T.C
YEDİTEPE UNIVERSITY

Mechanical Engineering Department

ME-456

Mechatronics

Group – 10

Project Report

Firefighter Robot

Instructor:

Koray Kadir ŞAFAK

Group Members:

Arda MANTAŞ	-	20130701044
Ozan Ongun SAYIN	-	20160705027
Turan Oğuzhan KOYUNOĞLU	-	20140701027

1 Table of Contents

1 Introduction

Mechatronics consists of a mixture of mechanics and electronics words, but it includes parts from many different areas. These can be line up as robotics, circuits, control, programming, etc.

It is a common working area of robotics, mechanics, control systems, computer, electronics and space sciences. Nowadays, many robots are working beyond their mechanic forms to automatize processes. It is said that production rates will increase to 40%-50% due to the development of mechatronics till 2025 and this will generate a contribution to world's economy between 1.7-2.2 trillion dollars. Also, this rapid development of robotics brings out some questions like what if factories do not need labor force since the robots takes place of humans. This may cause an increase in the number of unemployed people. For example: in San Francisco, California University has developed chemist robots, that distribute medicine automatically in 2011. The system has distributed over 350.000 medicine to the patients without any mistake. Another example is that Amazon has more than 15.000 robots to transmit orders faster. These robots provide an advantage to the company by decreasing the process time from 90 minutes to 15 minutes. They also can carry the load up to 750 kilograms, and lift these heavy packages to the upper shelves. Company can store 50% more products at the same area by the usage of robots, and they also decrease the frazzling rate of workers.

Mbot can be programmed by using Mblock and Arduino software. Mblock is available for Makeblock products, while Arduino is an open-source programming software. Arduino is used for this project.

Since the core of the Mbot is based on Arduino, Arduino's usage for prototyping is suitable. Sensors, LEDs, switch on/off button, battery connector, battery bed connector, USB connector, motor connector and connection points (RJ25) are available on the core.

Bluetooth and 2.4G module can be used for wireless communication, but wired connection is to be used for the communication between computer and motherboard.

2 Abstract

The aim of this project is to create a robot that detects the point/points of fire, and extinguish it using the loaded matter (water) in the tank. Flame sensor is used to detect whether the flame exists or not. A trigger mechanism is used to fight with fire, when the location of the flame detected. Flame sensor that is produced by Makeblock senses the flame with an angle of 60° within distance of 1 meter. The algorithm is constituted such that the robot does not pass any flame source while searching for flame in a specific area.

Ultrasonic sensor is used to prevent robot from hitting any obstacles around and to keep the robot within the specified area. Algorithm is developed so that the robot does not get damaged and keep searching for the flame and fight with it when it finds the flame.

In order to run the water pump, LN298 motor drive module is used. Another voltage source is used to supply power the to motor driver and essential connections between battery bed, motor drive module and motherboard are done.

3 Design of Robot

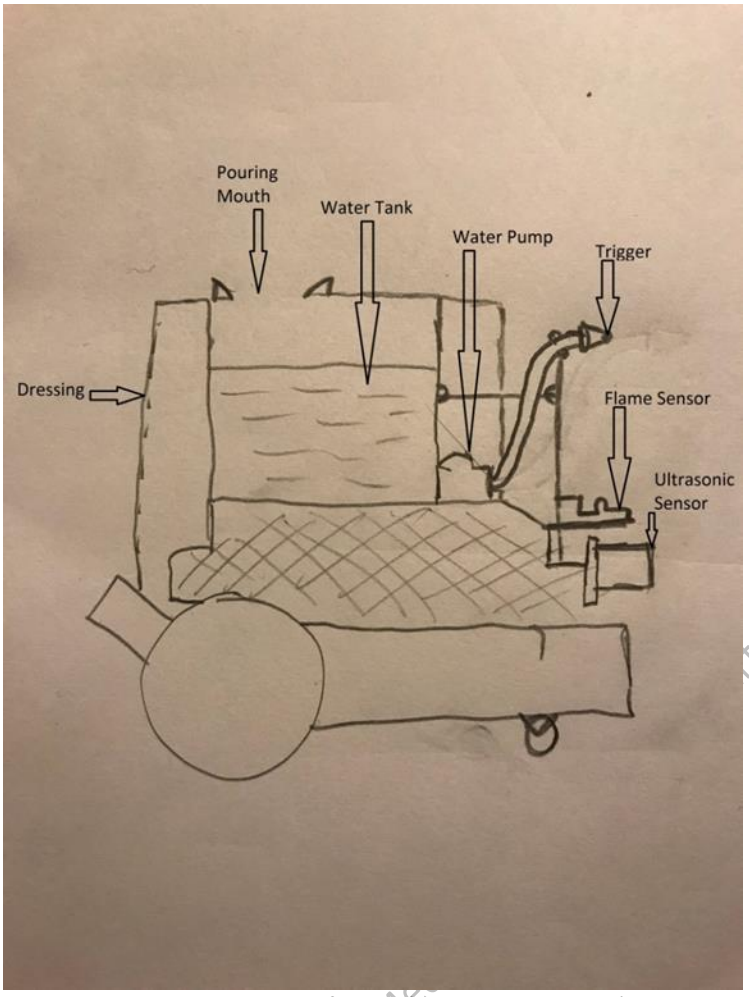

Figure 1- Preliminary design of robot (hand drawn sketch)

Preliminary sketch of the robot is given on the Figure-1 above. The pump was considered to place outside the tank, and connected to the tank on the bottom. Extra battery bed and motor drive module was not considered, since the design has overhauled due to the requirements and constraints faced during the project and development phases. Also, the locations of flame sensor and ultrasonic sensor has changed due to similar problems faced.

3.1 Mechanical Design

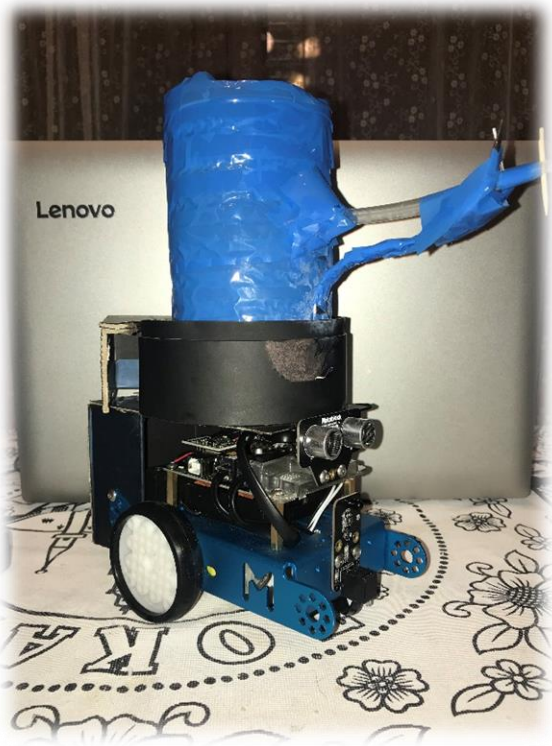

Figure 2 – Side view of robot

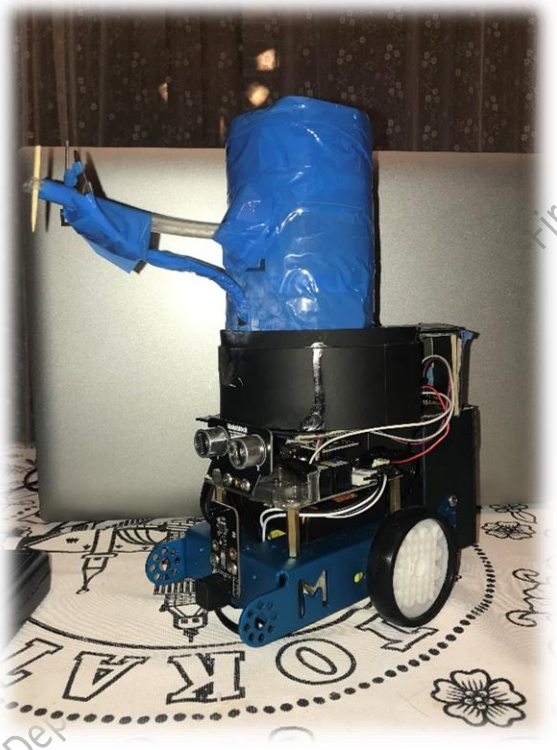

Figure 3 – Side view of robot

Figure 4 – Top view of robot including motor drive module and pump locations

3.2 Electrical Design

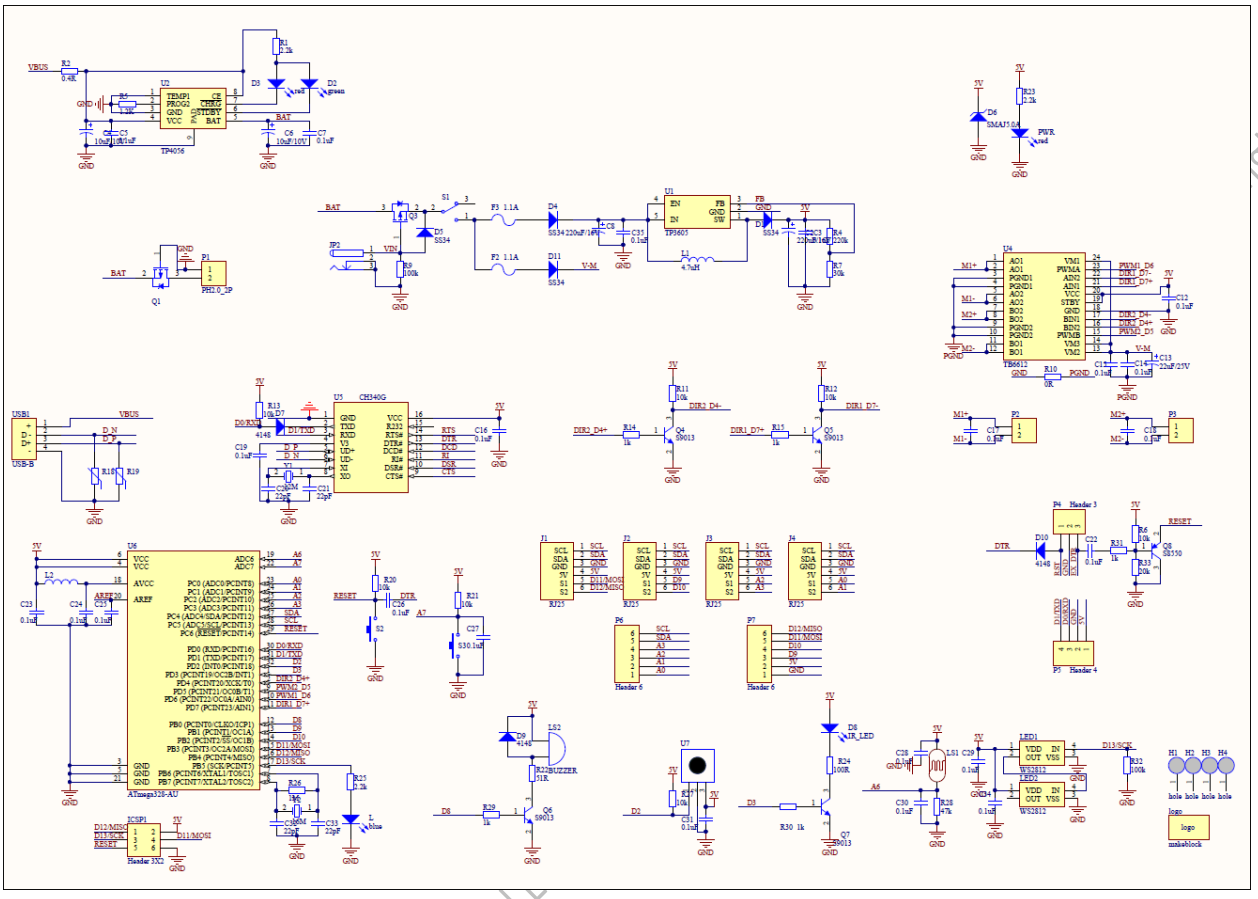

Figure 5 – Schematic of MCore

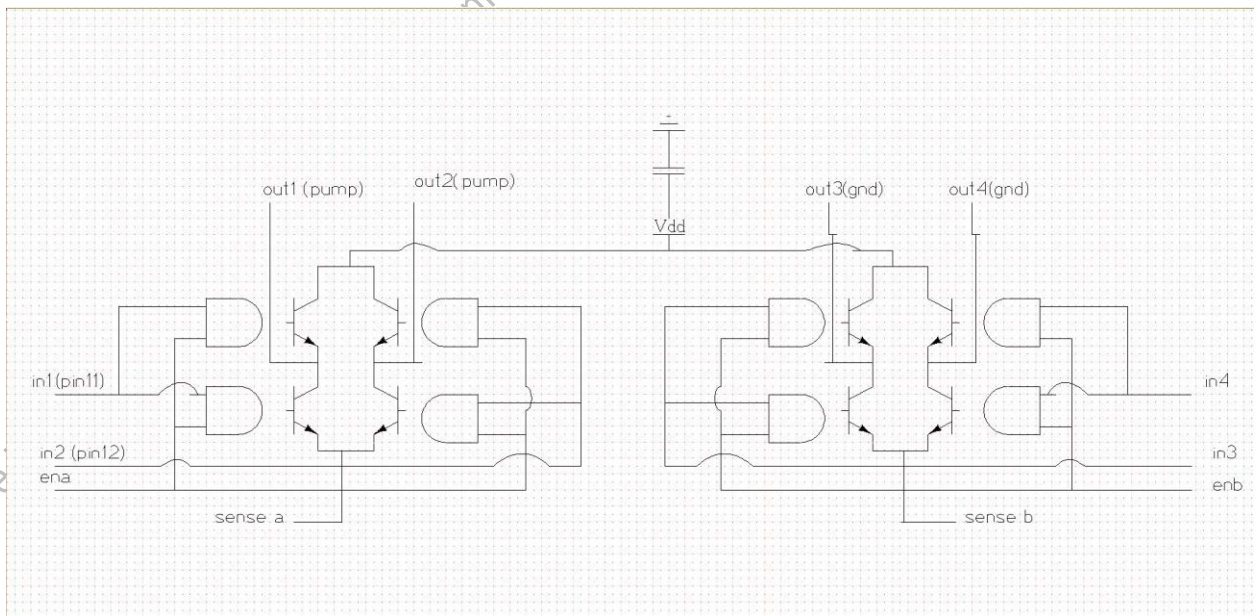

Figure 6 – Schematic of the connection of motor drive module to the supply voltage and motherboard

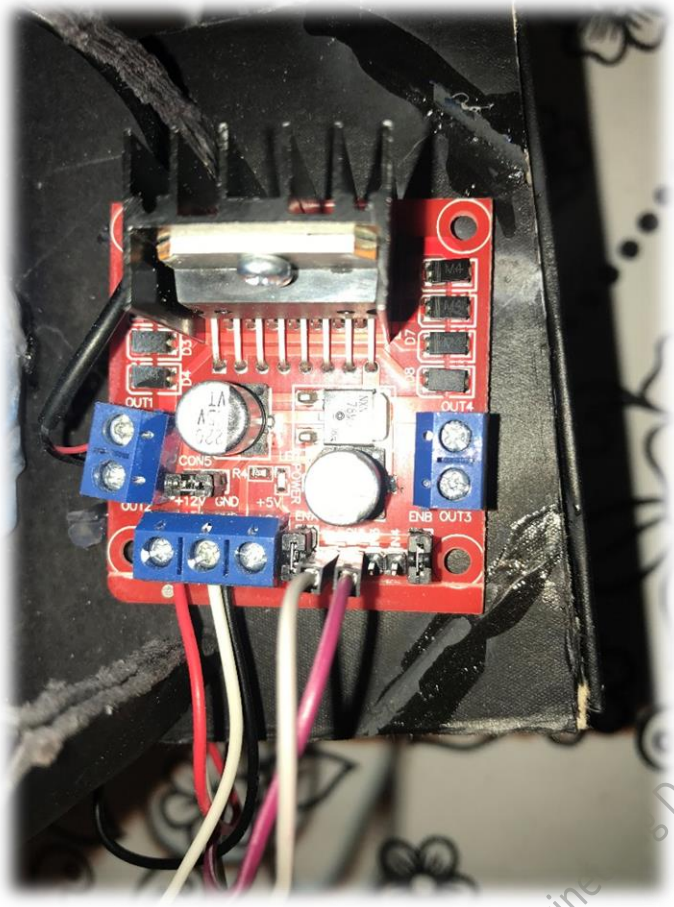

Figure 7 – Connections on the motor drive module (Between extra battery bed, motor drive module and motherboard)

3.3 Control System Design

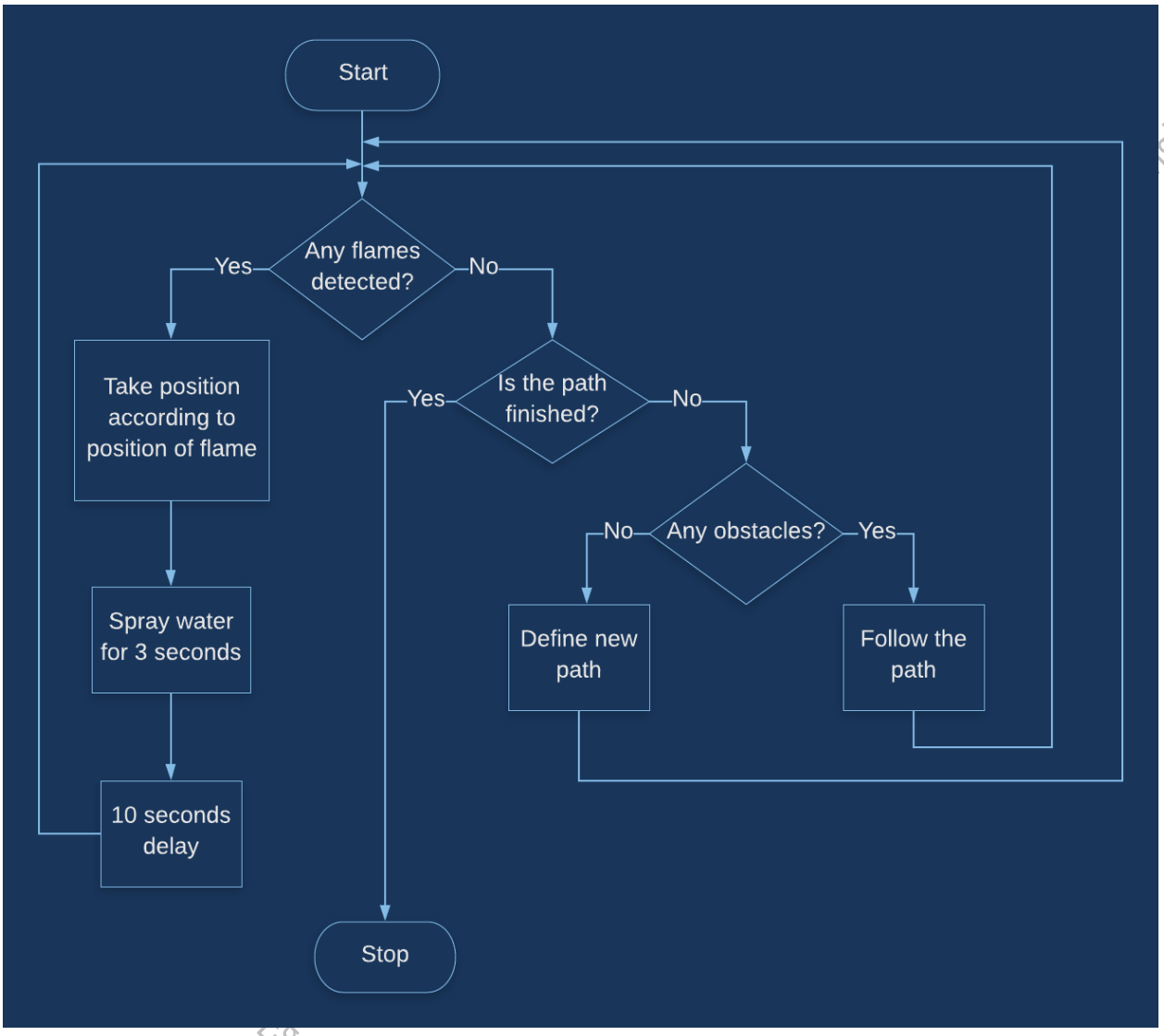

Figure 4 – Flowchart of motion of robot

#include <Arduino.h>	//Introducing Arduino
#include <Wire.h>	//Introducing Wired Connection
#include <SoftwareSerial.h>	//Introducing Software
#include <MeMCore.h>	//Introducing Motherboard
MeDCMotor motor1(M1);	//Introducing motor1(left motor) connected to M1
MeDCMotor motor2(M2);	//Introducing motor2(right motor) connected to M2
MeUltrasonicSensor ultraSensor (PORT_3);	//Introducing ultrasonic sensor connected to PORT 3
MeFlameSensor FlameSensor1 (PORT_4);	//Introducing flame sensor connected to PORT 4
void move(int direction, int speed)	
{	
int leftSpeed;	//direction and speed integers are defined in the function of move
int rightSpeed;	
leftSpeed = speed;	
rightSpeed = speed;	
if (direction == 0) {	//Values of left speed and right speed are identified as 0 initially
leftSpeed = speed;	

```
    rightSpeed = speed;
  }
  else if (direction == 1) {
    leftSpeed = speed;
    rightSpeed = -speed;
  } else if (direction == 2) {
    leftSpeed = -speed;
    rightSpeed = speed;
  } else if (direction == 3) {
    leftSpeed = speed;
    rightSpeed = speed;
  } else if (direction == 4) {
    leftSpeed = -speed;
    rightSpeed = -speed;
  }
  //0, stop, 1 forward, 2 backward, 3 left, 4 right.
  motor1.run((M1) == M1 ? - (leftSpeed) : (leftSpeed));
  motor2.run((M2) == M2 ? - (rightSpeed) : (rightSpeed));
}
double angle_rad = PI / 180.0;
double angle_deg = 180.0 / PI;
int in1 = 11;
int in2 = 12;
int a = -1;
int b = -1;
void setup()
{
  Serial.begin(115200);
  pinMode(in1, OUTPUT);
  pinMode(in2, OUTPUT);
  int a = -1;
  int b = -1;
}
void loop()
{
  Serial.print("Distance: ");
  Serial.print(ultraSensor.distanceCm() );
  Serial.println(" cm");
  delay(100);
  Serial.print("Analog Value is: ");
  Serial.print(FlameSensor1.readAnalog() );
  Serial.print("----Status: ");
  delay(100);
  int speed;
  int direction;
  if (FlameSensor1.readDigital() == Fire)
  {
    if ((ultraSensor.distanceCm()) > 50)
    {
      if (FlameSensor1.readDigital() == NoFire)
      {
        while (!(FlameSensor1.readDigital() == Fire))
        {
          move(4, 70);
        }
        move(0, 0);
        delay(1000);
        return;
```

```
//5 moves are introduced: 0 stop, 1 forward, 2
backward, 3 turn left, 4 turn right

//running principle of motors 1 and 2 are defined

//conversion of degrees and radians are defined
(unnecessary for our motion)
//IN1of MOSFET is connected to PIN11 of motherboard
//IN2 of MOSFET is connected to PIN12 of motherboard
//Two integers a and b are defined as "-1" to control
the robot when to turn left and right. Each turning
motion increases or decreases a and b one by one,
using the instantaneous values of a and b, robot know
whether it needs to turn left or right.
//IN1 and IN2 on MOSFET defined as outputs
//a and b values are both  defined as "-1" again in
setup, in order to prevent it to reset their values to
initial one when the code meets a return command.

//Printing the value of instantaneous distance

//Printing the value for each 0.1 seconds

//Printing whether fire detected or not

//Printing the status for each 0.1 seconds

//When it detects fire

//When the distance is over 50 cm

//During this time if sensor loses the flame

//Until detects the fire again

//Turn right with a power of 70/255

//After detecting the fire again stop and wait for 1
second
//Turn back to the void loop() again
```

```
    }
    else                                                    //If it does not lose flame
    {
      move(1, 100);                                         //Move forward with a power of 100/255
    }

  }
  else                                                      //If the loop didn't enter the code of distance>50cm
  {
    if (((ultraSensor.distanceCm()) > 40))                  //But distance still over 40 (40<distance<50)
    {
      move(0, 0);                                           //Stop
      digitalWrite(in1, HIGH);                              //Pump works for 3 seconds
      digitalWrite(in2, LOW);
      delay(3000);
      digitalWrite(in1, LOW);                               //Pump stops for 5 seconds
      digitalWrite(in2, LOW);
      delay(5000);
      return;                                               //Turn back to void loop() again
    }
    else                                                    //If it does not enter the code of distance <>50cm
    {                                                       which means robot closer than 50 cm
      move(2, 80);                                          //Move backwards two seconds
      delay(2000);
    }

  }
}
else                                                        //If it does not enter to code of fire detected
{                                                           //And if distance measured over 25cm
  if ((ultraSensor.distanceCm()) > 25)
  {
    while (!((ultraSensor.distanceCm()) <= 25))             //Until distance is below 25 cm
    {
      move(1, 100);                                         //Go forward with a power of 100/255
      if (FlameSensor1.readDigital() == Fire)               //During this move if fire is detected
      {
        return;                                             //Go back to void loop()
      }                                                     //(To be able to apply the code for fire)
    }
  }
  else                                                      //If fire is not detected
  {
    move(0, 0);                                             //Stop for 1 second when 25cm condition applied
    delay(1000);
    direction = 0;
    if (FlameSensor1.readDigital() == Fire)                 //During this move if fire is detected
    {
      return;                                               //Go back to void loop()
    }                                                       //(To be able to apply the code for fire)
  }
  if (direction == 0 && a == -1 && b == -1)                 //When it stops, checks the values of a & b to
  {                                                         understand which side to turn
    move(4,100);                                            //if condition provided, turn right with a power of
    delay(900);                                             100/255 for 0.9 second, then move forward with a
    move(1, 100);                                           power of 100/255 for 3 seconds and change the value
    delay(3000);                                            of "a" from -1 to 0
    a = 0;
```

if (FlameSensor1.readDigital() == Fire)
{
return;
}
}
	//During this move if fire is detected
	//Go back to void loop()
	//(To be able to apply the code for fire)

if (direction == 0 && a == 0 && b == -1)
{
move(4, 100);
delay(700);
b = 0;
if (FlameSensor1.readDigital() == Fire)
{
return;
}
}
	//If fire is not detected, checks the values of a & b to understand which side to turn
	//if condition provided, turn right with a power of 100/255 for 0.9 second, then move forward with a power of 100/255 for 3 seconds and change the value of "b" from -1 to 0
	//During this move if fire is detected
	//Go back to void loop()
	//(To be able to apply the code for fire)

if (direction == 0 && a == 0 && b == 0)
{
move(3, 100);
delay(900);
move(1, 100);
delay(3000);
a = -1;
if (FlameSensor1.readDigital() == Fire)
{
return;
}
}
	//If fire is not detected, checks the values of a & b to understand which side to turn
	//if condition provided, turn left with a power of 100/255 for 0.9 second, then move forward with a power of 100/255 for 3 seconds and change the value of "a" from 0 to -1
	//During this move if fire is detected
	//Go back to void loop()
	//(To be able to apply the code for fire)

if (direction == 0 && a == -1 && b == 0)
{
move(3, 100);
delay(700);
b = -1;
if (FlameSensor1.readDigital() == Fire)
{
return;
}
return;
}
}
	//If fire is not detected, checks the values of a & b to understand which side to turn
	//if condition provided, turn left with a power of 100/255 for 0.7 second and change the value of "b" from 0 to -1
	//During this move if fire is detected
	//Go back to void loop()
	//(To be able to apply the code for fire)
	//(By controlling a and b, robot moves a path like a ±step input till it detects fire)

_loop();
}
void _loop() {
}
	//Since the existence of flame is questioned in, during and after each move, robot gets the return command and goes back to the void loop() line and goes into the code again. Then it provides the condition of detecting flame and runs that part of the code.

Table 1– Final developed code of robot with explanations on the right side

4 Implementation

Final model of robot is quite successful about extinguishing fire. It can detect fire and according to the values in the uploaded code, it takes positions according to flame and running the pump regularly.

Video 1- Robot turns the corner, detects and extinguishes fire (https://youtu.be/jIE5zaDptiM)

Video 2- Robot searches for fire in the field, detects and extinguishes fire (https://youtu.be/vksdFVt8SNk)

Video 3- Robot searches for fire in the field, detects the first one and extinguishes it, then keep searching for the second one, detects and extinguishes them both (https://youtu.be/4kjknrIrKaw)

Video 4- Robot searches for fire and detects the fire in the pan and extinguishes it (https://youtu.be/1_a9Lus1URU)

5 Performance Evaluation

First of all, the mechanical motion calculations are never fit with the motion of the robot. The calculations are done on an occupancy rate of the battery, but as the battery power decreases the calculations and robot's movements haven't tied in. Also, "stop" command added between every line on the code that changes the direction of the robot, but couldn't have reached the intended stability values.

Second problem about the robot was that, it had to make two left turns after forward moves, and then two right turns after the next two forward moves. This problem was solved with using two integers "a" and "b" as stated in the Table-1 at Control System Design part above. Changing the values of a and b from -1 to 0 and 0 to -1, with and infinite loop until the robot detects fire, it is told robot when it should turn to which side.

Another problem faced is the targeting of the trigger. Complete success hasn't achieved about that yet, but solution suggestions for that problem is available. The most appropriate method is to use a second flame sensor. Connecting two flame sensors on the front face of the robot, and setting the angle between the sensors about 50°. Since both sensors has a sensitivity of angle of 60°, the crossed area when both sensors detect the flame will become 10°. Also mounting a second sensor will be useful in the situation of loosing the flame while moving towards it. The total flame sensitivity of robot will increase up to 110°, and when one of the sensors detects the fire, robots should turn the side of that sensor slowly until the other sensor detects the flame.

The code has used the 7242 bytes of program storage space, which is equal to 22% of its maximum value. Global variables have used 870 bytes which corresponds to 42% of dynamic memory, leaving 1178 bytes for local variables.