Table of Contents

Introduction

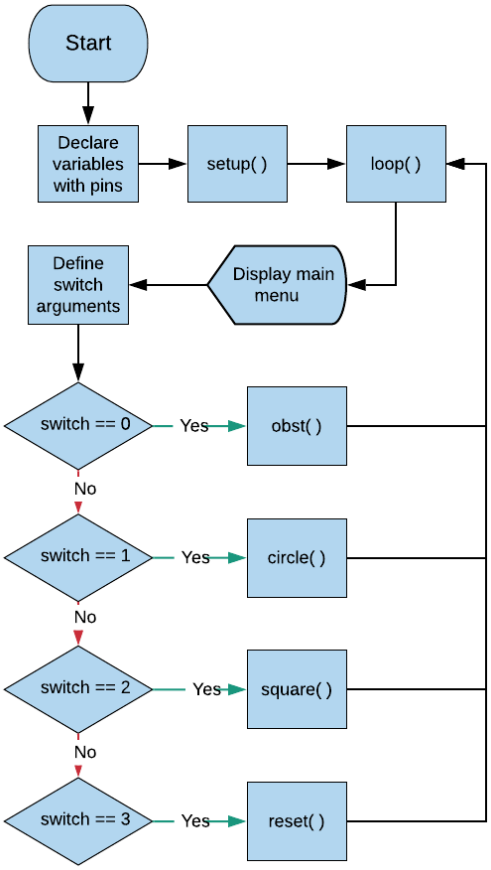
The goal of the assignment is to implement the knowledge of C, driven from the lectures and labs, into programming a small car. Using the features of the car and Arduino board and GLAM modules, one can complete a task. The modules to be used are two servomotors, one 2x16 LCD screen, two infrared sensors, one ultrasonic sensor and four switches(buttons) from the board. First part of the task was to create a start-up menu to display information about the task it can do. 1. Obstacle avoidance 2. Line following for a circle 3. Line following for a square 4. Reset. I used a tactic where I kept the sensors inside a line rather than making sensors not touching the black line.

Main function

In Arduino you start with a setup and loop functions, but to make the code cleaner I decided to create another five functions, four of which are the tasks from the assignment and fifth to print the start-up menu.

Setup function is used to initialize the inputs and attach modules to inputs. Servo motors and an ultrasonic sensor use `.attach( )` method and infrared uses `pinMode( )`. Arduino IDE initializes the LCD screen so re-initialization is not required.

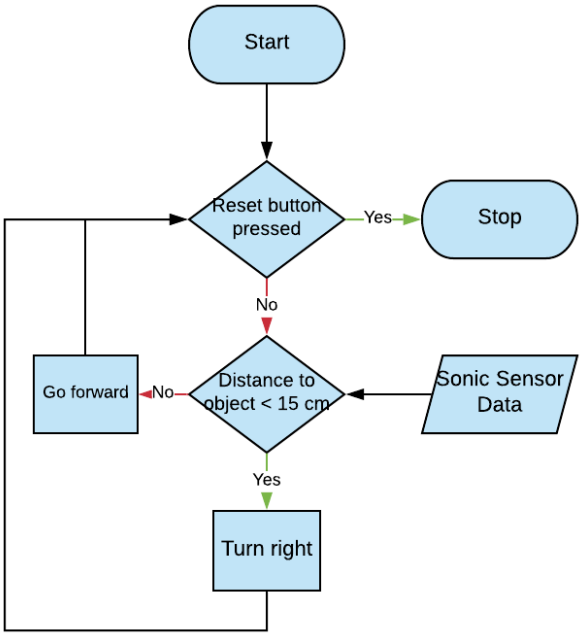
General overview



I designed my program the way it is easy to read and debug. I split the tasks into functions. The `switch` statement in the loop decides what function to run depending on what button was pressed. To define what button goes to what `case`, I used `else-if` conditions together with `ReadSwitch(SW_X)` method, where X is a number of switch(button). The flow chart doesn't have a termination statement like 'Stop' because car shuts when power is cut off.

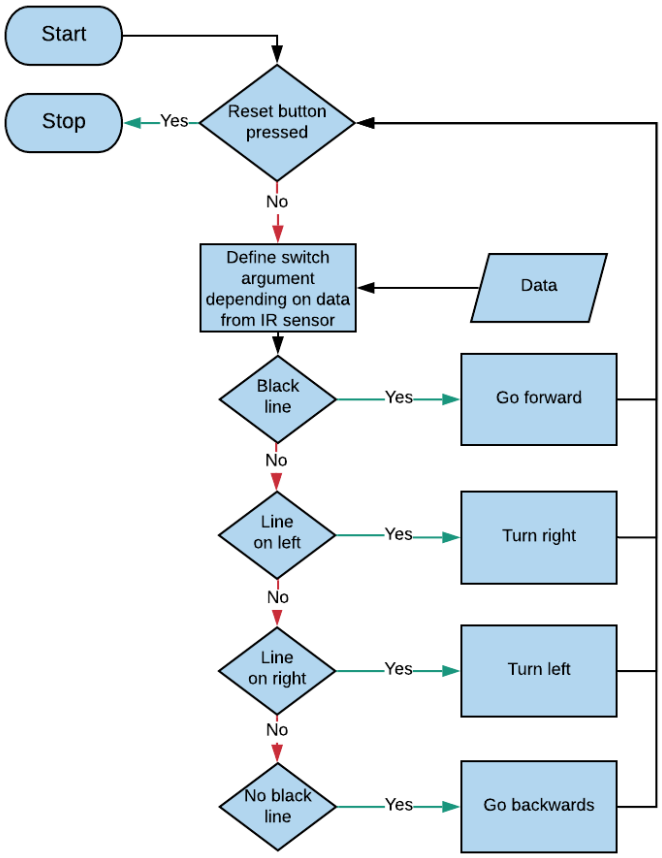
Obstacle avoiding

First task for a car in a list is obstacle avoidance. I, personally, found it the easiest task out of these three. For it I used sonic sensor and servo motors. Flow chart shows the logic of the function. In few words, the car will turn right, if it will detect any obstacle at distance of less than 15 centimeters. Otherwise it will go forward for 10ms. Small value for delay, gives high refreshing rate. It makes about a 100 degree turn. I used `delay(500)` for this. On practice the car first needs to send the sound impulse and then receive it. So, we need take a surface of an object into account, because if it is at an angle at which impulse is not received back or it absorbs the impulse, then it will be impossible to detect the object and therefore to avoid it.



Line following (Circle)

Next task was line following in circle. For line following, in this and next tasks I used infrared sensors to detect whether the is a line underneath the car. The IR works the way that, impulse won't be reflected if there is black, because it absorbs it. So, sensors will return 1 if impulse reflected and 0 if there was black. I modified the car the way that, it has IR sensors sticking out at the back, the car goes backwards too. This configuration lets car detect change further from axis of spinning (the point around which the car spins), to make a car more maneuverable.



My function starts from defining the outputs of IR sensors. Then goes the while loop `while(ReadSwitch(SW_4) == 0)` it will terminate when the switch will become active (be pressed). Inside the loop goes update of variables and depending on input, it will go forward, backward, turn left or right. When car goes forward or backwards the motors go at full speed, but when the car turns, one wheel is at full speed but opposite one slows down. This create a smooth turn without losing much speed. The commands I used for turning left and right are:

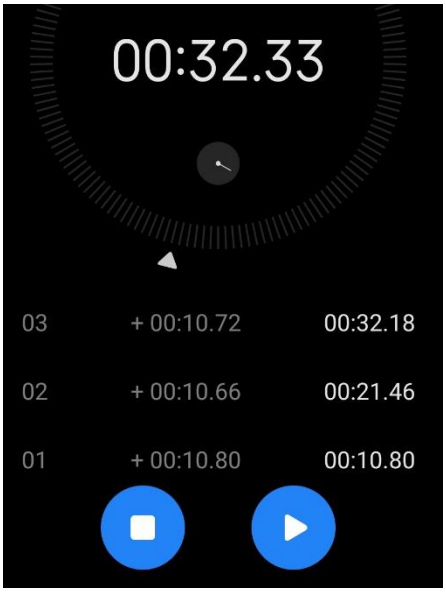
```
EmoroServo.write(servoRight, 1100);
EmoroServo.write(servoLeft, 2500);
delay(100);
```

and

```
EmoroServo.write(servoRight, 500);
EmoroServo.write(servoLeft, 1900);
delay(100);
```

respectively.

100ms for turning was chosen because it fitted the part with greater turn the best. The lower values would give a too little turn which led to touching the part without black line and slowing the car down as the angle becomes too sharp. And greater values made a car turn too much when unnecessary. For speed, I tried lower values and it led to either losing too much speed or too big turn. I timed the laps for my car. The results were as such:



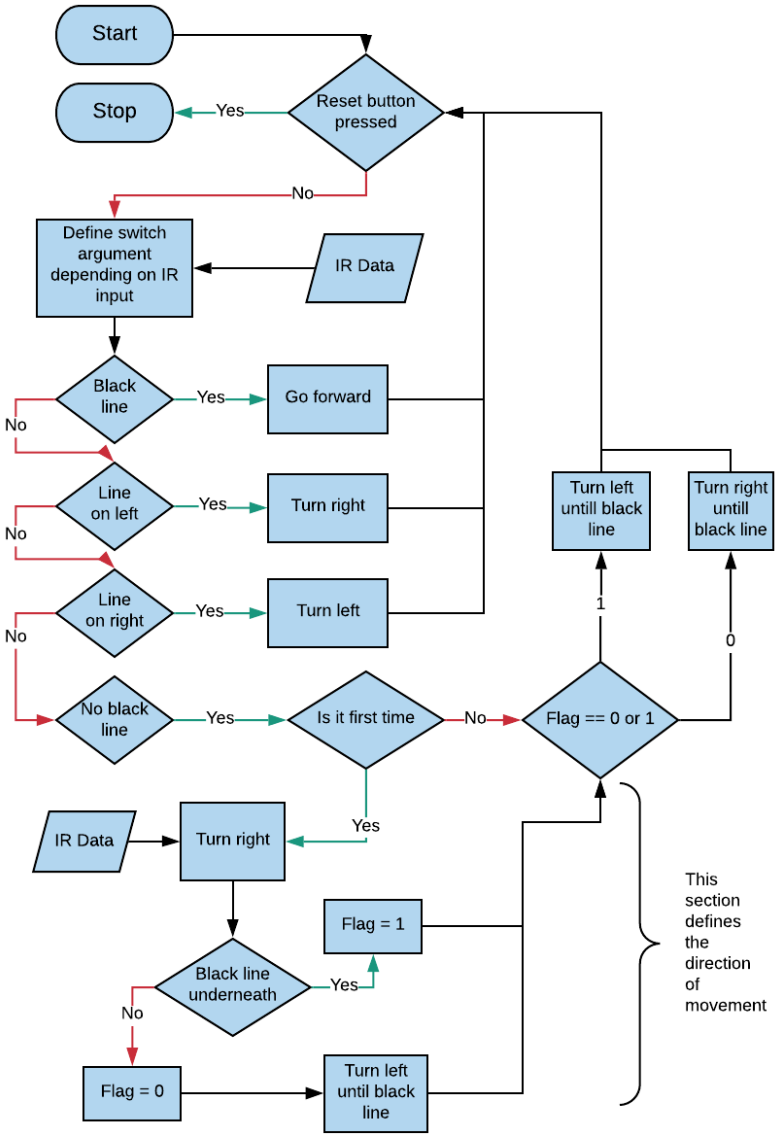
(figure 1)

(figure 2)

Figures above show stopwatch. Figure 1 shows the time taken to complete a loop when going clockwise. Every lap took about the same time. In average it took 10.72 seconds to finish one lap. Figure 2 shows the time taken for anticlockwise direction. In this case the average is 10.64.

Square

Last main task was line following in a square. This part is more challenging than others because it involved sharp turns. It is still line following so, I still used infrared sensors. I also used some part of the code from `circle()`. In particular, the part of defining the argument for switch statement. Flow chart below explains the general idea and logic of the function.



It starts from `while(ReadSwitch(SW_4) == 0)` so when the reset button is pressed the loop is interrupted. Then loop goes into switch statement to decide what it needs to do. Depending on position of the car at the moment of reading the input from infrared sensors. If no black line was detected what so ever, then it means that the car reached the corner of the road. For this case I designed the code in the way that car can remember which way it turned at the first time and this will define where the car is going to go at second turn and so on. Using this, my car would be able to go both ways and not depend on direction. But to implement this, I made it at first, turn to the right and check if there is a black line. If there is, then it's the right direction otherwise, turn left until detect the black line and go straight. Also, for turning to the right I used:

```
EmoroServo.write(servoRight, 500);
EmoroServo.write(servoLeft, 2500);
delay(130);
```

and

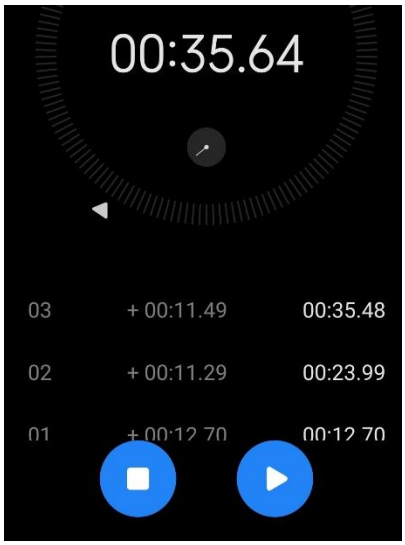
```
EmoroServo.write(servoRight, 500);
EmoroServo.write(servoLeft, 500);
delay(430);
```

I used these lines to estimate the 90 degree turn. The moving forward part is necessary to bring the wheels to the spot where lines cross and second part is a spin clockwise. I also used from 90 to 150 for delay of going forward and 400 to 500 for turning, but for some cases it spun either too far or too close and missed the line, for quite a time I used 110 and 450 combination but, then I found 130 and 430 that fitted the

situation better. Besides I used a stop for 75ms because without it the function received wrong input from sensors. After it has done the first turn, the car will keep turning at the same direction.

Also, I've timed the car to finish one loop. The results are as follows:

(figure 1)



(figure 2)

Stopwatch on the left shows the result for going around the square in a clockwise direction and on average it took about 11.29 seconds to complete one lap. Figure 2 shows the movement of the car in the counterclockwise direction. You can see it from the stopwatch. First lap took more (12.70) because it first went to right and only after it found its way and went left. From second lap on we can see that the time is about the same as for opposite direction.

Reset

Reset is a tiny function and only mentioned because its assigned to one of the buttons. All it does is, it clears the LCD screen and sets new speed to zero to stop the movement of the car.

```
Lcd.clear();
EmoroServo.write(servoRight, 1500);
EmoroServo.write(servoLeft, 1500);
```

Conclusion

The assignment taught me how to control embedded system with code. It was a more practical, more visualized, use of knowledge of C language, than assignment one. Experience from first assignment let me create a better design that would, firstly, take less space. Secondly, run faster. Thirdly, be easier to read. Nevertheless, I think that if I would be given to do this task again I would use gyroscope to turn for an exact angle and not just estimating it. Also, I believe I improved the commenting in the code and creating flow charts for assignment report.

References

No sources were used other than standard EmoRo list of functions(library):

https://moodle.essex.ac.uk/pluginfile.php/711215/mod_resource/content/0/EMoRo2560%20library.pdf

Code

```
int servoRight = SERVO_0;   //Declare Servos
int servoLeft = SERVO_1;

int infraRight = IO_0;      //Declare Infrared Sensors
int infraLeft = IO_1;

int sonic = GPP_0;      //Declare Ultrasonic Sensors

void setup() {
  InitEmoro(); // Initializes all available inputs and outputs on EMoRo 2560.

  Serial.begin(9600);

  //Attach infrared sensors
  pinMode(infraRight, INPUT_PULLUP);
  pinMode(infraLeft, INPUT_PULLUP);

  //Attach servos
  EmoroServo.attach(servoRight);
  EmoroServo.attach(servoLeft);

  //Attach ultrasonic sensors
  Ultrasonic.attach(sonic);
}

//Obstacle avoidance
void obst(){//-----------------------------------------------------------------Obst
  while(ReadSwitch(SW_4) == 0){
    Lcd.clear();
    Lcd.locate(0,0);
    Lcd.print("moving");
    Lcd.locate(1, 0);
    Lcd.print("forward.");

    int wait = 10;
    int speedR = 2500, speedL = 500; //speed
    int dis = Ultrasonic.read(sonic);//distance from ultrasonic sensor

    if(dis < 15){
      Lcd.clear();
      Lcd.locate(0,0);
      Lcd.print("obstacle");
      Lcd.locate(1,0);
      Lcd.print("detected.");
      //turn right
      speedR = 500;
      speedL = 500;
      wait = 500;
    }
```

```
    EmoroServo.write(servoRight, speedR);
    EmoroServo.write(servoLeft, speedL);
    delay(wait);
  }
}//--------------------------------------------------------------------Obst

//Follow the line(CIRCLE)
void circle(){//-----------------------------------------------------Circle
  //For line following in a circle use car backwards,
  //therefore car goes backwards pretending its a front
  int left, right; //infrared readings

  Lcd.clear();
  Lcd.locate(0,0);
  Lcd.print("Cycle Track");
  Lcd.locate(1,0);
  Lcd.print("Following");

  while(ReadSwitch(SW_4) == 0){
    left = digitalRead(infraLeft);
    right = digitalRead(infraRight);
    int movement;

      if(left == 0 && right == 0){
          movement = 0; //forward
      }else if(left == 1 && right == 0) {
          movement = 1; //right
      }else if(left == 0 && right == 1) {
          movement = 2; //left
      }else if(left == 1 && right == 1) {
          movement = 3; //none
      }

      switch(movement){
          case 0: //forward
              EmoroServo.write(servoRight, 500);
              EmoroServo.write(servoLeft, 2500);
              delay(10);
              break;
          case 1: //right
              EmoroServo.write(servoRight, 500);
              EmoroServo.write(servoLeft, 1900);
              delay(100);
              break;
          case 2: //left
              EmoroServo.write(servoRight, 1100);
              EmoroServo.write(servoLeft, 2500);
              delay(100);
              break;
          case 3: //no black line (backwards)
              EmoroServo.write(servoRight, 2500);
              EmoroServo.write(servoLeft, 500);
              delay(10);
              break;
      }
  }
```

```
}//-------------------------------------------------------------------------Circle

//Follow the line(SQUARE)
void square(){//-----------------------------------------------------------Square
    //For line following in a square use car backwards,
    //therefore car goes backwards pretending its a front
    int movement;
    int count = 0; //how many times it turned
    int flag; //if == 0 then turn left, if == 1 then turn right
    int left, right; //infrared readings

    while(ReadSwitch(SW_4) == 0){

        left = digitalRead(infraLeft);
        right = digitalRead(infraRight);

        if(left == 0 && right == 0){
            movement = 0; //forward
        }else if(left == 1 && right == 0) {
            movement = 1; //right
        }else if(left == 0 && right == 1) {
            movement = 2; //left
        }else if(left == 1 && right == 1) {
            movement = 3; //none
        }

        switch(movement){
            case 0: //forward
                EmoroServo.write(servoRight, 500);
                EmoroServo.write(servoLeft, 2500);
                delay(10);
                Lcd.clear();
                Lcd.locate(0,0);
                Lcd.print("Square Track");
                Lcd.locate(1,0);
                Lcd.print("Following");
                break;

            case 1: //right
                EmoroServo.write(servoRight, 500);
                EmoroServo.write(servoLeft, 2000);
                delay(50);
                break;

            case 2: //left
                EmoroServo.write(servoRight, 1000);
                EmoroServo.write(servoLeft, 2500);
                delay(50);
                break;

            case 3: //corner
                count++;
                if(count == 1){
                    //go forward little bit more
                    EmoroServo.write(servoRight, 500);
                    EmoroServo.write(servoLeft, 2500);
                    delay(130);
```

```
                    //turn 90 deg right
                    EmoroServo.write(servoRight, 500);
                    EmoroServo.write(servoLeft, 500);
                    delay(430);

                    //needs to have stop to have time to take readings
                    EmoroServo.write(servoRight, 1500);
                    EmoroServo.write(servoLeft, 1500);
                    delay(75);

                    //take a reading
                    left = digitalRead(infraLeft);
                    right = digitalRead(infraRight);

                    //if there is no black line...
                    if(left && right){
                        flag = 0; //left

                        left = digitalRead(infraLeft);
                        right = digitalRead(infraRight);

                        //turn left until find blackline
                        while(left && right){
                            //turn 180 opposite direction to check if there
is a line

                            EmoroServo.write(servoRight, 2500);
                            EmoroServo.write(servoLeft, 2500);
                            delay(10);

                            //take a reading again
                            left = digitalRead(infraLeft);
                            right = digitalRead(infraRight);
                        }

                    //...but if there is a line (NAND operand)
                    }else if(!(left && right)){
                        flag = 1; //right
                    }
                    left = digitalRead(infraLeft);
                    right = digitalRead(infraRight);
                }

                if(flag == 0){  //go anticlockwise(left)
                    EmoroServo.write(servoRight, 2500);
                    EmoroServo.write(servoLeft, 2500);
                    delay(20);

                    Lcd.clear();
                    Lcd.locate(0,0);
                    Lcd.print("Sharp turning");
                    Lcd.locate(1,0);
                    Lcd.print("-90 degrees.");

                }else if(flag == 1){ //go clockwise(right)
                    EmoroServo.write(servoRight, 500);
                    EmoroServo.write(servoLeft, 500);
```

```
                    delay(20);

                    Lcd.clear();
                    Lcd.locate(0,0);
                    Lcd.print("Sharp turning");
                    Lcd.locate(1,0);
                    Lcd.print("90 degrees.");
                }
                break;
        }
    }
}//--------------------------------------------------------------------Square

//Reset the conditions, go back to the menu
void reset(){//-----------------------------------------------------------Reset
    //wipe out LCD
    Lcd.clear();

    //Sets new speed to 0
    EmoroServo.write(servoRight, 1500);
    EmoroServo.write(servoLeft, 1500);
}//--------------------------------------------------------------------Reset

//Printing the menu
void initPrint() {//-----------------------------------------------initPrint
    //Column 1
    Lcd.locate(0, 0);     //Row 1
    Lcd.print("1.Obst");
    Lcd.locate(1, 0);     //Row 2
    Lcd.print("3.Square");

    //Column 2
    Lcd.locate(0, 8);     //Row 1
    Lcd.print("2.Circle");
    Lcd.locate(1, 8);     //Row 2
    Lcd.print("4.Reset");
}//--------------------------------------------------------------------initPrint

void loop() {
    initPrint();

    int button = 0;

    if(ReadSwitch(SW_1) == 1){
        button = 1;
    }else if(ReadSwitch(SW_2) == 1){
        button = 2;
    }else if(ReadSwitch(SW_3) == 1) {
        button = 3;
    }else if(ReadSwitch(SW_4) == 1) {
        button = 4;
    }

    switch(button) {
        case 1:
            obst();
            break;
```

```
            case 2:
                circle();
                break;
            case 3:
                square();
                break;
            case 4:
                reset();
                break;
        }
    }
```