ENG5009 Advanced Control 5

Laboratory 2 Worksheet

Introduction to Model

1. *Basic Operation*

For the lab assignment you will be using a model of a 4 wheeled robot. This model will be available on the ENG5009 Moodle page under the Lab and Assignment section.

The model is called: `DynamicalModel.m` which is described in *Model_Description_Rev1*. In addition, there is a simple simulation file, `RunModel.m`, that can be used to run the model. This file can be used as the starting point for your assignment.

Further files that may be of interest are:
`DrawRobot.m` – this file draws a representation of the robot on to a given figure
`WallGeneration.m` – supports the development of obstacles
`Sensor.m` – A function that provides a forward-looking sensor that returns the nearest objects on the right- and left-hand side of the robot.

The first stage is to learn how to use the model and the simulation file.

- *Straight line motion*
  Set `voltage_left` and `voltage_right` to 6
  Run the simulation

  Insert a plot of the x distance travelled answer grid

- *Turn counterclockwise*
  Set `voltage_left` to -6 and `voltage_right` to 6
  Run the simulation

  Insert a plot of the heading angle (psi) answer grid

- *Turn clockwise*
  Set `voltage_left` to 6 and `voltage_right` to -6
  Run the simulation

  Insert a plot of the heading angle (psi) angle answer grid

- *Complete a short forward, turn left, forward, turn right manoeuvre*
  To achieve this, you need to create a simple time-based controller that alters the `voltage_left` and `voltage_right` values as and when required. For example, you could use an if statement or a switch statement to alter the variables at particular time steps:

```
if timeStep == 1/stepSize_time
    voltage_left  = -6;
    voltage_right = 6;
end
```

Which would result in `voltage_left` becoming -6 volts and `voltage_right` becoming 6 volts at 1s into the simulation.

Insert a plot of the x/y position in the answer grid

2. *Running the Model with a Fuzzy Controller*

Develop a fuzzy controller that allows the robot to avoid obstacles, based on the tutorial example.

To include the obstacle detection:

```
   sensorOut = Sensor(state(timeStep,19), state(timeStep,20),
state(timeStep,24), obstacleMatrix)
```

And remember to draw all walls:

```
plot(wall_1(:,1), wall_1(:,2),'k-');
plot(wall_2(:,1), wall_2(:,2),'k-');
```

Provide a plot of the path of the system using the tutorial example.

Comment on the behaviour of the system.

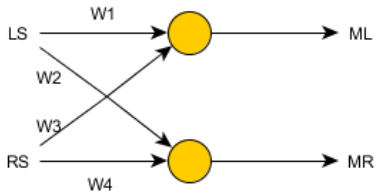
Comment on any changes you would make to improve the performance of the system.

Provide a plot of the path of the system with the changes you have implemented

3. *Running the Model with a Neural Controller*

In the tutorial we discussed a very simple Neural network controller that should be able to drive the robot forward while avoiding obstacles.

The first stage is to create a neural network controller based on the figure below and using the values calculated in the tutorial.



T1	0
T2	0
w1	-1.4
w2	1.2
w3	1.25
w4	-1

Figure: Simple Neural Network to create

Once this is done you are able to apply it the model.

A hint for this would be:

```
%% Neural network
[voltage_left, voltage_right] = NeuralController(sensorOut(1),
sensorOut(2), NeuralParameters);
voltages(timestep,:) = [voltage_left; voltage_left;
voltage_right; voltage_right]
```

Provide a plot of the path of the system using the tutorial example.

Comment on the behaviour of the system.

Comment on any changes you would make to improve the performance of the system.

Provide a plot of the path of the system with the changes you have implemented