Term Project

Drexel University

To: Dr. Christopher Peters

From: Ananiya Deneke

Date: 03/11/2022

Re: ECE 303 Lab Memo

Term Project

Purpose/Project Description

The purpose of project was to develop an electric vehicle test bed. This project has five areas of focus:

- Simulating a collision avoidance system with an ultrasonic sensor and a DC motor
- Security and Remote control with an RFID system, a buzzer, and an IR remote to control LED brightness and motor speed.
- Water and temperature sensor implementation to shut down the system in cases of high temperatures or low water level.
- An LCD screen display to display motor speed, coolant level, and battery temperature
- A GUI on MATLAB for displaying motor speed, LED brightness status, Low coolant alarm, and temperature display.

Parts Required

Hardware:

Breadboard, Arduino, Ultrasonic Sensor, DC Motor, LED's, buzzer, power source, RFID module, IR transmitter module, DHT temperature and water sensor, IR remote, IR receiver, and jumper wires

Software:

MATLAB, Arduino IDE

Functionality

We will start by setting up an ultrasonic sensor that signals the Motor to change speed based on how far the object is in front of the sensor. We will also setup three LEDs, so they turn on/off based on the same parameters. After that we connect the DHT sensor to the board so it can detect temperature and water values and printed them to the serial monitor or send the values to the GUI/LCD . Next, we setup a security system with the RFID module to allow only authorized users to access the test bed, we signal this to the user by using a buzzer that produces different sound frequencies based on user authorization. After that, we connect the IR receiver to the circuit and tested the buttons, after knowing which buttons to use, we use those buttons to change motor speed in 10% increments, and to change the brightness of two LED's that we use as headlights. Then, we connect the LCD display the circuit and slightly change our previous code to output the temperature and water level to the LCD instead of the serial monitor. Finally, we will be setting up a GUI in MATLAB with serial communication to the

Arduino, and we will be outputting, distance (calculated from sensor), Headlights, Motor Speed, Low coolant alarm, temperature display and alarm.

Current/Voltage budget

- Ultrasonic Sensor – 5V
- H bridge- 5V
- DC motor- 5V
- DHT water and temperature sensor- 5V
- RFID Module- 3.3V
- Arduino board- 5V
- IR remote – 3.3V
- LCD – 5V

Data transfer to computer/GUI

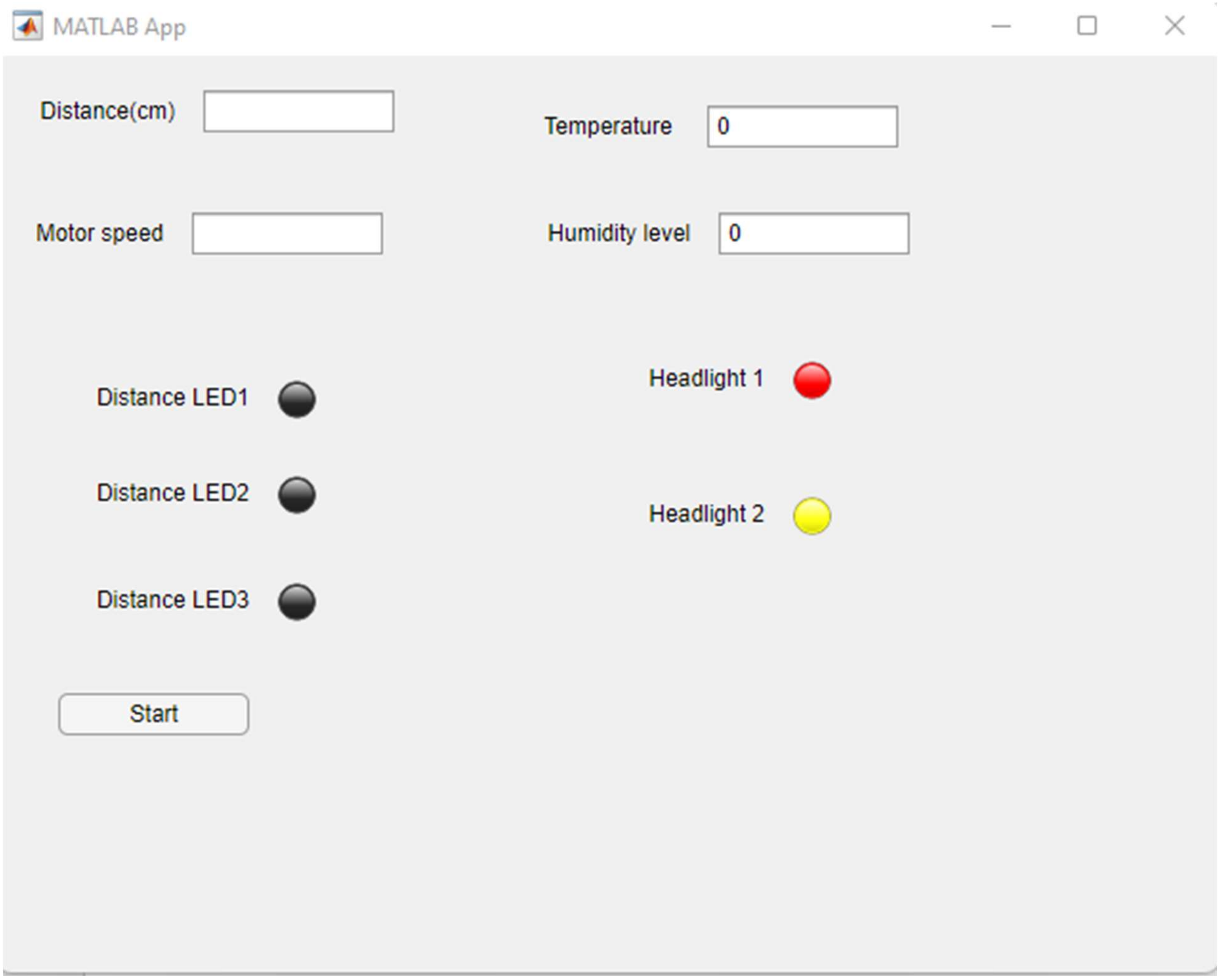
I started by uploading the code I wrote for each component (appended below) from the Arduino IDE to Arduino. I setup the code so that we output the following data to the serial monitor and send it to MATLAB's GUI through serial communication:

- distance of object from ultrasonic sensor,
- Motor" Speed "in percentages – when implementing the IR remote,
- temperature and humidity level to the Serial monitor.

On MATLAB, I started by creating a blank layout for our GUI and four field boxes, for temperature, humidity level, motor speed, and distance. I also added 5 lamps, 3 for the Distance LEDs and two for headlights. After I setup the layout, I created a Start-call back and coded our main MATLAB code in that section. I sent the data I need, as a string, through my serial port to MATLAB. Finally, I setup our MATLAB code to perform calculations on the value it reads from the Arduino, and change the color of the lamps, while also recording the other data points

Term Project

GUI on Matlab



MATLAB App

Distance(cm)

Temperature 0

Motor speed

Humidity level 0

Distance LED1 ●

Distance LED2 ●

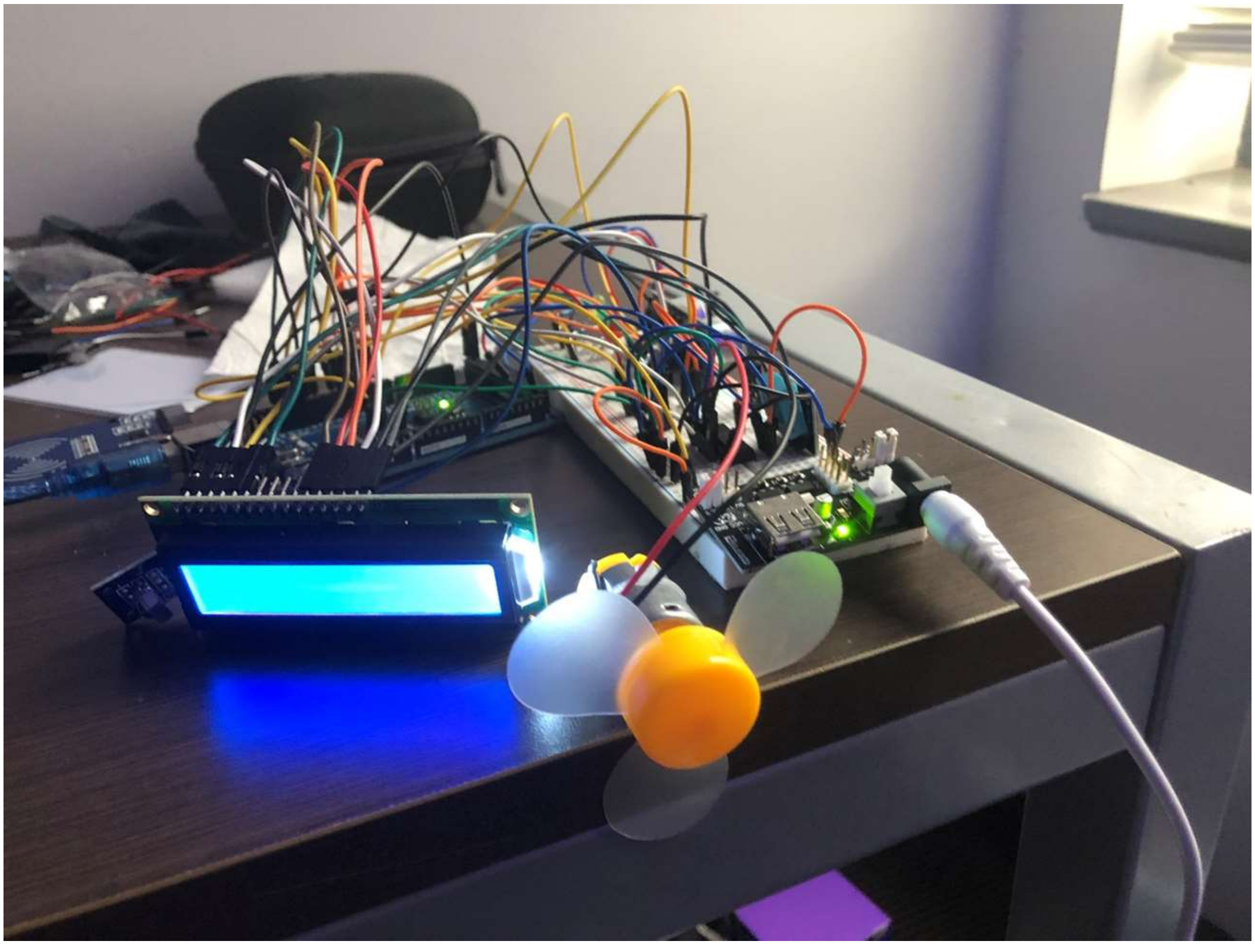
Distance LED3 ●

Headlight 1 ●

Headlight 2 ●

Term Project

Test bed Display



Appendices

Arduino code:

RFID

```
#include <SPI.h>
#include <MFRC522.h>
#define RST_PIN 9        // Configurable, see typical pin layout above
#define SS_PIN  53
MFRC522 mfrc522(SS_PIN, RST_PIN);  // Create MFRC522 instance
const int buzzer = 42;
void setup() {
        Serial.begin(9600);
        SPI.begin();                        // Init SPI bus
        mfrc522.PCD_Init();                 // Init MFRC522
  pinMode(buzzer, OUTPUT);
}


void loop() {
  String content= "";
  byte letter;
        if ( ! mfrc522.PICC_IsNewCardPresent()) { // Reset the loop if no new card present on the sensor/reader. This saves the entire process when idle.
                return;
        }
  if ( ! mfrc522.PICC_ReadCardSerial()) {    //select one of the cards
                return;
        }
  for (byte i = 0; i < mfrc522.uid.size; i++)
  {
    Serial.print(mfrc522.uid.uidByte[i] < 0x10 ? " 0" : " ");
    Serial.print(mfrc522.uid.uidByte[i], HEX);
    content.concat(String(mfrc522.uid.uidByte[i] < 0x10 ? " 0" : " "));
    content.concat(String(mfrc522.uid.uidByte[i], HEX));
  }
  Serial.println();
  Serial.print("Message : ");
  content.toUpperCase();
  if (content.substring(1) == "4B 9B E2 4C"){
    Serial.println("Authorized access");
    Serial.println();
    tone(buzzer, 5000);
    delay(1000);
    noTone(buzzer);
    delay(3000);
  }

 else {
    Serial.println(" Access denied");
    tone(buzzer, 1000);
    delay(1000);
    noTone(buzzer);
    delay(3000);
  }
}
```

Interlock, LCD, and ultrasonic sensor/motor

```
#include <dht.h>        //library for DHT11 sensor
#include <LiquidCrystal.h>
#define dht_apin A15
dht DHT;
LiquidCrystal lcd(12,13,24,25,26);
const int trigPin = 9;
const int echoPin = 10;
int forward = 7;
const int backward = 5;
long duration;
```

Term Project

```
int distance;

int LED1 = 2;

int LED2 = 3;

int LED3 = 4;

LiquidCrystal lcd(

void setup() {

  lcd.begin(16, 4);

  digitalWrite(LED1,HIGH);

  pinMode(trigPin, OUTPUT); // Sets the trigPin as an Output

  pinMode(echoPin, INPUT); // Sets the echoPin as an Input

  Serial.begin(9600); // Starts the serial communication

  delay(500);

  pinMode(forward,OUTPUT);

  pinMode(LED1,OUTPUT);

  pinMode(LED2,OUTPUT);

  pinMode(LED3,OUTPUT);

  pinMode(backward,OUTPUT);

}

void loop() {

  DHT.read11(dht_apin);               // read temperature and water sensor

  lcd.setCursor(0,0);

  lcd.write(DHT.humidity);

  lcd.setCursor(0,1);

  lcd.write(DHT.temperature);

  //Serial.println(DHT.temperature);

  delayMicroseconds(2);

  digitalWrite(trigPin, HIGH);

  delayMicroseconds(10);

  digitalWrite(trigPin, LOW);

  duration = pulseIn(echoPin, HIGH);

  distance = duration * 0.034 / 2;

  lcd.setCursor(0,2);

  lcd.write(distance);}
```

```
 Serial.println(distance);
 digitalWrite(LED1, LOW);
 digitalWrite(LED2, LOW);
 digitalWrite(LED3, LOW);
 analogWrite(backward, 0);

 if( (distance >= 30) && (distance < 60)){
  digitalWrite(LED1, HIGH);
 // analogWrite(forward, 255);
 }
 if((distance > 7) && (distance <30)){
   digitalWrite(LED2, HIGH);
   digitalWrite(LED1, HIGH);
   analogWrite(forward, 120);


 }
 if( distance <= 7){
   digitalWrite(LED3, HIGH);
    digitalWrite(LED2, HIGH);
   digitalWrite(LED1, HIGH);
   analogWrite(forward, 0);
 }
 delay(1000);
 }
```

IR Remote

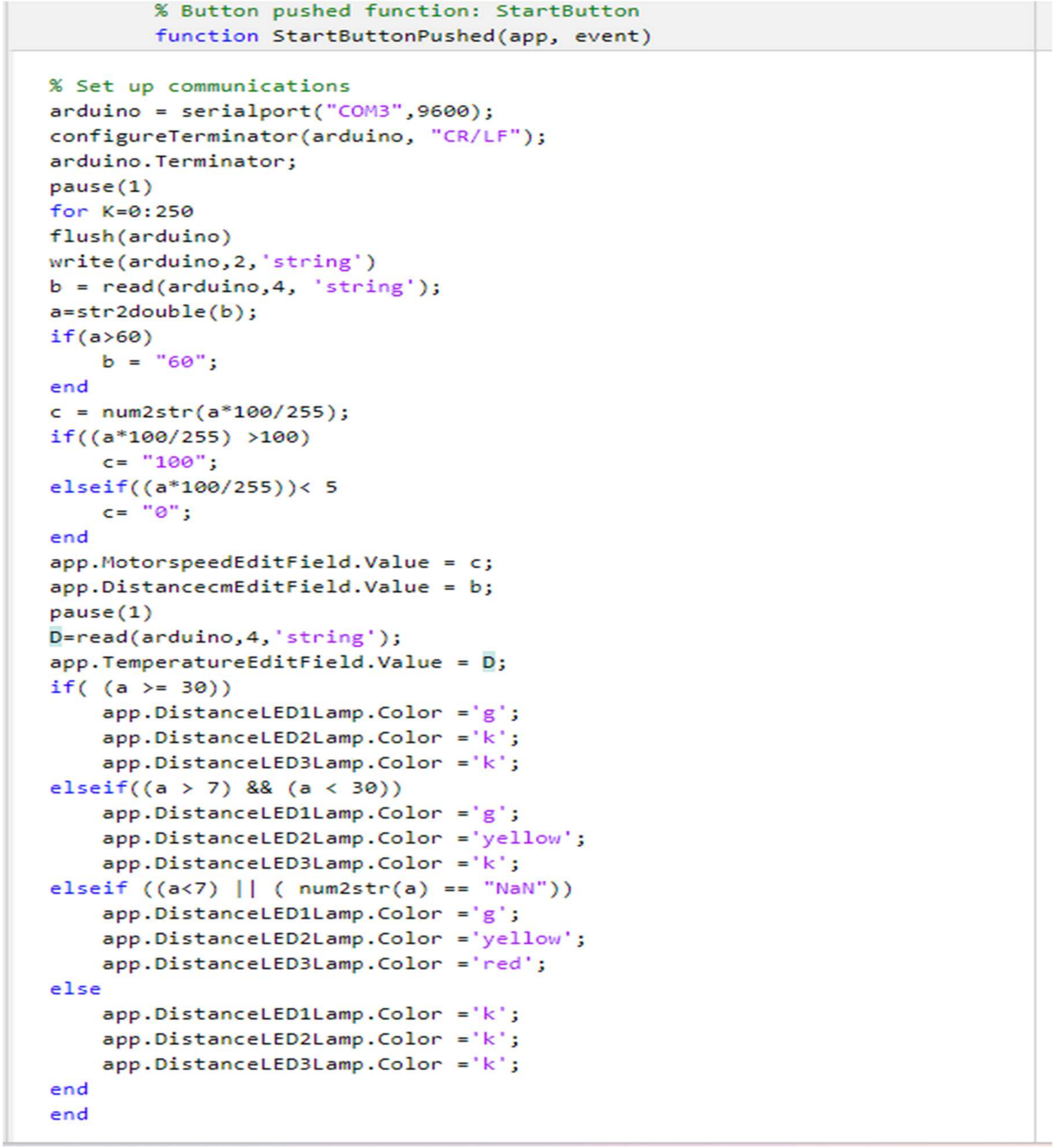
```
#include <IRremote.h> // use the library for IR


const int receiver = 11; // pin 1 of IR receiver to Arduino digital pin 11
const int ledPin =  4;
const int forward = 7;
int Led = 4;
```

Term Project

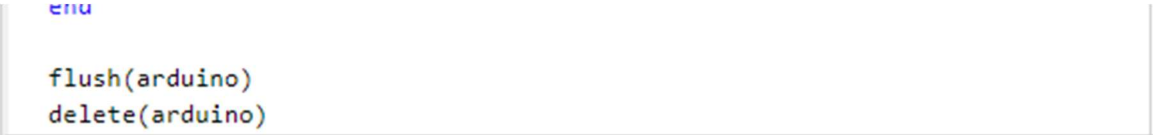
```
int fadeValue;

int motorSpeed = 0;

int lastCounter = 1;

int counter;


IRrecv irrecv(receiver); // create instance of 'irrecv'


decode_results results;


void setup()


{

  Serial.begin(9600);

  pinMode(ledPin, OUTPUT);

  pinMode(forward,OUTPUT);

  irrecv.enableIRIn();

}


void loop()

{

  counter = lastCounter;

  if (irrecv.decode(&results))

  {

    if (results.value == 16724175) // Button 1 pressed

    {

      counter ++;

    }

    if (results.value == 16718055) // Button 2 pressed

    {

      counter --;

    }

  irrecv.resume();

  }
  if (counter == 5)
    {
      digitalWrite( Led , HIGH);
      delay(500);
      digitalWrite( Led , LOW);
      delay(100);
    }
if (counter > 5){      //maximum for counter = 5
   counter = 1;
 }

 if (counter < 2){    //minimum for counter = 1
   counter = 1;
 }

 switch (counter){    //depending on the counter the fadevalue is sent to the led and the motor speed

 case 1:
 fadeValue = 00;
 break;

 case 2:
 fadeValue = 50;
 break;

 case 3:
 fadeValue = 120;
 break;

 case 4:
 fadeValue = 185;
 break;
 case 5:
 fadeValue = 255;
 break;


 }
 if( results.value == 16743045 ) {
   analogWrite(forward, motorSpeed);
   motorSpeed = motorSpeed + 25.5;        //ten percent increments
   delay(500);
 }
 analogWrite(ledPin , fadeValue);    //set led with PWM value
 lastCounter=counter;


}
```

Term Project

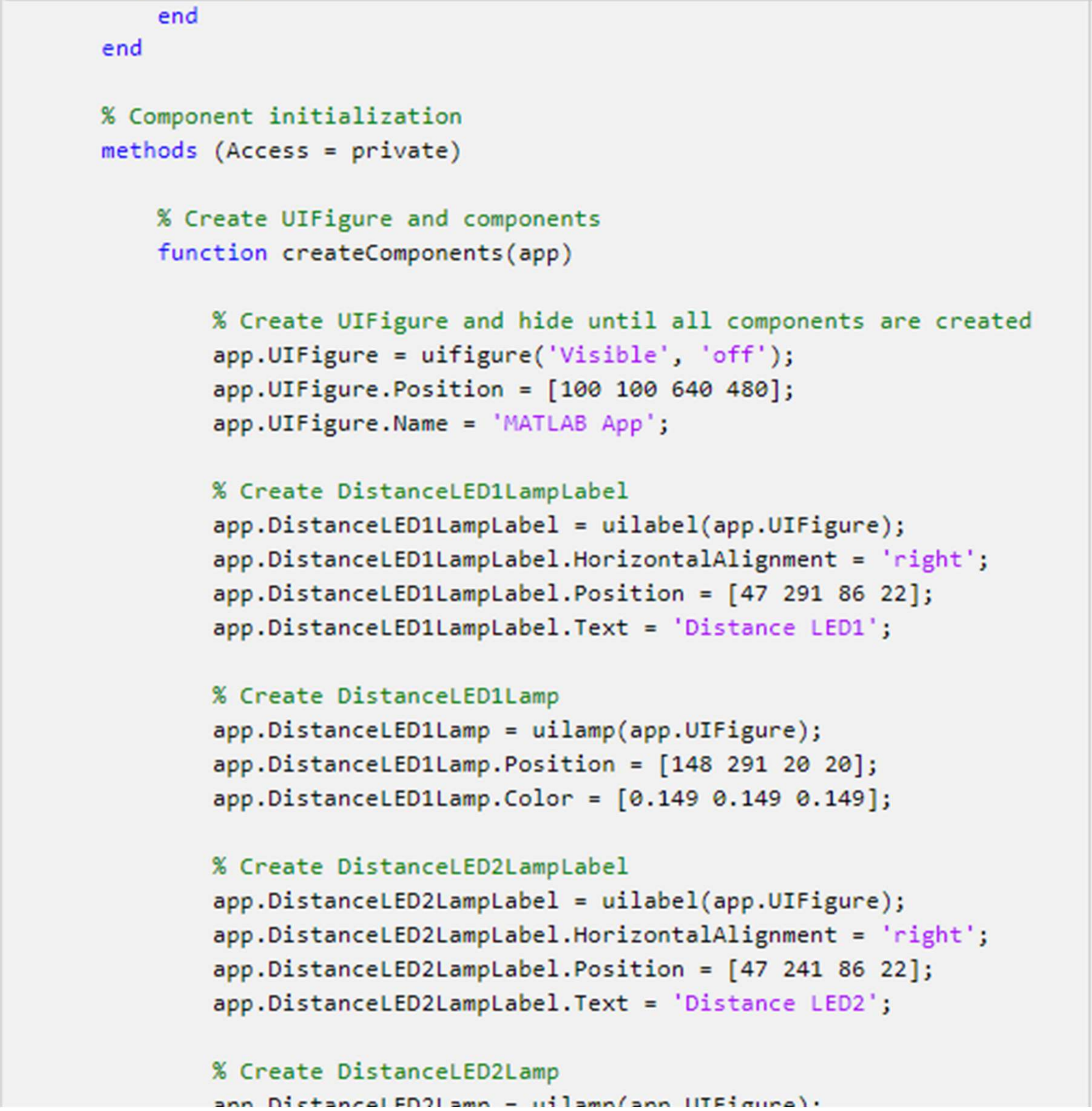
MATLAB GUI code



```
        % Button pushed function: StartButton
        function StartButtonPushed(app, event)

% Set up communications
arduino = serialport("COM3",9600);
configureTerminator(arduino, "CR/LF");
arduino.Terminator;
pause(1)
for K=0:250
flush(arduino)
write(arduino,2,'string')
b = read(arduino,4, 'string');
a=str2double(b);
if(a>60)
    b = "60";
end
c = num2str(a*100/255);
if((a*100/255) >100)
    c= "100";
elseif((a*100/255))< 5
    c= "0";
end
app.MotorspeedEditField.Value = c;
app.DistancecmEditField.Value = b;
pause(1)
D=read(arduino,4,'string');
app.TemperatureEditField.Value = D;
if( (a >= 30))
    app.DistanceLED1Lamp.Color ='g';
    app.DistanceLED2Lamp.Color ='k';
    app.DistanceLED3Lamp.Color ='k';
elseif((a > 7) && (a < 30))
    app.DistanceLED1Lamp.Color ='g';
    app.DistanceLED2Lamp.Color ='yellow';
    app.DistanceLED3Lamp.Color ='k';
elseif ((a<7) || ( num2str(a) == "NaN"))
    app.DistanceLED1Lamp.Color ='g';
    app.DistanceLED2Lamp.Color ='yellow';
    app.DistanceLED3Lamp.Color ='red';
else
    app.DistanceLED1Lamp.Color ='k';
    app.DistanceLED2Lamp.Color ='k';
    app.DistanceLED3Lamp.Color ='k';
end
end
```



```
end

flush(arduino)
delete(arduino)
```



```
            end
        end

        % Component initialization
        methods (Access = private)

            % Create UIFigure and components
            function createComponents(app)

                % Create UIFigure and hide until all components are created
                app.UIFigure = uifigure('Visible', 'off');
                app.UIFigure.Position = [100 100 640 480];
                app.UIFigure.Name = 'MATLAB App';

                % Create DistanceLED1LampLabel
                app.DistanceLED1LampLabel = uilabel(app.UIFigure);
                app.DistanceLED1LampLabel.HorizontalAlignment = 'right';
                app.DistanceLED1LampLabel.Position = [47 291 86 22];
                app.DistanceLED1LampLabel.Text = 'Distance LED1';

                % Create DistanceLED1Lamp
                app.DistanceLED1Lamp = uilamp(app.UIFigure);
                app.DistanceLED1Lamp.Position = [148 291 20 20];
                app.DistanceLED1Lamp.Color = [0.149 0.149 0.149];

                % Create DistanceLED2LampLabel
                app.DistanceLED2LampLabel = uilabel(app.UIFigure);
                app.DistanceLED2LampLabel.HorizontalAlignment = 'right';
                app.DistanceLED2LampLabel.Position = [47 241 86 22];
                app.DistanceLED2LampLabel.Text = 'Distance LED2';

                % Create DistanceLED2Lamp
                app.DistanceLED2Lamp = uilamp(app.UIFigure);
```