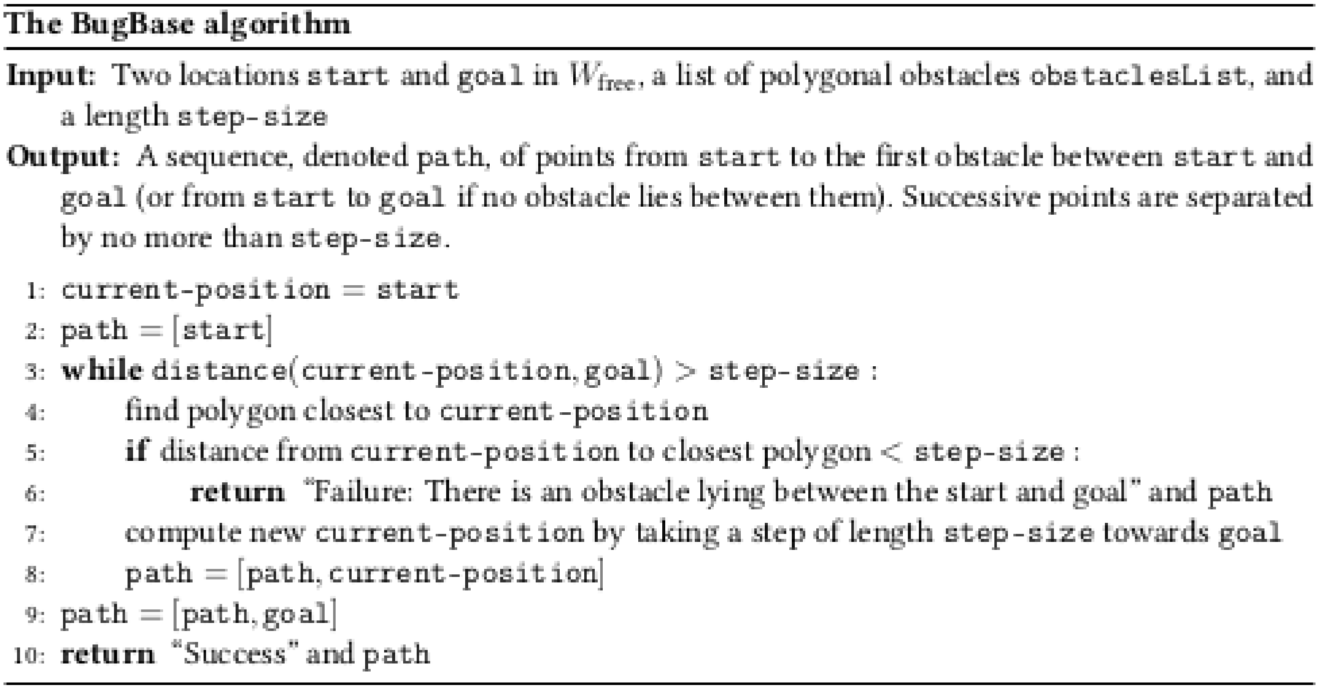
SC-627 Assignment 1

Dikshant 180040033

E1.8: Programming project: The Bug 1 algorithm

(i) the Bug Base algorithm



The BugBase algorithm

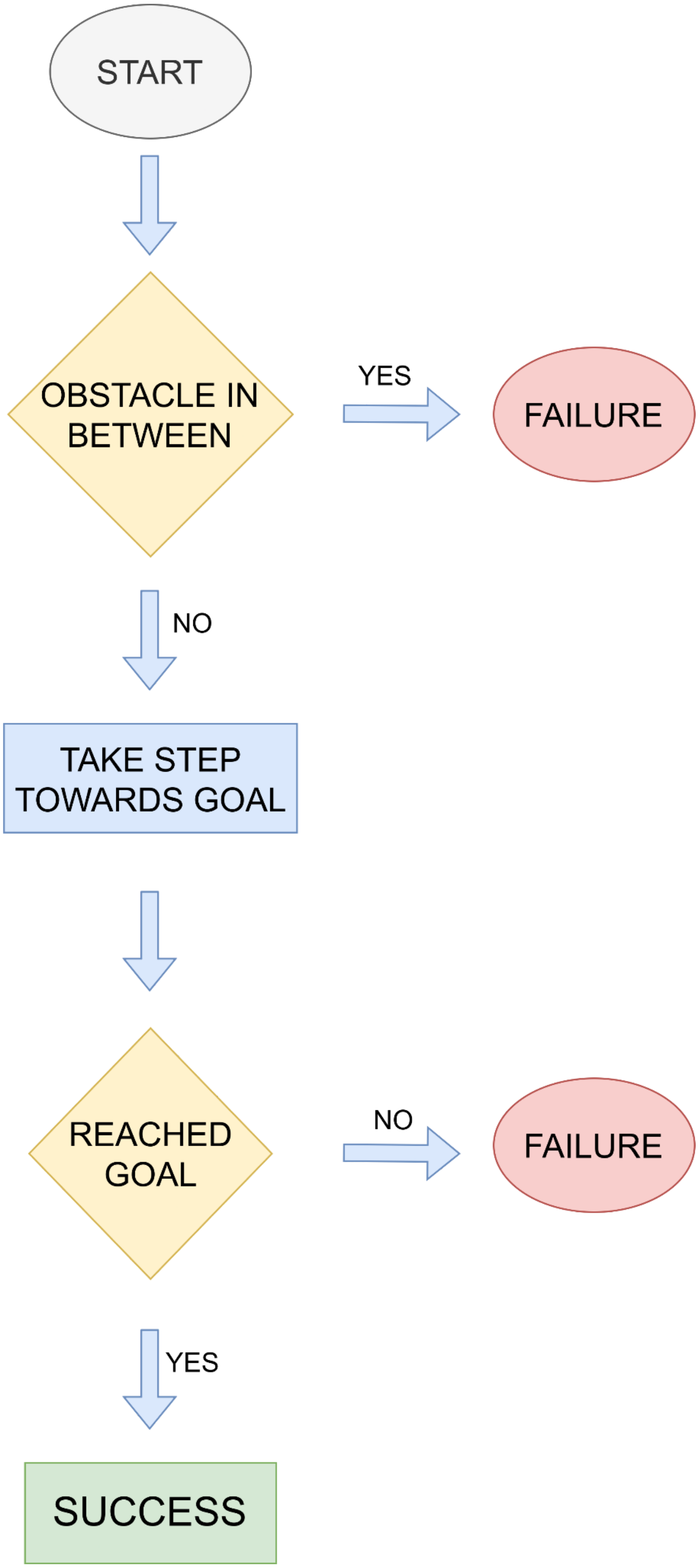
Input: Two locations start and goal in W_{free}, a list of polygonal obstacles obstaclesList, and a length step-size

Output: A sequence, denoted path, of points from start to the first obstacle between start and goal (or from start to goal if no obstacle lies between them). Successive points are separated by no more than step-size.

```
current-position = start
path = [start]
while distance(current-position, goal) > step-size :
    find polygon closest to current-position
    if distance from current-position to closest polygon < step-size :
        return "Failure: There is an obstacle lying between the start and goal" and path
    compute new current-position by taking a step of length step-size towards goal
    path = [path, current-position]
path = [path, goal]
return "Success" and path
```

Algorithm Description: Move forward a step size towards the goal until you reach it. If an obstacle is encountered in between then terminate the algorithm.

A typical flowchart of bug base algorithm can be depicted as follows:


START
OBSTACLE IN BETWEEN
YES
FAILURE
NO
TAKE STEP TOWARDS GOAL
REACHED GOAL
NO
FAILURE
YES
SUCCESS

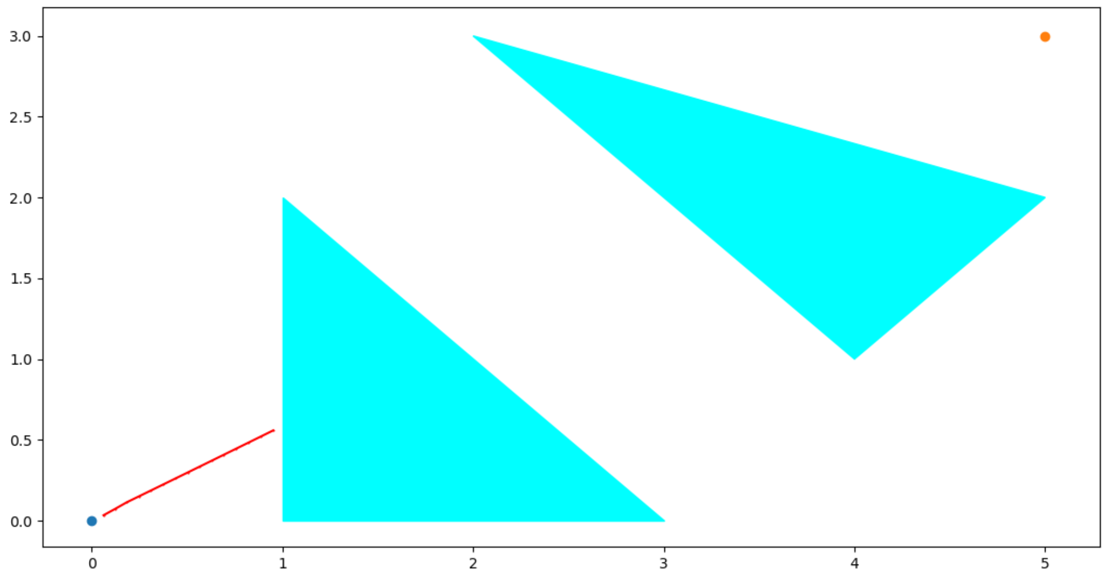
(ii) Modification in BugBase to Bug1: Instead of terminating the algorithm when we encounter an obstacle, we will circumnavigate around the obstacle and find the best leave point from the obstacle to reach the goal.
I have used the *computeDistancePointToPolygon()* function to compute the distance from the obstacle. It will return the minimum distance, a parameter representing whether the minimum distance is from an edge or from a vertex and the closest obstacle. Later on, I have used the *computeTangentVectorToPolygon()* function to estimate the unit tangent vector from the obstacle so that the bot can start circumnavigating around the obstacle. Now, in the main *bug1.py,* I have defined two more functions namely, *circumnavigate()* and *leave()* which will help the bot in circumnavigating and leaving from the best point respectively.

(iii) Code submitted on GitHub

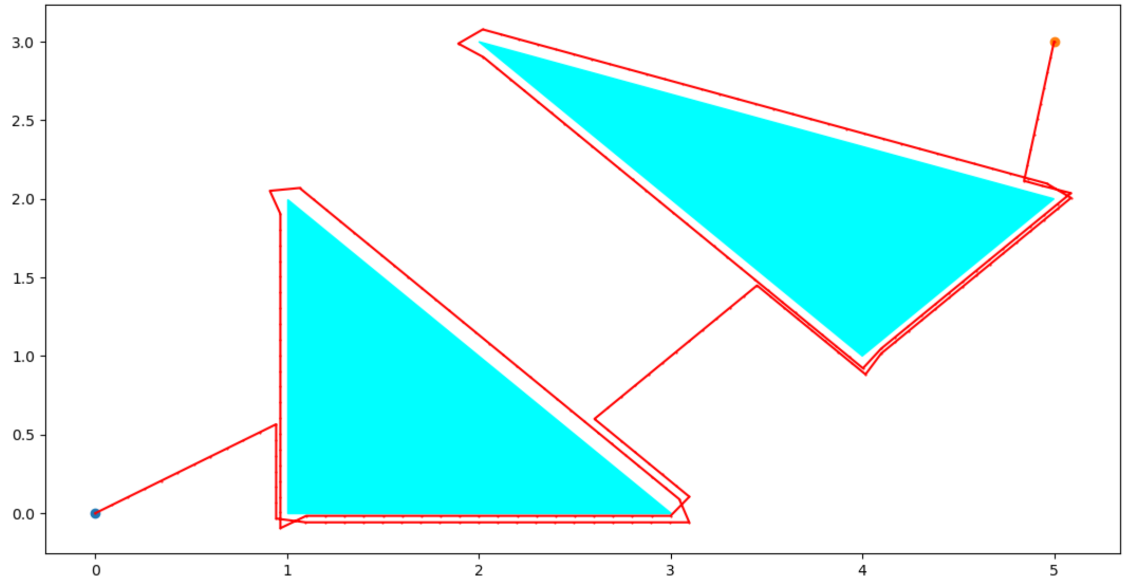
(iv) Code submitted on GitHub

Results:

Path Traced by BugBase algorithm:

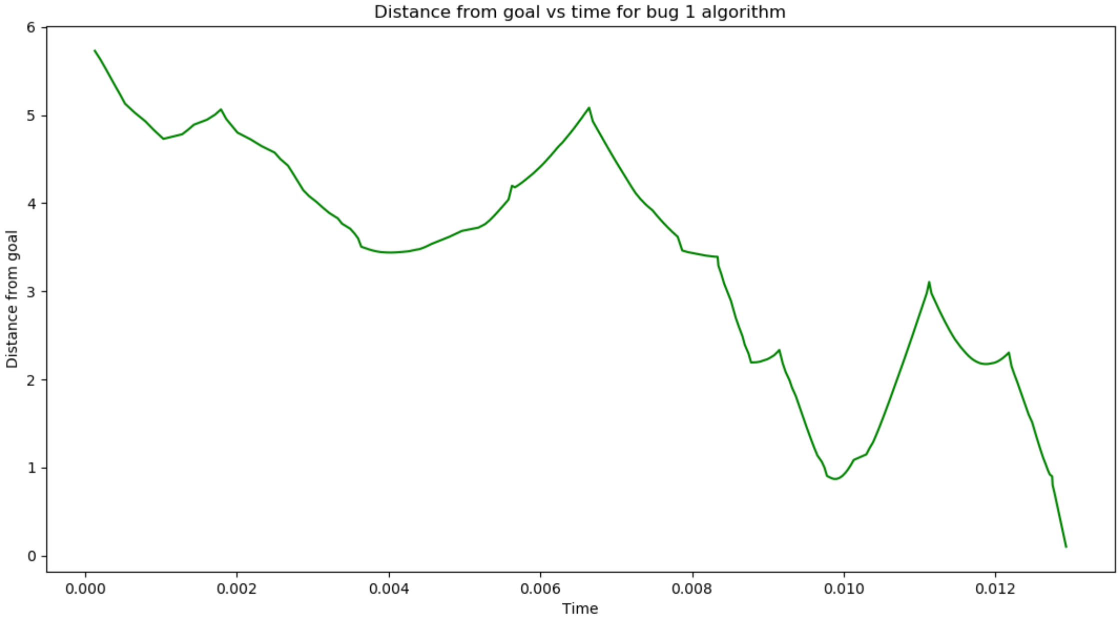


The path traced by Bug1 algorithm:



Total Path Length: 25.4 m
Computing Time: 503 sec

The plot of the distance from the bug's position to the goal as a function of time.


Distance from goal vs time for bug 1 algorithm
Distance from goal
Time