

ME 145
Robotic Planning and Kinematics

Lab Session No. 2

BUG 1 ALGORITHM

Instructions

Submit your code through iLearn. Your code and reports are due on Wednesday, 10/25, at 5:59 pm. Late submissions will not be accepted.

E1.8 Programming Project: The Bug 1 algorithm (80 points).

Requires Exercises (E1.6) and (E1.7). In this programming project, you are asked to implement the Bug 1 algorithm. For your convenience we provide below a detailed pseudocode implementation of a simple Bug algorithm called BugBase; this pseudocode implementation contains all detailed steps required to execute a bug algorithm, but does not have any logic for getting around obstacles.

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.

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

Perform the following tasks and document them in a concise project report:

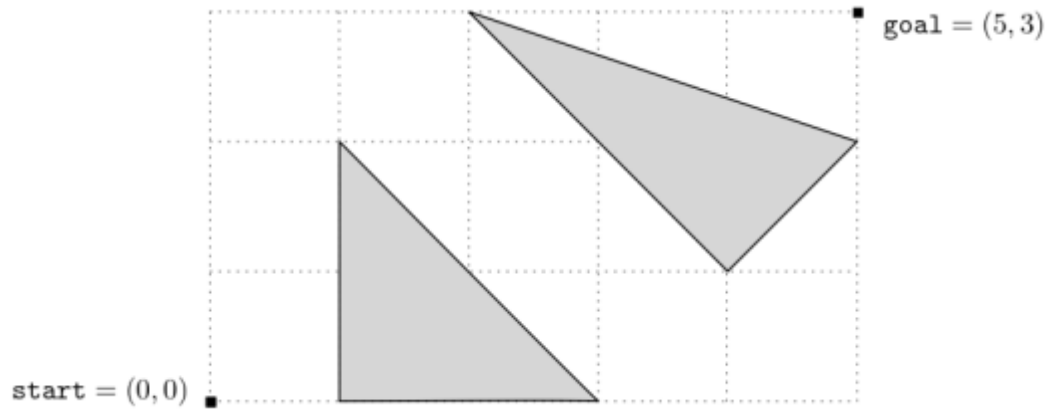
- (i) (15 points) Sketch a flowchart and implement the BugBase algorithm.
- (ii) (5 points) Describe in a paragraph how you will modify BugBase to implement the Bug 1 algorithm. Explain the roles of the geometric functions in Exercises (E1.6) and (E1.7) and what new logic will be needed.
- (iii) (40 points) Implement a fully-functioning version of Bug 1, described as follows:

computeBug1

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 goal or returns an error message if such a path does not exist. Successive points are separated by no more than step-size and are computed according to the Bug 1 algorithm.

(iv) (20 points) Test your program on the following environment: start = (0, 0) and goal = (5, 3) step-size = 0.1
obstaclesList = {(1, 2), (1, 0), (3, 0)}, {(2, 3), (4, 1), (5, 2)}



Include in your report a plot of the path taken by your bug from start to goal, the total path length and the computing time, and a plot of the distance from the bug's position to the goal as a function of time.