**#include <Servo.h>**

**Servo myservo; //Define an instance of a servo**

**/\*\*JM1Motor controls pins\*\*/**

**#define PWM1 3**

**#define M1 2**

**/\*\*JM4Motor controls pins\*\*/**

**#define PWM4 11**

**#define M4 8**

**/\*\*JM2Motor controls pins\*\*/**

**#define PWM2 5**

**#define M2 4**

**/\*\*JM3Motor controls pins\*\*/**

**#define PWM3 6**

**#define M3 7**

**/\*\*Ultrasonic Sensor interface\*\*/**

**#define EchoPin 13 //ECHO to D13**

**#define TrigPin 12 //TRIG to D12**

**uint8\_t speed1 = 100; //Right front motor speed**

**uint8\_t speed4 = 100; //Right rear motor speed**

**uint8\_t speed2 = 100; //Left front motor speed**

**uint8\_t speed3 = 100; //Left rear motor speed**

**float distance\_M, distance\_L, distance\_R;**

**void setup() {**

**Serial.begin(9600); //Set baud rate to 9600**

**myservo.attach(9); //The pins of the servo are connected to D9**

**/\*\*All motor control pins are set to output mode\*\*/**

**pinMode(M1, OUTPUT);**

**pinMode(PWM1, OUTPUT);**

**pinMode(M2, OUTPUT);**

**pinMode(PWM2, OUTPUT);**

**pinMode(M3, OUTPUT);**

**pinMode(PWM3, OUTPUT);**

**pinMode(M4, OUTPUT);**

**pinMode(PWM4, OUTPUT);**

**pinMode(EchoPin, INPUT); // input mode set**

**pinMode(TrigPin, OUTPUT); // output mode set**

**myservo.write(90); //Rotate to 90 degrees**

**delay(100);**

**}**

**void loop() {**

**distance\_M = Get\_Distance(); //Get the distance and save in the distance variable**

**if (distance\_M < 20) { //When the distance in front is less than 20cm**

**Stop(); //Robot stop**

**delay(500); //Delay 500ms**

**myservo.write(180); //Ultrasonic head turns left**

**delay(500); //Delay 500ms**

**distance\_L = Get\_Distance(); //Assign the left ultrasonic distance to variable a1**

**delay(100); //Read values when stable**

**myservo.write(0); //ltrasonic cradle head turns right**

**delay(500); //Delay 500ms**

**distance\_R = Get\_Distance(); //Assign the right ultrasonic distance to variable a2**

**delay(100); //Read values when stable**

**myservo.write(90); //Return to the 90 degree position**

**delay(500);**

**if (distance\_L > distance\_R) { //When the distance on the left is greater than right**

**Turn\_Left(); //Robot turns left**

**delay(300); //Turn left 700 ms**

**} else {**

**Turn\_Right(); //Robot turns right**

**delay(300);**

**}**

**}**

**else { //If the distance in front is >=20cm, the robot will advance**

**Advance(); //advance**

**}**

**}**

**float Get\_Distance(void) { //Ultrasonic detects the distance**

**float dis;**

**digitalWrite(TrigPin, LOW);**

**delayMicroseconds(2);**

**digitalWrite(TrigPin, HIGH); //Give the TRIG a high level of at least 10 (µs) to trigger**

**delayMicroseconds(10);**

**digitalWrite(TrigPin, LOW);**

**dis = pulseIn(EchoPin, HIGH) / 58.2; //Work out the distance**

**delay(50);**

**return dis;**

**}**

**/\*\*The car advance \*\*/**

**void Advance(void) {**

**digitalWrite(M1, LOW);**

**analogWrite(PWM1, speed1);**

**digitalWrite(M4, LOW);**

**analogWrite(PWM4, speed4);**

**digitalWrite(M2, LOW);**

**analogWrite(PWM2, speed2);**

**digitalWrite(M3, LOW);**

**analogWrite(PWM3, speed3);**

**}**

**/\*\*The car moves back\*\*/**

**void Back(void) {**

**digitalWrite(M1, HIGH);**

**analogWrite(PWM1, 255 - speed1);**

**digitalWrite(M4, HIGH);**

**analogWrite(PWM4, 255 - speed4);**

**digitalWrite(M2, HIGH);**

**analogWrite(PWM2, 255 - speed2);**

**digitalWrite(M3, HIGH);**

**analogWrite(PWM3, 255 - speed3);**

**}**

**/\*\*The car rotates left\*\*/**

**void Turn\_Left(void) {**

**digitalWrite(M1, LOW);**

**analogWrite(PWM1, speed1);**

**digitalWrite(M4, LOW);**

**analogWrite(PWM4, speed4);**

**digitalWrite(M2, HIGH);**

**analogWrite(PWM2, 255 - speed2);**

**digitalWrite(M3, HIGH);**

**analogWrite(PWM3, 255 - speed3);**

**}**

**/\*\*The car rotates right\*\*/**

**void Turn\_Right(void) {**

**digitalWrite(M1, HIGH);**

**analogWrite(PWM1, 255 - speed1);**

**digitalWrite(M4, HIGH);**

**analogWrite(PWM4, 255 - speed4);**

**digitalWrite(M2, LOW);**

**analogWrite(PWM2, speed2);**

**digitalWrite(M3, LOW);**

**analogWrite(PWM3, speed3);**

**}**

**/\*\*The car stops\*\*/**

**void Stop(void) {**

**digitalWrite(M1, LOW);**

**analogWrite(PWM1, 0);**

**digitalWrite(M2, LOW);**

**analogWrite(PWM2, 0);**

**digitalWrite(M3, LOW);**

**analogWrite(PWM3, 0);**

**digitalWrite(M4, LOW);**

**analogWrite(PWM4, 0);**

**}**