CS-657 Intelligent Systems
Assignment No. 1
Due September 19, 2022 at 2:00 PM

You are to design a rule-base expert system for a mobile robot to move from its current position to a goal position while avoiding obstacles. Suppose the terrain (surface that the robot moves on) is represented by the array A, divided into m by n square cells, and at any time the robot is in one of these cells with (x, y) coordinates, where x = 0, 1,2,…,m-1 and y = 0,1, 2, …, n-1, and x =y=0 is the upper left corner cell, and the x and y axis are as shown in Fig 1. When the robot is in position (x,y), it can move to one of the three cells "in front of it", as shown in Fig. 1(a). Therefore, when the robot is in the cell (x,y), as shown in Fig. 1(a) shown as a hollow (white) shape, it can move to position (x,y+1) with no rotation of its axis (i.e. no steering), to (x-1, y+1) or (x+1, y+1) with 45 degrees rotation (steering) as shown with gray shapes. Similarly if the robot is in (x+1,y+1) with it axis rotated 45 degrees, it can go to one of the three cells in front of it with orientations as shown in Fig. 1 (a) with black shapes. It can also rotate +/- 45 degrees on the spot, i.e. without moving to another call. Note that in any cell, the robot can have 8 different orientations. In addition it can reverse and go back to the previous position and orientation. Note that the robot can rotate in multiples of +/- 45 degrees on spot, but each 45 degree rotation is considered as one move.

The robot has three sonar sensors that detect obstacles in their line of sight along the robot axis and along the +/- 45 degree angle as show is Fig 1(b). Therefore it will detect obstacles shown in red circles, but will not be able to detect obstacles shown in black circles. This is due to the fact that the sonar can only detect the first obstacle it sees, but on it left or right or not behind it

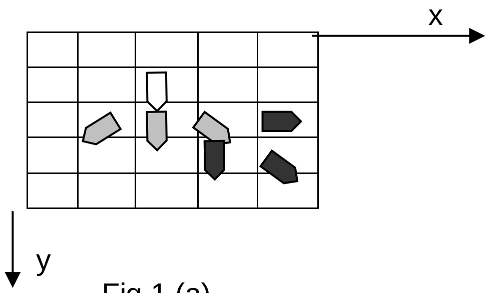

Fig 1 (a)

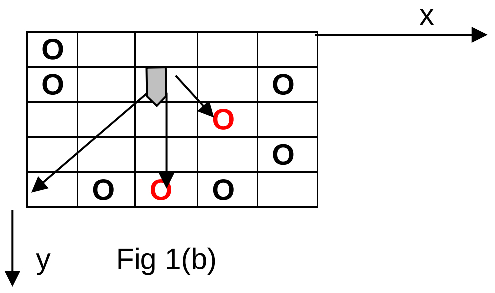

Fig 1(b)

The goal of the robot is to move from a start cell to a specified destination cell <u>without hitting any obstacle, and to make as few moves as possible</u>. A move to a front cell, back to previous location, or each 45 degree rotation on spot, is considered as one move. The **data base** (array of locations) initially contains the location of the robot and its orientation, and the values of m and n, and the location of the goal cell. All other cells are assumed unknown. The array A[x,y]=0 if the cell blocked (contain obstacles), A[x,y]=1 if the cell is free, and A[x,y]=2 if it is yet unknown. Each time obstacle sensing takes place, or the robot moves, data base is to be updated, that is A[x,y]=2 becomes either A[x,y]=0 or A[x,y]=1. The actual environment is represented by the array B[x,y], which whose cells are either B[x,y]=0 or B[x,y]=1. This array is used by the sensor to update the array A after sensing takes place. (See below as to how B is generated).

2	2	2	2	2	2	2
2	S	2	2	2	2	2
2	2	2	2	2	2	2
2	2	2	2	2	2	2
2	2	2	2	2	2	2
2	2	2	2	2	D	2

A. Initial robot map

1	1	1	0	0	1	1
1	S	1	1	1	0	0
1	1	1	0	0	1	1
0	1	1	0	1	0	1
1	0	0	1	1	1	1
1	0	1	1	0	D	0

B. Actual map (not known to robot)

2	2	2	2	2	2	2
2	S	2	2	2	2	2
1	1	1	2	2	2	2
2	1	2	0	2	2	2
2	0	2	2	2	2	2
2	2	2	2	2	D	2

C. After robots senses the environment.

Assume that you are the "expert" in devising rules for the robot motion. You are to design the rule base in the form of IF <condition> THEN <action> rules so as to achieve the goal of the robot which is the minimum number of move to reach the destination. Now perform a computer simulation as follows:

1. Program the rules, and then use the inference engine to move from one square to another to get to destination in as few moves as possible moves. Record the number of moves, locations that the robot has visited (for example using a stack). Remember that there are two main aspects to each move: goal seeking (i.e. move towards the destination) and collision avoidance (i.e. avoid the obstacles, go around them, etc.).
2. Create three actual environments, B[x,y] in a m=45 by n=35 grid by randomly setting some of the cells to 0 (i.e. an obstacle), (a) an environment with 10% of randomly picked cells having obstacle, (b) with 20%, and (c) with 45% obstacles. Let the initial position of the robot be at (7,5) and the goal be at (40,30). The initial position and orientation and goal position are known to the robot. Note that the environment is not initially known to the robot, but as it moves it will gather information about the obstacles. The grader will test your program with her own randomly generated environments.
3. Please submit your program and a short report (e.g. 2-3 pages) about your rule base, any interesting feature of your program, experiments you performed, results you obtained and a discussion and conclusions. Place submit to Canvas your assignment containing your program, executable and a readme file if needed, together with a report.

Note 1: You must develop your program in the rule-base expert system setting consisting of a rule base, and a data-base (map of the environment discovered so far), After each move you must go through the if-then rules sequentially and use the data-base to determine which rule is fired (see forward and backward chaining).

Note 2: This assignment leaves a lot for you to explore. For example you may want to:

(a) run the simulations for several environments, different initial positions of the robot, and find the average number of moves. Then apply Dijkstra's shortest path algorithm (assuming everything is known about the environment) and compare your result with it.

(b) develop graphics to show the moves

You will earn extra credit for any of the above (a) and (b), and for other exceptionally good work.