#define light\_FR 14 //LED Front Right pin A0 for Arduino Uno

#define light\_FL 15 //LED Front Left pin A1 for Arduino Uno

#define light\_BR 16 //LED Back Right pin A2 for Arduino Uno

//#define light\_BL 17 LED Back Left pin A3 for Arduino Uno

#define horn\_Buzz 18 //Horn Buzzer pin A4 for Arduino Uno

#define ENA\_m1 5 // Enable/speed motor Front Right

#define ENB\_m1 6 // Enable/speed motor Back Right

#define ENA\_m2 10 // Enable/speed motor Front Left

#define ENB\_m2 11 // Enable/speed motor Back Left

#define IN\_11 2 // L298N #1 in 1 motor Front Right

#define IN\_12 3 // L298N #1 in 2 motor Front Right

#define IN\_13 4 // L298N #1 in 3 motor Back Right

#define IN\_14 7 // L298N #1 in 4 motor Back Right

#define IN\_21 8 // L298N #2 in 1 motor Front Left

#define IN\_22 9 // L298N #2 in 2 motor Front Left

#define IN\_23 12 // L298N #2 in 3 motor Back Left

#define IN\_24 13 // L298N #2 in 4 motor Back Left

int command; //Int to store app command state.

int speedCar = 100;

int speedCarM = 95;// 50 - 255.

int speed\_Coeff = 4;

const int trigPin = A5;

const int echoPin = A3;

// defines variables

long duration;

int distance;

int safetyDistance;

boolean lightFront = false;

boolean lightBack = false;

boolean horn = false;

void setup() {

pinMode(light\_FR, OUTPUT);

pinMode(light\_FL, OUTPUT);

pinMode(light\_BR, OUTPUT);

pinMode(light\_BR, OUTPUT);

pinMode(horn\_Buzz, OUTPUT);

pinMode(ENA\_m1, OUTPUT);

pinMode(ENB\_m1, OUTPUT);

pinMode(ENA\_m2, OUTPUT);

pinMode(ENB\_m2, OUTPUT);

pinMode(IN\_11, OUTPUT);

pinMode(IN\_12, OUTPUT);

pinMode(IN\_13, OUTPUT);

pinMode(IN\_14, OUTPUT);

pinMode(IN\_21, OUTPUT);

pinMode(IN\_22, OUTPUT);

pinMode(IN\_23, OUTPUT);

pinMode(IN\_24, OUTPUT);

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

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

Serial.begin(9600);

}

void goAhead(){

digitalWrite(IN\_11, HIGH);

digitalWrite(IN\_12, LOW);

analogWrite(ENA\_m1, speedCar);

digitalWrite(IN\_13, LOW);

digitalWrite(IN\_14, HIGH);

analogWrite(ENB\_m1, speedCar);

digitalWrite(IN\_21, LOW);

digitalWrite(IN\_22, HIGH);

analogWrite(ENA\_m2, speedCar);

digitalWrite(IN\_23, HIGH);

digitalWrite(IN\_24, LOW);

analogWrite(ENB\_m2, speedCar);

}

void goBack(){

digitalWrite(IN\_11, LOW);

digitalWrite(IN\_12, HIGH);

analogWrite(ENA\_m1, speedCar);

digitalWrite(IN\_13, HIGH);

digitalWrite(IN\_14, LOW);

analogWrite(ENB\_m1, speedCar);

digitalWrite(IN\_21, HIGH);

digitalWrite(IN\_22, LOW);

analogWrite(ENA\_m2, speedCar);

digitalWrite(IN\_23, LOW);

digitalWrite(IN\_24, HIGH);

analogWrite(ENB\_m2, speedCar);

}

void goBackM(){

digitalWrite(IN\_11, LOW);

digitalWrite(IN\_12, HIGH);

analogWrite(ENA\_m1, speedCarM);

digitalWrite(IN\_13, HIGH);

digitalWrite(IN\_14, LOW);

analogWrite(ENB\_m1, speedCarM);

digitalWrite(IN\_21, HIGH);

digitalWrite(IN\_22, LOW);

analogWrite(ENA\_m2, speedCarM);

digitalWrite(IN\_23, LOW);

digitalWrite(IN\_24, HIGH);

analogWrite(ENB\_m2, speedCarM);

}

void goRight(){

digitalWrite(IN\_11, LOW);

digitalWrite(IN\_12, HIGH);

analogWrite(ENA\_m1, speedCar);

digitalWrite(IN\_13, HIGH);

digitalWrite(IN\_14, LOW);

analogWrite(ENB\_m1, speedCar);

digitalWrite(IN\_21, LOW);

digitalWrite(IN\_22, HIGH);

analogWrite(ENA\_m2, speedCar);

digitalWrite(IN\_23, HIGH);

digitalWrite(IN\_24, LOW);

analogWrite(ENB\_m2, speedCar);

}

void goLeft(){

digitalWrite(IN\_11, HIGH);

digitalWrite(IN\_12, LOW);

analogWrite(ENA\_m1, speedCar);

digitalWrite(IN\_13, LOW);

digitalWrite(IN\_14, HIGH);

analogWrite(ENB\_m1, speedCar);

digitalWrite(IN\_21, HIGH);

digitalWrite(IN\_22, LOW);

analogWrite(ENA\_m2, speedCar);

digitalWrite(IN\_23, LOW);

digitalWrite(IN\_24, HIGH);

analogWrite(ENB\_m2, speedCar);

}

void goAheadRight(){

digitalWrite(IN\_11, HIGH);

digitalWrite(IN\_12, LOW);

analogWrite(ENA\_m1, speedCar/speed\_Coeff);

digitalWrite(IN\_13, LOW);

digitalWrite(IN\_14, HIGH);

analogWrite(ENB\_m1, speedCar/speed\_Coeff);

digitalWrite(IN\_21, LOW);

digitalWrite(IN\_22, HIGH);

analogWrite(ENA\_m2, speedCar);

digitalWrite(IN\_23, HIGH);

digitalWrite(IN\_24, LOW);

analogWrite(ENB\_m2, speedCar);

}

void goAheadLeft(){

digitalWrite(IN\_11, HIGH);

digitalWrite(IN\_12, LOW);

analogWrite(ENA\_m1, speedCar);

digitalWrite(IN\_13, LOW);

digitalWrite(IN\_14, HIGH);

analogWrite(ENB\_m1, speedCar);

digitalWrite(IN\_21, LOW);

digitalWrite(IN\_22, HIGH);

analogWrite(ENA\_m2, speedCar/speed\_Coeff);

digitalWrite(IN\_23, HIGH);

digitalWrite(IN\_24, LOW);

analogWrite(ENB\_m2, speedCar/speed\_Coeff);

}

void goBackRight(){

digitalWrite(IN\_11, LOW);

digitalWrite(IN\_12, HIGH);

analogWrite(ENA\_m1, speedCar/speed\_Coeff);

digitalWrite(IN\_13, HIGH);

digitalWrite(IN\_14, LOW);

analogWrite(ENB\_m1, speedCar/speed\_Coeff);

digitalWrite(IN\_21, HIGH);

digitalWrite(IN\_22, LOW);

analogWrite(ENA\_m2, speedCar);

digitalWrite(IN\_23, LOW);

digitalWrite(IN\_24, HIGH);

analogWrite(ENB\_m2, speedCar);

}

void goBackLeft(){

digitalWrite(IN\_11, LOW);

digitalWrite(IN\_12, HIGH);

analogWrite(ENA\_m1, speedCar);

digitalWrite(IN\_13, HIGH);

digitalWrite(IN\_14, LOW);

analogWrite(ENB\_m1, speedCar);

digitalWrite(IN\_21, HIGH);

digitalWrite(IN\_22, LOW);

analogWrite(ENA\_m2, speedCar/speed\_Coeff);

digitalWrite(IN\_23, LOW);

digitalWrite(IN\_24, HIGH);

analogWrite(ENB\_m2, speedCar/speed\_Coeff);

}

void stopRobot(){

digitalWrite(IN\_11, LOW);

digitalWrite(IN\_12, LOW);

analogWrite(ENA\_m1, speedCar);

digitalWrite(IN\_13, LOW);

digitalWrite(IN\_14, LOW);

analogWrite(ENB\_m1, speedCar);

digitalWrite(IN\_21, LOW);

digitalWrite(IN\_22, LOW);

analogWrite(ENA\_m2, speedCar);

digitalWrite(IN\_23, LOW);

digitalWrite(IN\_24, LOW);

analogWrite(ENB\_m2, speedCar);

}

void loop(){

// Clears the trigPin

digitalWrite(trigPin, LOW);

delayMicroseconds(2);

// Sets the trigPin on HIGH state for 10 micro seconds

digitalWrite(trigPin, HIGH);

delayMicroseconds(10);

digitalWrite(trigPin, LOW);

// Reads the echoPin, returns the sound wave travel time in microseconds

duration = pulseIn(echoPin, HIGH);

// Calculating the distance

distance= duration\*0.034/2;

safetyDistance = distance;

if (Serial.available() > 0) {

command = Serial.read();

stopRobot(); //Initialize with motors stopped.

if (lightFront) {digitalWrite(light\_FR, HIGH); digitalWrite(light\_FL, HIGH);}

if (!lightFront) {digitalWrite(light\_FR, LOW); digitalWrite(light\_FL, LOW);}

if (lightBack) {digitalWrite(light\_BR, HIGH); digitalWrite(light\_BR, HIGH);}

if (!lightBack) {digitalWrite(light\_BR, LOW); digitalWrite(light\_BR, LOW);}

if (horn) {digitalWrite(horn\_Buzz, HIGH);}

if (!horn) {digitalWrite(horn\_Buzz, LOW);}

if (safetyDistance <= 20){

digitalWrite(light\_BR, HIGH);

digitalWrite(horn\_Buzz, HIGH);

delay(500);

goBackM();

delay(750);

}

else{

switch (command) {

case 'F':goAhead();break;

case 'B':goBack();break;

case 'L':goLeft();break;

case 'R':goRight();break;

case 'I':goAheadRight();break;

case 'G':goAheadLeft();break;

case 'J':goBackRight();break;

case 'H':goBackLeft();break;

case '0':speedCar = 100;break;

case '1':speedCar = 115;break;

case '2':speedCar = 130;break;

case '3':speedCar = 145;break;

case '4':speedCar = 160;break;

case '5':speedCar = 175;break;

case '6':speedCar = 190;break;

case '7':speedCar = 205;break;

case '8':speedCar = 220;break;

case '9':speedCar = 235;break;

case 'q':speedCar = 255;break;

case 'W':lightFront = true;break;

case 'w':lightFront = false;break;

case 'U':lightBack = true;break;

case 'u':lightBack = false;break;

case 'V':horn = true;break;

case 'v':horn = false;break;

}

}

}

}