SYSC 5104: assignment 1 Final report

Vacuum Cleaning robot simulator

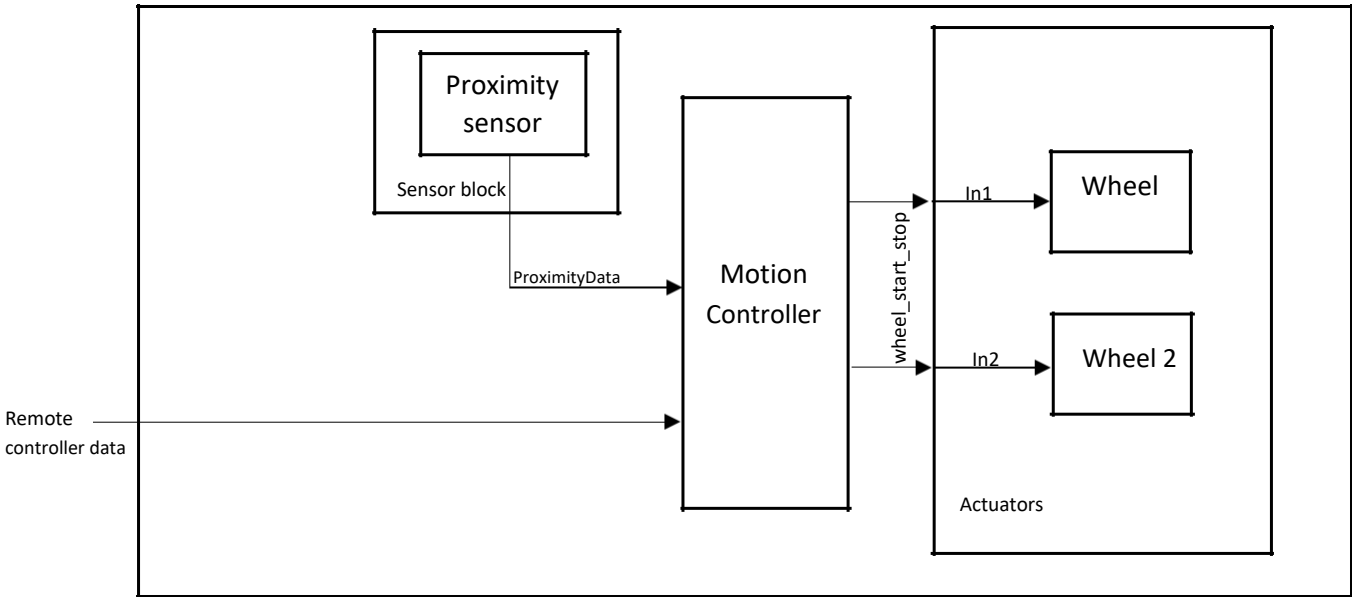
Ritvik Joshi (101271693)
Winter 2023
Systems and computer engineering

Conceptual model of Vacuum cleaning robot

PART I

Vacuum cleaning robot is an autonomous robot which performs the cleaning of a room/area which is already mapped. For the purpose of assignment, I am considering one sensor one input from remote controller and two outputs for motor. The start stop commands are sent by the remote controller for controlling the robot. The proximity sensor is used for the safety purpose. And the two wheels used for motion will receive the data from controller. Actuation block is a coupled model consisting of two atomic models of wheels which are identical.

Sketch of the model structure



Description of the behavior of the component

1. **Sensor Block –**

   This is responsible for data acquisition of the robot.

   a. **Proximity Sensor:** this sensor detects if some obstacle suddenly appears in front of the robot. The sensor data is 0 or 1. If no obstacle is present then the data will be 0 else the data will be 1. This will help the robot avoiding any ascendant.

2. **Actuators–**

   This block contains all the actuator atomic models. The main 2 actions performed by the robot are actuation of suction pump and actuation of base wheels/motors.

   a. **Left Wheel and Write Wheel:** These are exactly same models which receive the values from controller and do not produce any output for this scope.

3. **Motion Controller –**

This is the atomic model responsible for deciding the control action for robot. When it receives start command from the remote controller, it will send forward command to the motor. When it receives stop it will stop the motors. In case if the remote controller starts the action but proximity sensor detects any object, it will stop the action until the object is moved. Once the object is moved, it will restart the forward motion.

PART II

Model behaviour -

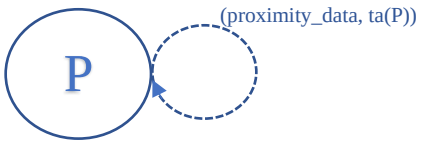
As shown in the figure, the model has one input and currently 0 outputs. The remote-control input indicates the external signal to start or stop the wheels. The Vacuum cleaning robot Simulator has 3 main components: sensor block, Actuators, and motion controller. The sensor data acquisition is done from the sensor block. The sensor sends the data values to the motion controller block. The motion controller block has the control logic for decision making. It takes the sensor data and external input from the remote controller to make its decision whether to start the motors or stop the motor. If the remote-control input is active and the distance from obstacle is safe, then the robot start command is sent to the actuators coupled model. The sensor block has atomic model proximity sensor within. The actuators block contains 2-wheel models which are 2 different instances of wheel atomic model.

Formal Specification

The formal specifications <S, X, Y, δint, δext, λ, ta > for the atomic models are defined as follows:

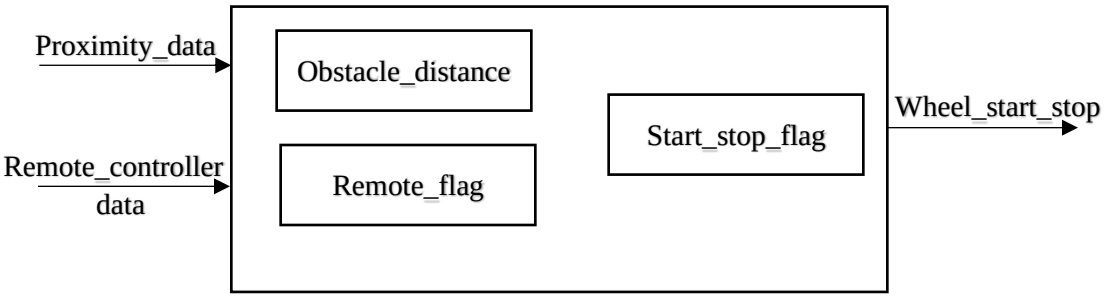
Proximity_sensor:

S = {P}
X = {}
Y = {proximity_data}
δint() : δint(P) = P
δext = ()
λ(P) = proximity_data
ta = sampling_time



Motion_controller:

S = {obstacle_distance, remote_controller_in, start_stop_flag}
X = {proximity_data, remote_controller_data}
Y = {wheel_start_stop}



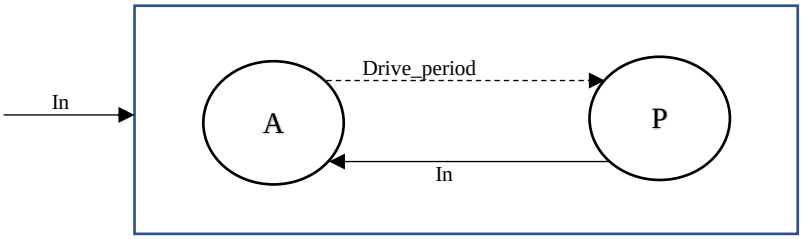
δint(!remote_flag)
	set sigma = infinite

δext(obstacle_distance, remote_flag)
	if(remote_flag)
	{
	If(obstacle_distance > 10)
		Start_stop_flag =1;
	Else
		Start_stop_flag =0;
	}
	Else	//remote)flag == 0
	{
		Start_stop_flag = 0;
	}
λ =
	send wheel_start_stop = Start_stop_flag;

ta(remote_flag) = command_time;
ta(!remote_flag) = infinite;

Wheel/Wheel2:

S = {active, passive}
X = {In1}
Y = {}
δint: δint(active) = passive
δext: δext (passive, In1) = active
λ() = null
ta(active) = drive_period
ta(passive) = infinite



Sensor_block:

X = {};
Y = {proximity_data};
D = {proximity_sensor};
I() = ();
Z(proximity_sensor) = self;

Actuators:

X = {wheel_start_stop};
Y = {out1, out2};
D = {wheel, wheel2};
I(wheel) = self;
I(wheel2) = self;
Z = ();

Vacuume_robot:

X = {Remote_controller_in};
Y = {}
D = {saensor_block, Motion_controller, Actuators};
I(motion_controller) = {Sensor_block, self};
I(Actuators) = motion_controller ;
Z(sensor_block) = motion_controller;
Z(motion_controller) = Actuators;
SELECT: ({motion_controller, Actuators, Sensor_block }) = motion_controller;

Test strategy:

The models are tested with black box testing method as recommended by the reference report. The test cases created in the files try to cover maximum possible scenarios.

PART III

Test Cases and Execution results:

Atomic model proximity_sensor:

The proximity sensor model creates random number between 0 to 100 as sensor output. The test does not require any form of input. The test asks the model to generate 10 outputs with a standards frequency rate. The results are as expected.

Output –

time;model_id;model_name;port_name;data

0;1;proximity_sensor;;sampling rate0sensor value0
0;1;proximity_sensor;Proximity_data;0
0;1;proximity_sensor;;sampling rate10sensor value34
10;1;proximity_sensor;Proximity_data;34
10;1;proximity_sensor;;sampling rate10sensor value44
20;1;proximity_sensor;Proximity_data;44
20;1;proximity_sensor;;sampling rate10sensor value63
30;1;proximity_sensor;Proximity_data;63
30;1;proximity_sensor;;sampling rate10sensor value30
40;1;proximity_sensor;Proximity_data;30
40;1;proximity_sensor;;sampling rate10sensor value1
50;1;proximity_sensor;Proximity_data;1
50;1;proximity_sensor;;sampling rate10sensor value9
60;1;proximity_sensor;Proximity_data;9
60;1;proximity_sensor;;sampling rate10sensor value53
70;1;proximity_sensor;Proximity_data;53
70;1;proximity_sensor;;sampling rate10sensor value57
80;1;proximity_sensor;Proximity_data;57
80;1;proximity_sensor;;sampling rate10sensor value57
90;1;proximity_sensor;Proximity_data;57

90;1;proximity_sensor;;sampling rate10sensor value20
90;1;proximity_sensor;;sampling rate10sensor value20

Atomic model motion_controller:
Motion controller takes the remote controller input and sensor input from proximity sensor. The two input files are provided to the test input file. The test cases checks the cases for proximity data less than 10 for remote input 1 and 0. The output generated is sent to wheels atomic models. The output was stored in csv file.
Inputs (motion_controller_input_test_proximity.txt) -
10 24
20 14
30 43
40 76
50 9
60 76
70 54
80 7
90 32
100 90

Input (motion_controller_input_test_Remote_Control.txt) –
10 1
20 1
30 1
40 1
50 1
60 1
70 0
80 0
90 0
100 0
Output –
time;model_id;model_name;port_name;data
0;1;motion_controller;;{0 0 0}
0;2;iestream2;;10
0;3;iestream;;10
10;1;motion_controller;wheel_start_stop;0
10;1;motion_controller;;{1 24 1}
10;2;iestream2;out;1
10;2;iestream2;;10
10;3;iestream;out;24
10;3;iestream;;10
10;1;motion_controller;wheel_start_stop;1
10;1;motion_controller;;{1 24 1}
20;1;motion_controller;;{1 14 1}
20;2;iestream2;out;1
20;2;iestream2;;10
20;3;iestream;out;14
20;3;iestream;;10
20;1;motion_controller;wheel_start_stop;1
20;1;motion_controller;;{1 14 1}
30;1;motion_controller;;{1 43 1}
30;2;iestream2;out;1
30;2;iestream2;;10

30;3;iestream;out;43
30;3;iestream;;10
30;1;motion_controller;wheel_start_stop;1
30;1;motion_controller;;{1 43 1}
40;1;motion_controller;;{1 76 1}
40;2;iestream2;out;1
40;2;iestream2;;10
40;3;iestream;out;76
40;3;iestream;;10
40;1;motion_controller;wheel_start_stop;1
40;1;motion_controller;;{1 76 1}
50;1;motion_controller;;{1 9 0}
50;2;iestream2;out;1
50;2;iestream2;;10
50;3;iestream;out;9
50;3;iestream;;10
50;1;motion_controller;wheel_start_stop;0
50;1;motion_controller;;{1 9 0}
60;1;motion_controller;;{1 76 1}
60;2;iestream2;out;1
60;2;iestream2;;10
60;3;iestream;out;76
60;3;iestream;;10
60;1;motion_controller;wheel_start_stop;1
60;1;motion_controller;;{1 76 1}
70;1;motion_controller;;{0 54 0}
70;2;iestream2;out;0
70;2;iestream2;;10
70;3;iestream;out;54
70;3;iestream;;10
70;1;motion_controller;wheel_start_stop;0
70;1;motion_controller;;{0 54 0}
80;1;motion_controller;;{0 7 0}
80;2;iestream2;out;0
80;2;iestream2;;10
80;3;iestream;out;7
80;3;iestream;;10
80;1;motion_controller;wheel_start_stop;0
80;1;motion_controller;;{0 7 0}
90;1;motion_controller;;{0 32 0}
90;2;iestream2;out;0
90;2;iestream2;;10
90;3;iestream;out;32
90;3;iestream;;10
90;1;motion_controller;wheel_start_stop;0
90;1;motion_controller;;{0 32 0}
100;1;motion_controller;;{0 90 0}
100;2;iestream2;out;0
100;2;iestream2;;inf
100;3;iestream;out;90
100;3;iestream;;inf
100;1;motion_controller;wheel_start_stop;0
100;1;motion_controller;;{0 90 0}
100;1;motion_controller;;{0 90 0}

100;2;iestream2;;inf
100;3;iestream;;inf

Atomic model wheel:

Wheel atomic model receives the data from text file through the input port. The test just checks whether the input is received correctly. The test input and output file are as follows.

Input (wheel_input.txt) –

10 1
20 1
30 0
40 0
50 1
60 1
70 0
80 0
90 1
100 0

Output –

time;model_id;model_name;port_name;data
0;1;wheel;;64
0;2;iestream;;10
10;1;wheel;;1
10;2;iestream;out;1
10;2;iestream;;10
20;1;wheel;;1
20;2;iestream;out;1
20;2;iestream;;10
30;1;wheel;;0
30;2;iestream;out;0
30;2;iestream;;10
40;1;wheel;;0
40;2;iestream;out;0
40;2;iestream;;10
50;1;wheel;;1
50;2;iestream;out;1
50;2;iestream;;10
60;1;wheel;;1
60;2;iestream;out;1
60;2;iestream;;10
70;1;wheel;;0
70;2;iestream;out;0
70;2;iestream;;10
80;1;wheel;;0
80;2;iestream;out;0
80;2;iestream;;10
90;1;wheel;;1
90;2;iestream;out;1
90;2;iestream;;10
100;1;wheel;;0
100;2;iestream;out;0
100;2;iestream;;inf
100;1;wheel;;0
100;2;iestream;;inf

Coupled model vacuume_robot (main):
The complete model is simulated in this testing. The sample tests from time 0 to 20 and 100 to 120 is pasted below. These includes all the cases for remote controller input and sensor data input combinations. For this simulation test we have one extra atomic model which sends the remote input data in toggling fashion.

time;model_id;model_name;port_name;data
0;1;remote_control_In;;0
0;2;motion_controller;;{0 0 0}
0;4;proximity_sensor;;sampling rate0sensor value0
0;6;wheel2;;176
0;7;wheel;;176
0;1;remote_control_In;Ext_Remote_control;0
0;1;remote_control_In;;1
0;2;motion_controller;;{0 0 0}
0;4;proximity_sensor;Proximity_data;0
0;4;proximity_sensor;;sampling rate4sensor value34
0;2;motion_controller;wheel_start_stop;0
0;2;motion_controller;;{0 0 0}
0;6;wheel2;;0
0;7;wheel;;0
4;2;motion_controller;;{0 34 0}
4;4;proximity_sensor;Proximity_data;34
4;4;proximity_sensor;;sampling rate4sensor value44
4;2;motion_controller;wheel_start_stop;0
4;2;motion_controller;;{0 34 0}
4;6;wheel2;;0
4;7;wheel;;0
8;2;motion_controller;;{0 44 0}
8;4;proximity_sensor;Proximity_data;44
8;4;proximity_sensor;;sampling rate4sensor value63
8;2;motion_controller;wheel_start_stop;0
8;2;motion_controller;;{0 44 0}
8;6;wheel2;;0
8;7;wheel;;0
12;2;motion_controller;;{0 63 0}
12;4;proximity_sensor;Proximity_data;63
12;4;proximity_sensor;;sampling rate4sensor value30
12;2;motion_controller;wheel_start_stop;0
12;2;motion_controller;;{0 63 0}
12;6;wheel2;;0
12;7;wheel;;0
16;2;motion_controller;;{0 30 0}
16;4;proximity_sensor;Proximity_data;30
16;4;proximity_sensor;;sampling rate4sensor value1
16;2;motion_controller;wheel_start_stop;0
16;2;motion_controller;;{0 30 0}
16;6;wheel2;;0
16;7;wheel;;0
20;2;motion_controller;;{0 1 0}
20;4;proximity_sensor;Proximity_data;1
20;4;proximity_sensor;;sampling rate4sensor value9
20;2;motion_controller;wheel_start_stop;0
20;2;motion_controller;;{0 1 0}

20;6;wheel2;;0
20;7;wheel;;0

100;1;remote_control_In;Ext_Remote_control;1
100;1;remote_control_In;;0
100;2;motion_controller;;{1 81 1}
100;4;proximity_sensor;Proximity_data;81
100;4;proximity_sensor;;sampling rate4sensor value24
100;2;motion_controller;wheel_start_stop;1
100;2;motion_controller;;{1 81 1}
100;6;wheel2;;1
100;7;wheel;;1
102;6;wheel2;;1
102;7;wheel;;1
104;2;motion_controller;;{1 24 1}
104;4;proximity_sensor;Proximity_data;24
104;4;proximity_sensor;;sampling rate4sensor value50
104;2;motion_controller;wheel_start_stop;1
104;2;motion_controller;;{1 24 1}
104;6;wheel2;;1
104;7;wheel;;1
106;6;wheel2;;1
106;7;wheel;;1
108;2;motion_controller;;{1 50 1}
108;4;proximity_sensor;Proximity_data;50
108;4;proximity_sensor;;sampling rate4sensor value93
108;2;motion_controller;wheel_start_stop;1
108;2;motion_controller;;{1 50 1}
108;6;wheel2;;1
108;7;wheel;;1
110;6;wheel2;;1
110;7;wheel;;1
112;2;motion_controller;;{1 93 1}
112;4;proximity_sensor;Proximity_data;93
112;4;proximity_sensor;;sampling rate4sensor value65
112;2;motion_controller;wheel_start_stop;1
112;2;motion_controller;;{1 93 1}
112;6;wheel2;;1
112;7;wheel;;1
114;6;wheel2;;1
114;7;wheel;;1
116;2;motion_controller;;{1 65 1}
116;4;proximity_sensor;Proximity_data;65
116;4;proximity_sensor;;sampling rate4sensor value70
116;2;motion_controller;wheel_start_stop;1
116;2;motion_controller;;{1 65 1}
116;6;wheel2;;1
116;7;wheel;;1
118;6;wheel2;;1
118;7;wheel;;1

120;2;motion_controller;;{1 70 1}
120;4;proximity_sensor;Proximity_data;70
120;4;proximity_sensor;;sampling rate4sensor value52
120;2;motion_controller;wheel_start_stop;1
120;2;motion_controller;;{1 70 1}
120;6;wheel2;;1
120;7;wheel;;1