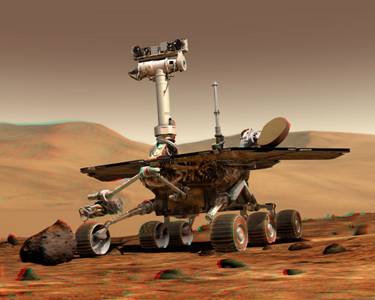
**Robotics I – Project 1 Fall 2020**

**Jumping Right In – Competition: Navigating a point robot on a previously unknown map**

[](https://www.google.com/url?sa=i&rct=j&q=&esrc=s&source=images&cd=&cad=rja&uact=8&ved=2ahUKEwjUkKrqsYHdAhVRRqwKHd30BqkQjRx6BAgBEAU&url=http://www.cs.cmu.edu/~reliability/&psig=AOvVaw2hamXlLTl8lC22tDj-rsT9&ust=1535052362299319)

**Project Assignment:** Given a previously unknown 100 x 100 meter map containing 4 walls and several obstacles (provided immediately before project submission), devise a method to navigate a point robot to a pre-defined goal location given a pre-defined starting location. Methods can be either “deliberative” or “reactive” based

**Competition:** Submit your algorithm as either deliberative or reactive and use it to move to the goal in fewer moves than your classmates to receive bonus points.

**Map Details:** The provided matlab code contains example binary occupancy maps (OccupancyMap\_v1 and OccupancyMap\_v2) which should be used to develop and verify your working algorithm. Binary occupancy maps contain a 2D array of binary values which correspond to geometric locations in a robotic workspace. The value of 1 represents an obstacle at that location and a value of 0 represents free space at that location.

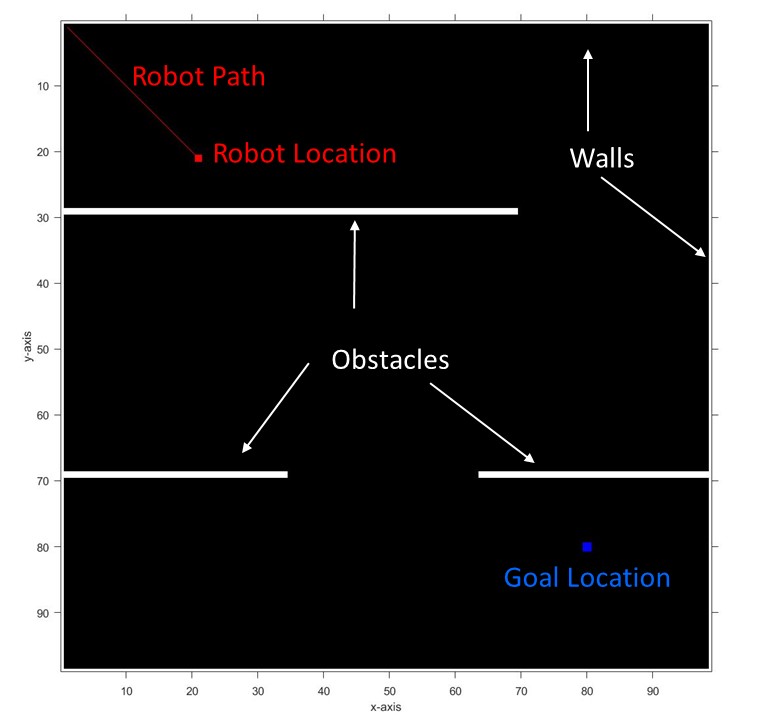
Immediately before the demonstration and competition, a new and previously unknown map will be emailed out. This map will contain the same basic structure as the example binary occupancy maps but will have a different layout of obstacles and different starting and ending locations.

**Provided functions:** The provided Matlab code contains functions to plot the occupancy map and the location of the goal and robot. It also contains functions to move the robot and to test a movement of the robot. These functions should be used instead of user defined functions.

**Rules and Notes:**

1. The robot can only be moved with the provided moveRobot() Matlab function and this function cannot be altered.
2. The pathLength variable can only be set as an output to the moveRobot() function as provided in the example.
3. An Occupancy Map will be provided at the time of the competition. This map can be queried by the program but cannot be altered by the program.
4. All moveRobot() calls must be done in the provided while loop. However, other code may be inserted anywhere in the provided script as long as it doesn’t affect the competition result.
5. The robot can only be moved one discrete cell at a time in any of the cardinal or diagonal directions.
6. The robot cannot move through the walls or any obstacles.
7. There will be a competition containing two major and distinct categories, reactive algorithms and deliberative algorithms.
8. The results will be judged and the student who can navigate to the goal location with the smallest path length (or movements) given the appropriate category will win the competition and receive bonus points. If there is a tie between winners, the extra points will be evenly distributed between winners.
9. A createMaps.m script is provided that can be used to create new maps given you figure out how to change the code and save and load the maps.
10. The professor reserves the right to disqualify any unethical hack of the code which gives an unfair advantage. However, this right will be used only under limited circumstances.

**Example Map:**

****