**Pre-Lab 2 Report                                        Alan Palayil**

ECE 100-00X                                                                 Teammates:  Mathew, Rezwan

Prof. Oruklu                                                                                           Lab Date: 9/06/19

TA: Your TA                                                                                         Due Date: 9/06/19

**Problem Statement**

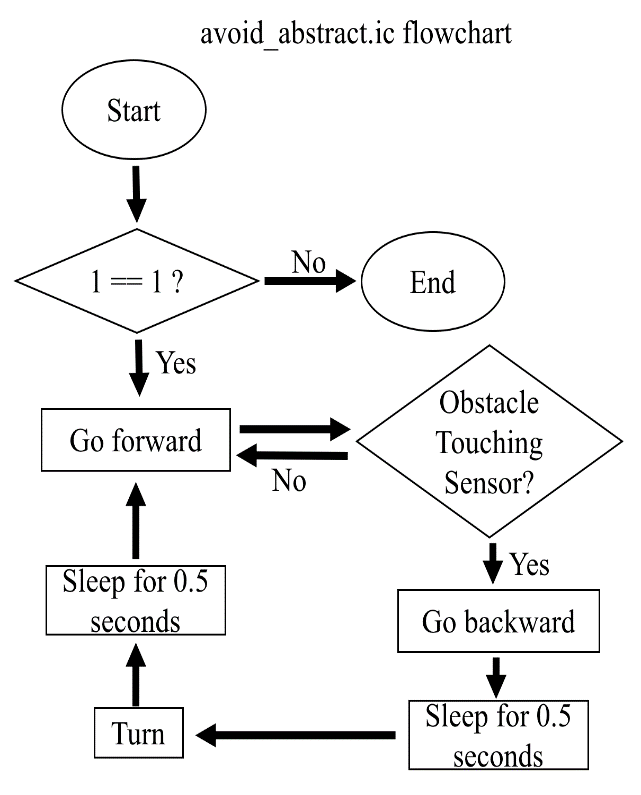
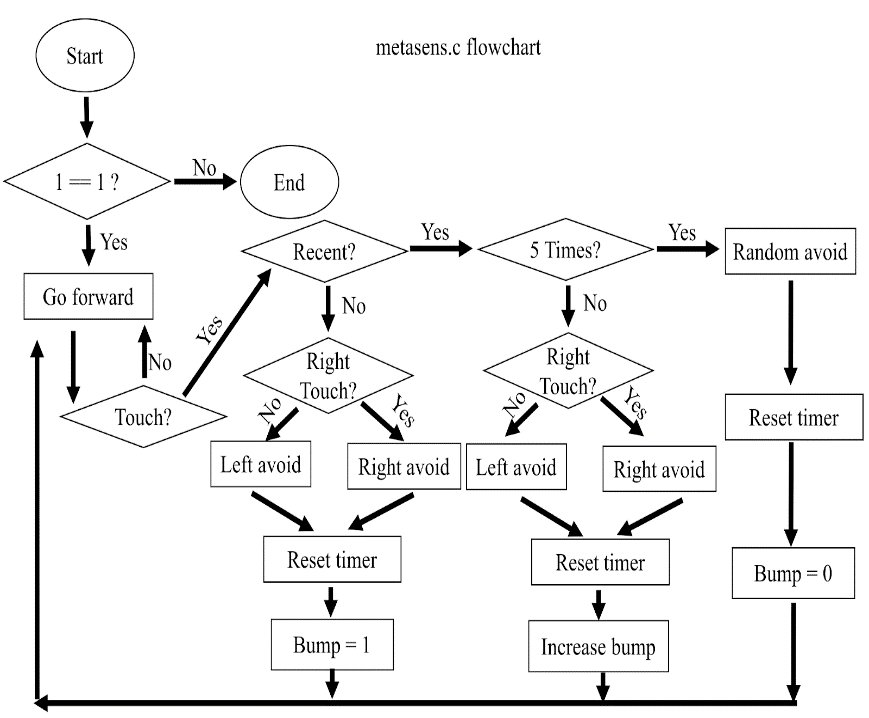
The purpose of this lab is to use the Interactive C, Handy Board, and Handybug, to learn and understand the different advances the Handybug can take based-off of robot behavior in meta-sensing in solving complex tasks.

**Investigation/Research**

There are a few different ways of dealing with an unpredictable maze, which include randomness and meta-sensing solutions. At the same time, it is not always the most effective function for a method of solving the maze. Type coercion is a standard way for conversion across various numeric representations. This is one of the techniques used in randomness. The action can be triggered through a certain number that can be generated from type coercion, taking in a random input. Beeper test which yields different frequencies of sound (pitch of noise) can be used to test random function. Consider a bump-and-turn loop function. Executing a complex task through small functions or steps is called the emergent phenomena. During a situation in which the solution becomes more difficult to predict, using the emergent phenomena can help account for different situations that are pre-program to every possible outcome for every possible situation. Another approach to solving a complex problem would be through meta-sensing.  Meta-sensing can have set rules that help restrict the options for a function to execute. When the program considers a various set of circumstances, presets it into the software, and chooses an outcome based on the set of conditionals it must answer. An example of this would be having a sensor-action rule that would not only sense if the robot hit the wall, but if it successfully hit it within a certain timeframe. One way to combine the meta-sensing data with a random function is by slowly limiting the range of numbers for a random function and making it an option once all other conditional approaches are exhausted. This will widen the Handy Bug’s scope in solving the maze. Having a counter which can be used to reset the variable function can track how long the Handybug is taking to execute a solution for its situation.

**Alternative Solutions**

There were two solutions provided via the textbook which include avoid\_abstract.ic and metasens.c. The first alternate solution is avoid\_abstract.ic. This program is a simple function that will execute left turns whenever it is stuck in a corner or senses any obstacle from its touch sensor.  For the Handybug to successfully move in a maze, there would need to be no right turns. After a while, if the robot gets stuck in a dead end and only turns left, it would not complete the maze with other random situations such as dead ends or right turns. The other alternative solution is metasens.c. From the textbook, it considers both individual right and left sensing inputs from the two sensors as well as having a bump counting variable. In order to move through the maze, it uses a random function which states after a turn more than 4 times, the Handybug would have moved in a circular path and moved back to the beginning.  After the count variable reaches five times, it will execute a random movement that will hopefully move it out of the trapped situation.

**Optimum Solution**

The optimum solution would be a variation of metasens.c including a variation for the random avoid category to go in possibly the opposite direction it originally had been turning because it may have possibly gone in circles.  By changing the random function and resetting the timer, the robot does move in a different direction, but it may be stuck in a dead end. In that case, it needs to make a 180 degree turn to get out if it is stuck in a narrow path. The metasens.c considers different sensor inputs from the right and left sensors while counting the amount of times it repeats a task.  In this sense, there is a chance that it will do a random function to possibly get out of a tricky situation.

**References**

1.  Martin, Fred G. 2001. *Robotic Explorations: A Hands-On Introduction to Engineering*. New Jersey: Prentice Hall.

2.  Oruklu, Erdal. 2015. *ECE 100* *Lecture Notes*. Chicago: Illinois Institute of Technology, Electrical and Computer Engineering Department.

**Appendix**

turtle.ic

int LEFT\_MOTOR = 3;

int RIGHT\_MOTOR = 0;

void forward ()

{

fd (LEFT\_MOTOR);

fd (RIGHT\_MOTOR);

}

void backward ()

{

bk (LEFT\_MOTOR);

bk (RIGHT\_MOTOR);

}

void right ()

{

fd (LEFT\_MOTOR);

bk (RIGHT\_MOTOR);

}

    void left ()

{

fd (RIGHT\_MOTOR);

bk (LEFT\_MOTOR);

}

void stop ()

{

off (LEFT\_MOTOR);

off (RIGHT\_MOTOR);

}

void main ()

{

forward ();

if (digital (10))

{

backward ();

sleep (0.5);

left ();

}

}

avoid\_abstract.ic

void main () {

while (1) {

forward ();

if (digital (10))

{

backward ();

sleep (0.5);

left ();

}

}

}

timer.c

float\_timer;

void reset\_timer () {

\_timer = seconds ();

}

float timer () {

return seconds () - \_timer;

return seconds () - \_timer;

}

metasens

int LEFT\_TOUCH= 10;

int RIGHT\_TOUCH= 11;

void main () {

int recent\_bumps= 0;

reset\_timer ();

while (1) {

forward ();

if (digital (LEFT\_TOUCH) {

if (timer () < 2.) {

if (recent\_bumps == 5) {

random\_avoid ();

reset\_timer ();

recent\_bumps= 0;

} else {

left\_avoid ();

reset\_timer ();

recent\_bumps++;

}

} else {

left\_avoid ();

reset\_timer ();

recent\_bumps= 1;

}

}

if (digital (RIGHT\_TOUCH) {

if (timer () < 2.) {

if (recent\_bumps == 5) {

random\_avoid ();

reset\_timer ();

  recent\_bumps= 0;

} else {

  right\_avoid ();

  reset\_timer ();

  recent\_bumps++;

  }

} else {

  right\_avoid ();

  reset\_timer ();

  recent\_bumps= 1;

}

}

}

}