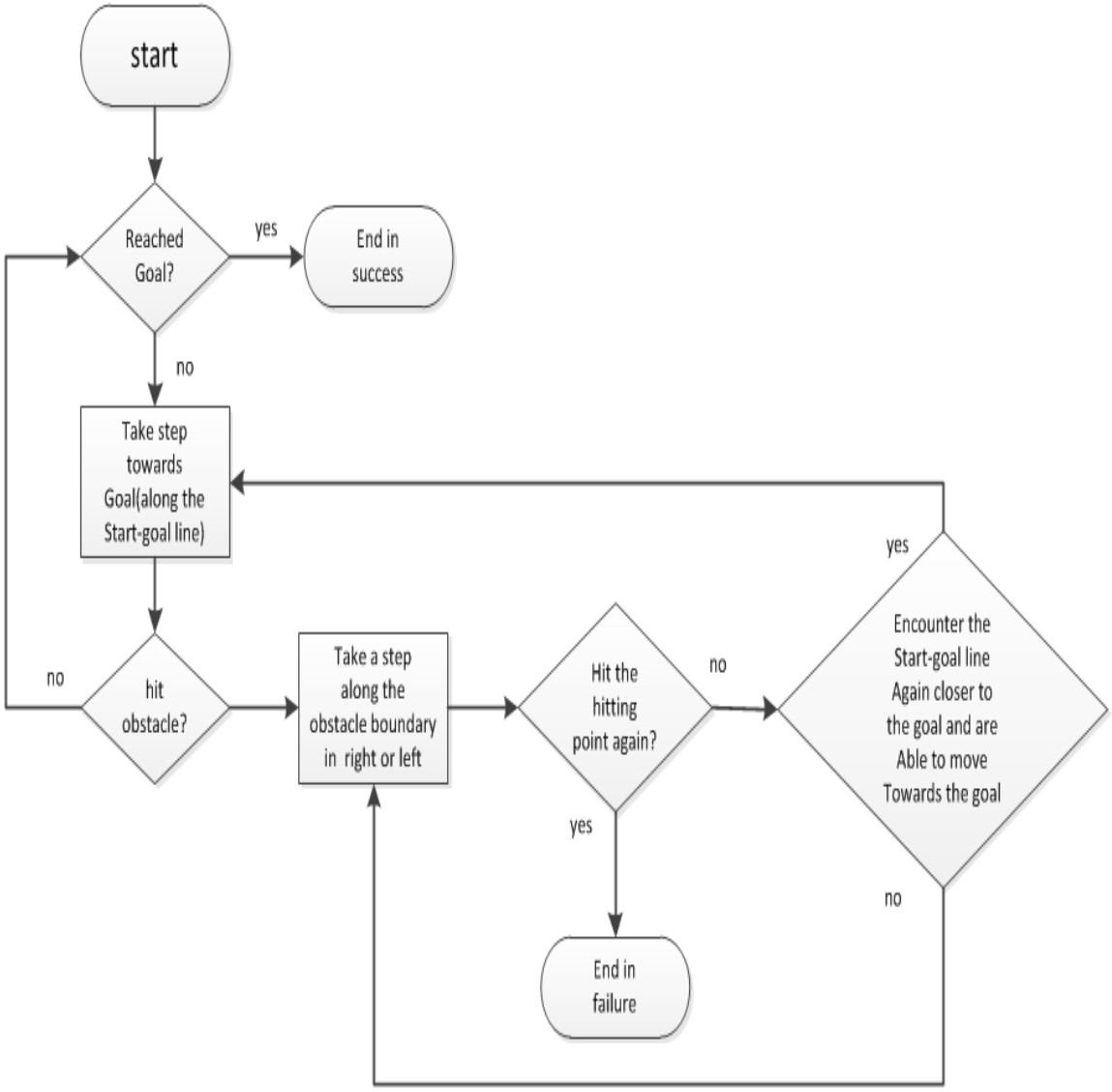


Figure 2: Flowchart

I draw the flowchart as shown in Figure 2.

## 2.2 ii

For the second part, I think if the robot chooses to turn into different directions when it hits the obstacle, the results could be totally different. I will show all these circumstances.

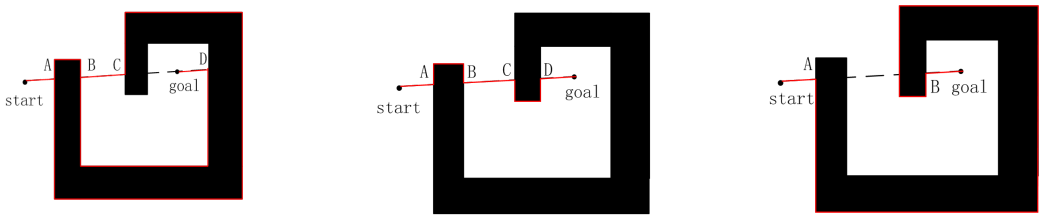


Figure 3: Robot's path using Bug 2 algorithm

From the above figures, the left one shows the result when the robot would turn left each time it encounters an obstacle; the middle one shows the result when the robot would first turn left and then turn right at different hitting points on the obstacle; the right one shows the result when the robot would turn right and in this case, it would only encounter the obstacle once. By comparison among these three figures, it seems that although Bug 2 algorithm could successfully find a path towards the goal for any turning directions, they greatly pose an effect on the amount of time for the robot to reach the goal. This means the different choices could bring in different results and only one of them is the best one for a certain workspace configuration.

## 2.3 iii

In this part, I am required to sketch a graph of the distance between the robot and goal location along the path. In order for the convenience, I just take the left figure as an example to show how I handle this problem and the solutions of the other figures could be obtained in the same way.

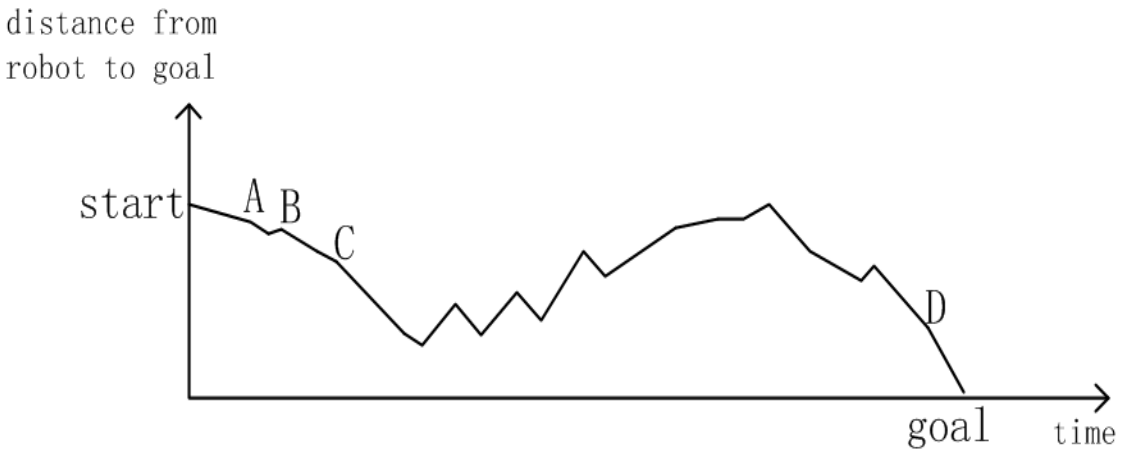


Figure 4: The graph of distance

From the figure 4, you can see the continuous function itself(with search phase) is not a monotonic function, but if we select the values at hit points and leave points and together form a new discrete function, it is always a monotonically decreasing function.

### 2.4 iv

The reason is really simple. Each time the robot encounter the obstacle, it is only allowed to leave the obstacle at the leave point which makes it closer to the goal. If it can not find such a point, it means there does not exist such a possibly path between the start point and the goal point, which contradicts to what the question assumes. For the same reason, the robot could not hit the same obstacle twice because the distance function at each hit points and leave points is a monotonically decreasing function. Besides, there are finite number of obstacles in the environment, so the distance could finally decease to a certain value at the leave point of the obstacle which is closest to the goal point. if the path really exist, there must be some free space around the goal point. In other words, the robot could always reach the goal point along the start-goal line starting from the leave points of the closest obstacle. Hence, it could finally reach the goal point.

## 3 Programming:Lines and segments

### 3.1 computeLineThroughTwoPoints

It is easy to derive the analytic formulas as I show below:

$$\begin{cases} x_1a + y_1b + c = 0 \\ x_2a + y_2b + c = 0 \\ a^2 + b^2 = 1 \end{cases} \tag{1}$$

$$\begin{cases} (x_1 - x_2)a + (y_1 - y_2)b = 0 \\ a^2 + b^2 = 1 \end{cases} \tag{2}$$

From the above equations, we can easy obtain the answers.

$$\begin{cases} a = \frac{1}{\sqrt{1+(\frac{x1-x2}{y1-y2})^2}} \\ b = -\frac{x1-x2}{y1-y2} * a \\ c = -x1 * a - y1 * b \end{cases} \quad \begin{cases} a = -\frac{1}{\sqrt{1+(\frac{x1-x2}{y1-y2})^2}} \\ b = -\frac{x1-x2}{y1-y2} * a \\ c = -x1 * a - y1 * b \end{cases} \tag{3}$$

The correctness of the answers could be esaily done.

### 3.2 computeDistancePointToLine

From the above part, we could obtain the analytic formula of a line by point $p_1$ and point $p_2$. As we learnt from high school, the distance($D$) from a point($p = (x1, y1)$) to a line($ax + by + c = 0$) could be obtained through a simple formula:

$$D = \frac{\|ax_1 + by_1 + c\|}{sqrta^2 + b^2} \tag{4}$$

### 3.3 computeDistancePointToSegment