// Nawaf Abdullah

// CMPSC 201 SU 2016

// Date 8-2-16

// HW #10

// Robot Simulation Using Sonar

**1.0 Objective**

* Description of the objective (problem statement)

**2.0 Obstacle course**

* A screenshot of the of the maze used for the simulation

**3.0 MATLAB script**

* The m-file script used for the MATLAB simulation with comments

**3.0 Screenshots of test trials**

* Test case #1
* Test case #2
* Test case #3

**4.0 Conclusions**

* Conclusions about the performed simulation

**1.0 Objective**

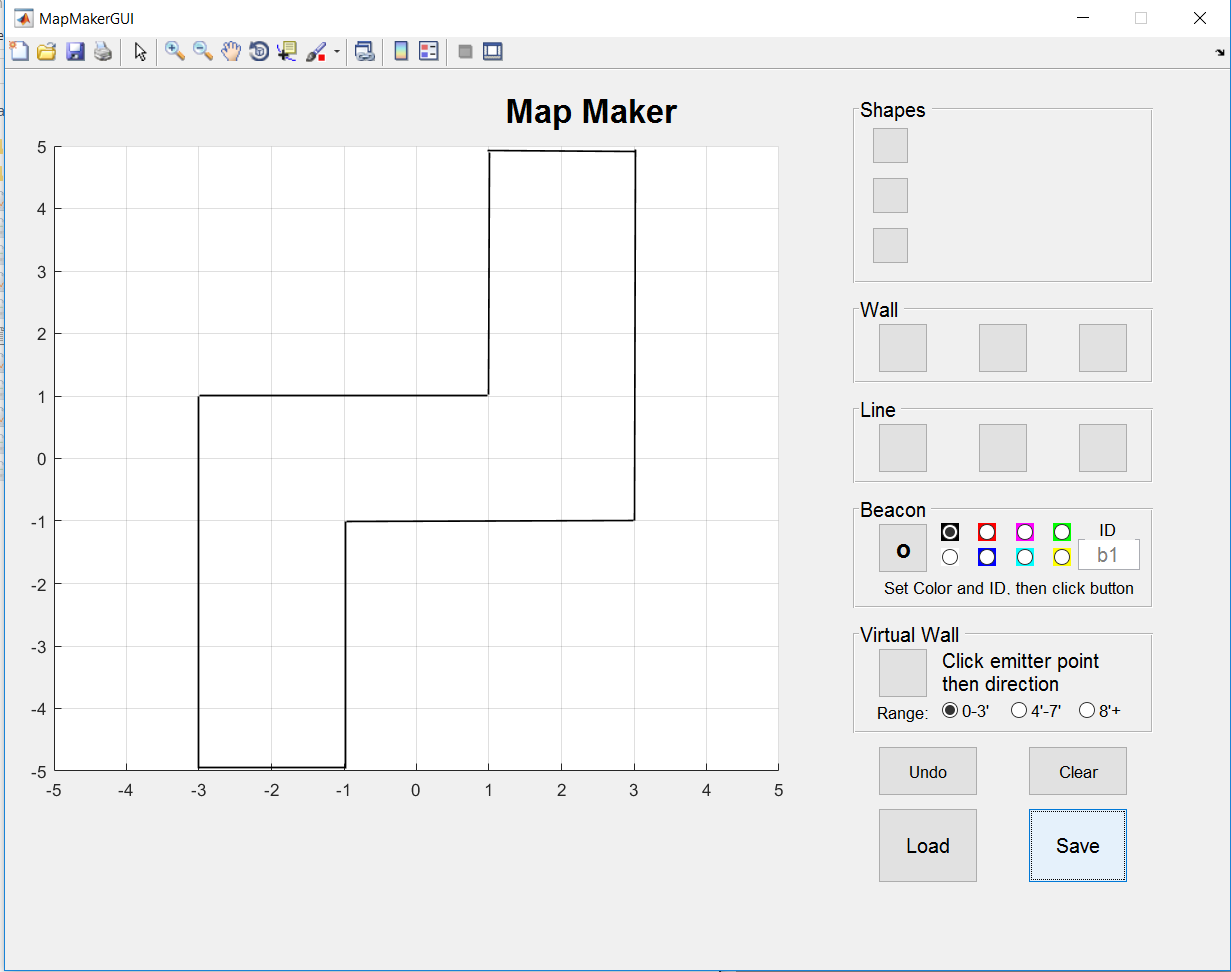
* Description of the objective (problem statement):

In this MATLAB simulation, sonar sensors will be used to navigate a robot through a maze.

**2.0 Obstacle course**

* A screenshot of the of the maze used for the simulation:

The map is created using the file MapMakerGUI.m provided, a wall is placed at both ends of the maze so that the robot stays within the maze.



**3.0 MATLAB script**

* The m-file script used for the MATLAB simulation with comments:

function driveForwardUntilWall2(serPort)

% Robot drives forward until it senses a wall using front sonar

SetDriveWheelsCreate(serPort, 0.5, 0.5);

while(1) % robot is moving during this loop

distance = ReadSonarMultiple(serPort, 2);

fprintf('sonar distance = %.1f\n', distance) % for debugging

[x y th] = OverheadLocalizationCreate(serPort);

plot(x, y, 'o'); pause(0.1)

if distance < 1.0 % check to see if sonar d < 1m

break

end % end if statement

end % end while loopSetDriveWheelsCreate(serPort, 0, 0) % turn off motors

turnAngle(serPort, .3, -90)

SetDriveWheelsCreate(serPort, 0.5, 0.5);

while(1) % robot is moving during this loop

distance = ReadSonarMultiple(serPort, 2);

fprintf('sonar distance = %.1f\n', distance) % for debugging

[x y th] = OverheadLocalizationCreate(serPort);

plot(x, y, 'o'); pause(0.1)

if distance < 1.0 % check to see if sonar d < 1m

break

end % end if statement

end % end while loopSetDriveWheelsCreate(serPort, 0, 0) % turn off motors

turnAngle(serPort, .3, 90)

SetDriveWheelsCreate(serPort, 0.5, 0.5);

while(1) % robot is moving during this loop

distance = ReadSonarMultiple(serPort, 2);

fprintf('sonar distance = %.1f\n', distance) % for debugging

[x y th] = OverheadLocalizationCreate(serPort);

plot(x, y, 'o'); pause(0.1)

if distance < 1.0 % check to see if sonar d < 1m

break

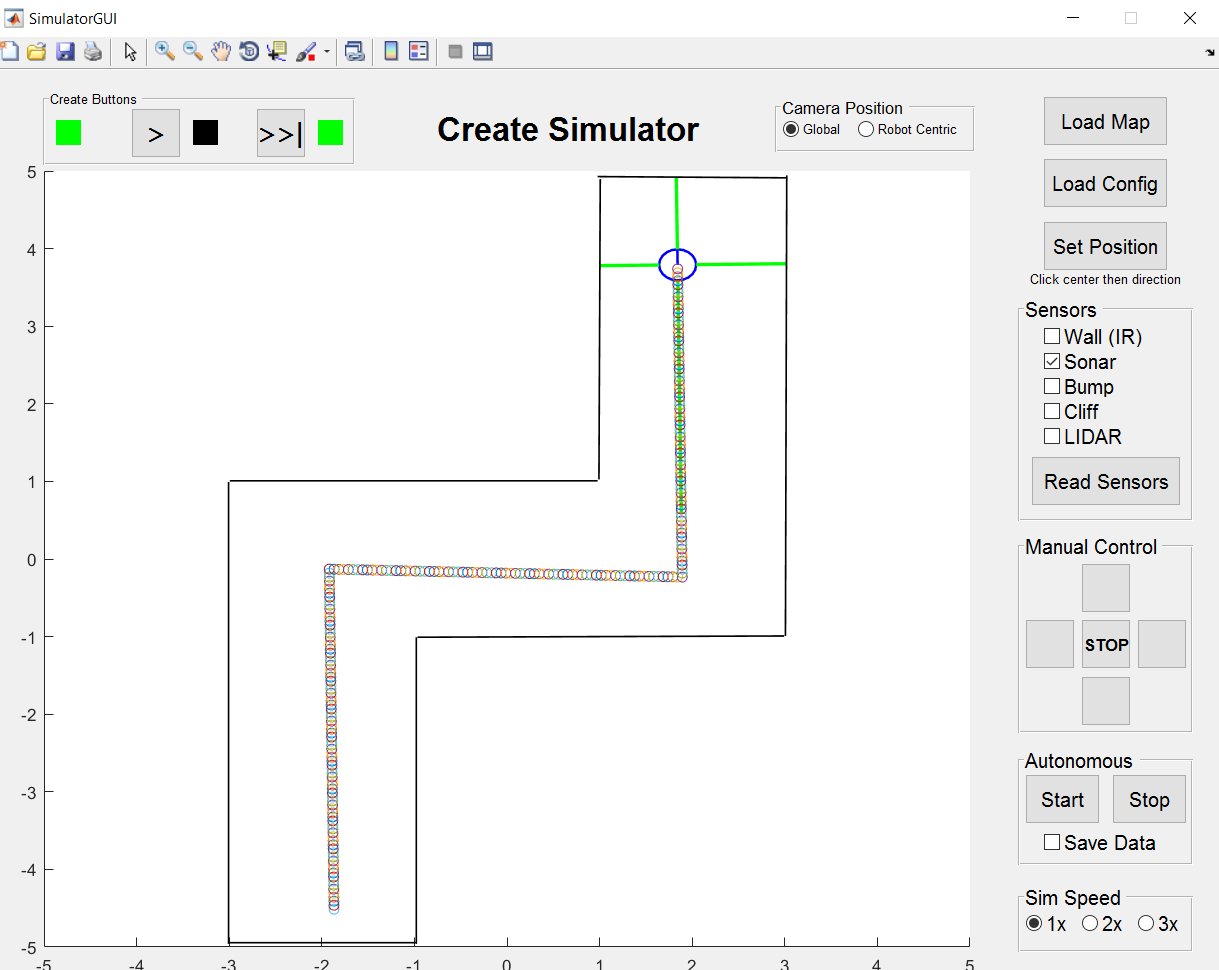
end % end if statement

end %while loopSetDriveWheelsCreate(serPort, 0, 0) % turn off motors

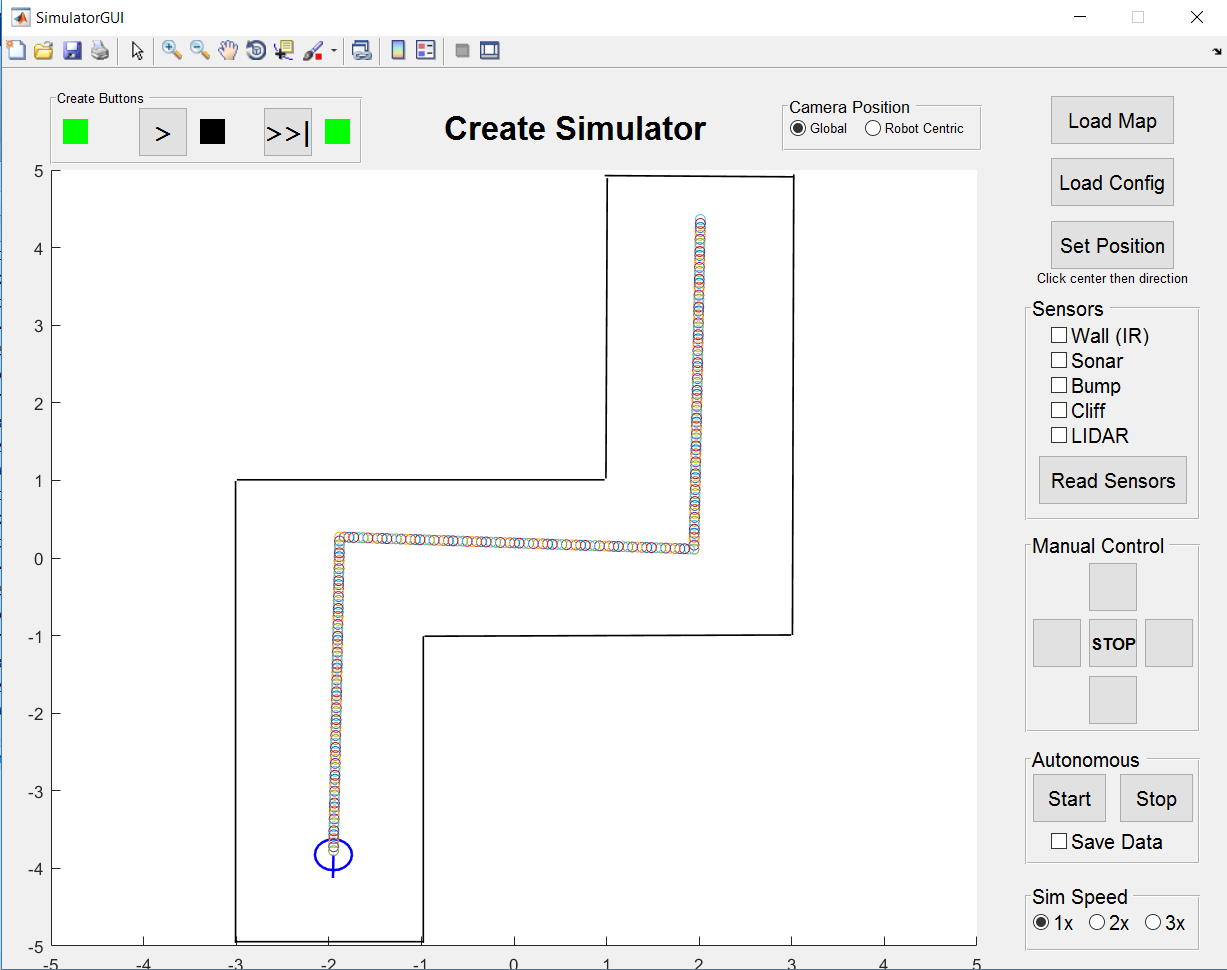
SetDriveWheelsCreate(serPort, 0,0) % turn off motors

**3.0 Screenshots of test trials**

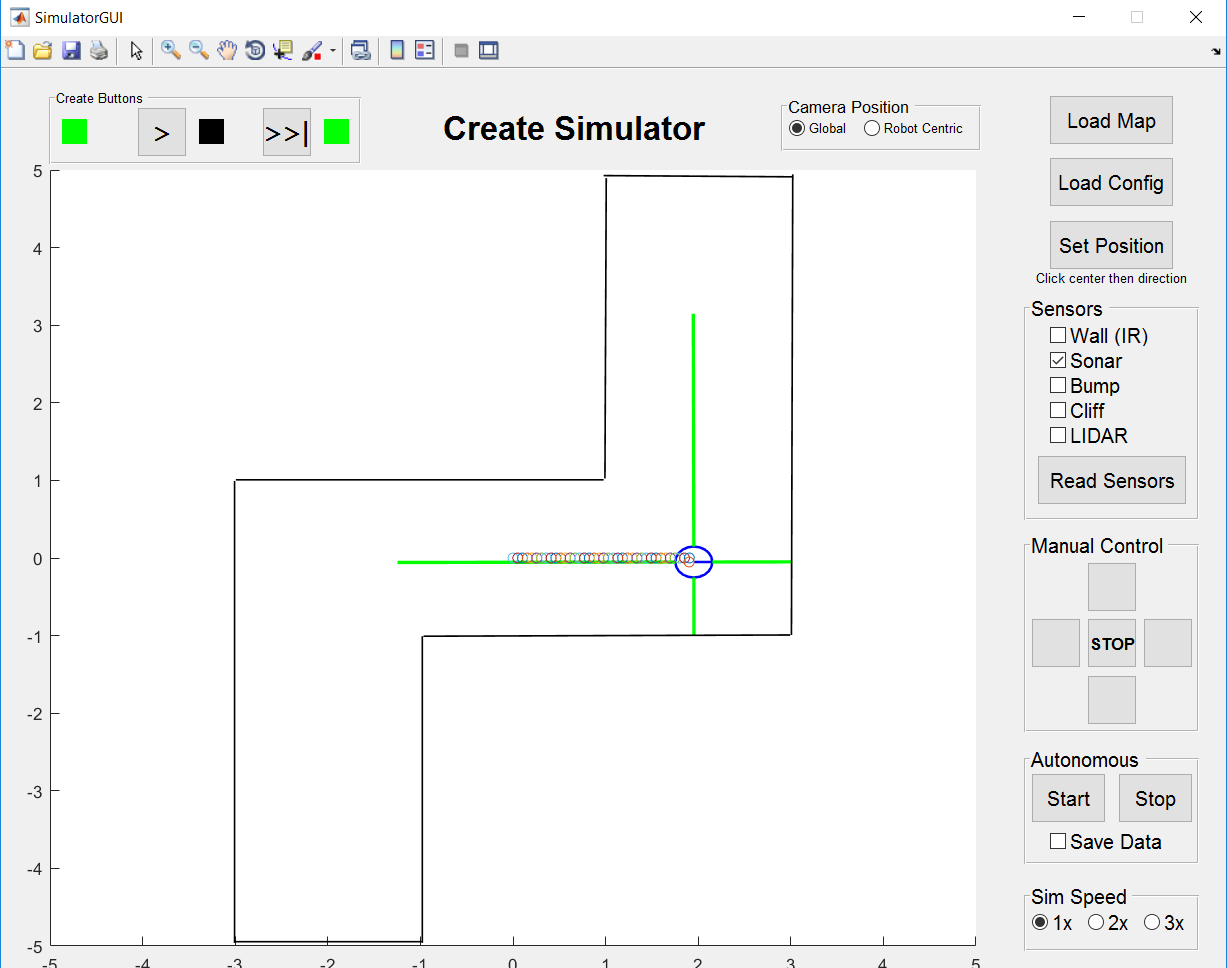
* Test case #1



* Test case #2



* Test case #3



**4.0 Conclusions**

* Conclusions about the performed simulation

In Test trial #3, the robot stopped after reaching less than 1 meter of the wall ahead of it, and then it turned 90 degrees clockwise, and then it turned 90 degrees counter clock wise to find the same wall before the first turn ahead of it, and then it stopped. This was due to not enough coding to account for such scenarios. This can be fixed by making the turn robot choose to turn right or left based on the right and left sensors input, that is, the robot the will turn 90 or -90 based on what sensor (left of right) gives a larger distance.