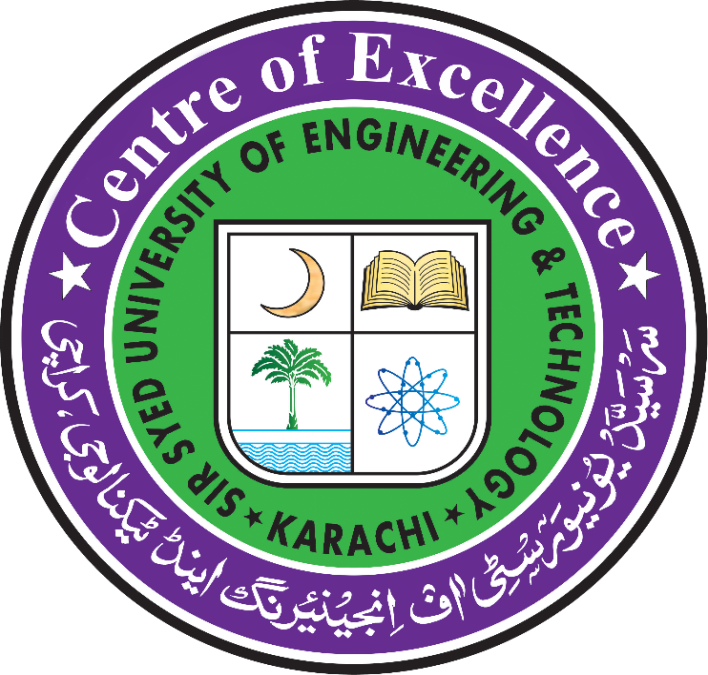
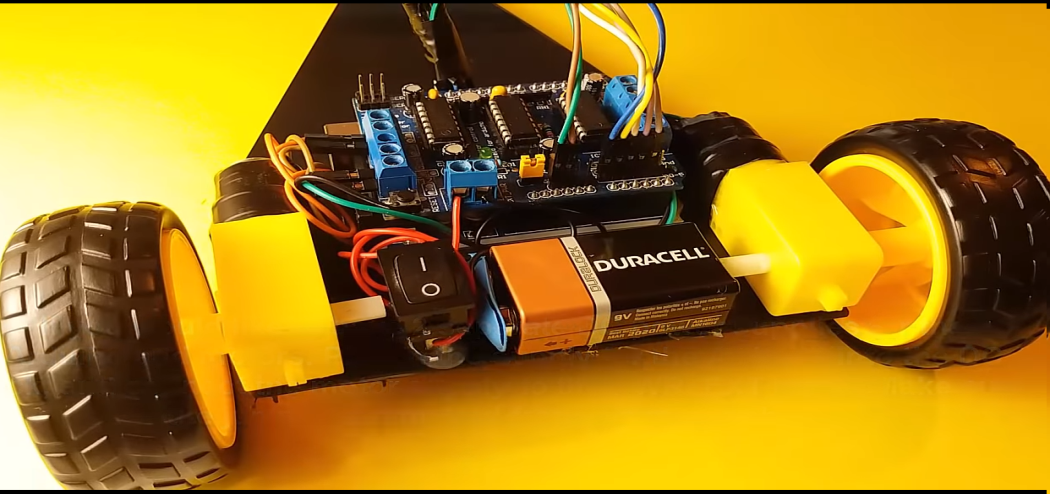
**PROJECT REPORT OF**

**LINEAR CONTROL SYSTEM**



**Project Title:**

**LINE FOLLOWING ROBOT**

****

**Group Members: Roll No.:**

1. M.Ali Shaikh 2017-TE-027
2. M.Arsalan Mirza 2017-TE-028
3. Abdul Qayooum Qureshi 2017-TE-036

**SUBMITTED TO: MISS MARIA ASHRAF**

**Line Following Robot**

A **Line Following Robot** is toy car type robot which follows a particular line (Black or White Line) it is Autonomous Robot which is consist of Micro-Controller and Pair of [IR](http://www.studentsheart.com/photodiode-working-applications/)**(Infrared) Sensors** and DC Gear Motors.

We will put Some Instructions into an [Arduino](http://www.studentsheart.com/arduino-learn-arduino/)( Micro-Controller) and that instruction will decide whether a Robot have to run in the forward direction or in the left direction or right direction. So I hope now you have an idea of Line Following Robot.

Usually, the visual line is the path in which the line follower robot goes and it will be a black line on a white surface but the other way (white line on a black surface) is also possible. Certain advanced Line Follower Robots use invisible magnetic field as their paths.

Large line follower robots are usually used in industries for assisting the automated production process. They are also used in military applications, human assistance purpose, delivery services etc.

Line follower Robot is one of the first robots that beginners and students would get their first robotic experience with. In this project, we have designed a simple Line Follower Robot using Arduino and some other components.

**Working**

1) DC Motors connect to the Motor Shield

2) Left Motors connect to M1

3) Right Motors connect to M2

4) The QTR-8RC Reflectance Sensor Connections

The Sensor VCC connect to the Arduino +5V

The Sensor GND connect to the Arduino Ground

The Sensor pin 2 connect to the Arduino Analog 0

The Sensor pin 3 connect to the Arduino Analog 1

The Sensor pin 4 connect to the Arduino Analog 2

The Sensor pin 5 connect to the Arduino Analog 3

The Sensor pin 6 connect to the Arduino Analog 4

5) Battery and on / off button connection

The (+) from the battery is connect to a leg of the button

Connect a cable to the other legs of the button. This cable is for Motor Shield (+)

6) Attach wheels

Copy the Source Code and install the Library:

1) Open the new Arduino IDE page and Delete everything on the page

2) Get the code from my web site and Paste empty Arduino IDE page

3) First Install the Arduino Libraries

3.1) Install the AF\_Motor Arduino library

Close the Arduino IDE

Download the library;

https://github.com/adafruit/Adafruit-Motor-Shield-library

Uncompress the ZIP file onto your desktop

Place the AFMotor folder into your arduinosketchfolder/libraries folder

Rename the uncompressed folder AFMotor

3.2) Install the QTRSensors library

Download the library;

https://github.com/pololu/qtr-sensors-arduino/releases

Rename the folder “qtr-sensors-arduino-xxxx” to “QTRSensors”

Drag the “QTRSensors” folder into the “libraries” directory inside your Arduino sketchbook directory

4) After installing the library, restart the Arduino IDE

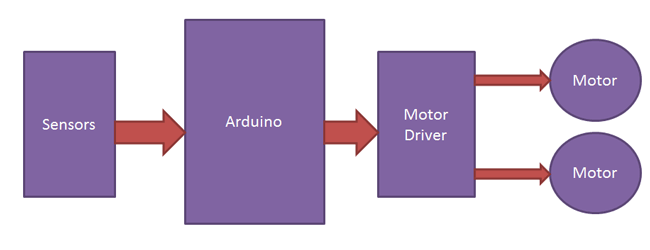
5) Now you can see the libraries you have installed

6) When you verify, you will not see any errors

**Components used in our Project**

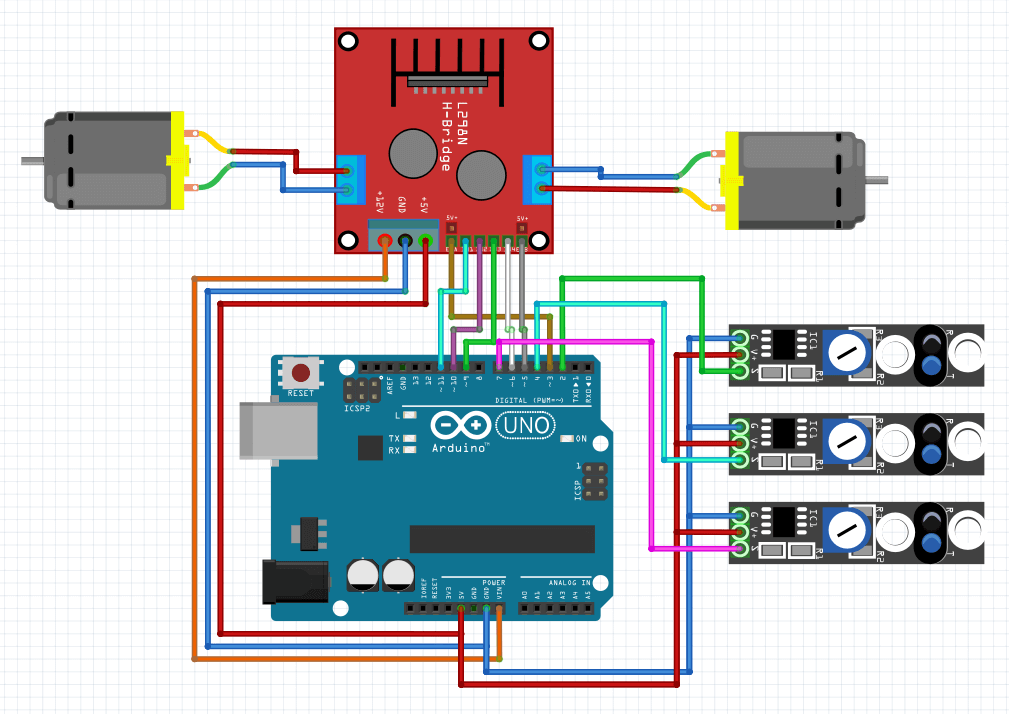
1. Arduino Board
2. Motor Shield L293D
3. QTR-8RC Reflectance Sensor
4. DC Motor and wheel x2
5. Battery Buckle
6. Jumper Wires
7. Bovine Wheel
8. Chassis
9. Battery 9V
10. Button
11. Glue Gun

**Block diagram of Line Following Robot**

****

Here we have the block diagram, of Line Following Robot, where we are using one QTR-8RC Reflectance Sensor and arduino, Motor Shield L293D and two simple DC motors and 9V battery. Here we have the complete block diagram of Line Following Robot, we have black line path where our Line Following Robot will run and it will only detect black line and complete black line path in just 60 seconds.

**Circuit diagram of Line Following Robot**



Its a circuit diagram of Line following Robot circuit where you can see and read the connections easily and point out the pin numbers also.

**Coding**

#include <AFMotor.h> //Adafruit Motor Shield Library. First you must download and install AFMotor library

#include <QTRSensors.h> //Pololu QTR Sensor Library. First you must download and install QTRSensors library

//Mert Arduino https://bit.ly/MertArduino

AF\_DCMotor motor1(1, MOTOR12\_1KHZ ); //create motor #1 using M1 output on Motor Drive Shield, set to 1kHz PWM frequency

AF\_DCMotor motor2(2, MOTOR12\_1KHZ ); //create motor #2 using M2 output on Motor Drive Shield, set to 1kHz PWM frequency

#define KP 2 //experiment to determine this, start by something small that just makes your bot follow the line at a slow speed

#define KD 5 //experiment to determine this, slowly increase the speeds and adjust this value. ( Note: Kp < Kd)

#define M1\_minumum\_speed 150 //minimum speed of the Motor1

#define M2\_minumum\_speed 150 //minimum speed of the Motor2

#define M1\_maksimum\_speed 250 //max. speed of the Motor1

#define M2\_maksimum\_speed 250 //max. speed of the Motor2

#define MIDDLE\_SENSOR 4 //number of middle sensor used

#define NUM\_SENSORS 5 //number of sensors used

#define TIMEOUT 2500 //waits for 2500 us for sensor outputs to go low

#define EMITTER\_PIN 2 //emitterPin is the Arduino digital pin that controls whether the IR LEDs are on or off. Emitter is controlled by digital pin 2

#define DEBUG 0

//Mert Arduino https://bit.ly/MertArduino

//sensors 0 through 5 are connected to analog inputs 0 through 5, respectively

QTRSensorsRC qtrrc((unsigned char[]) { A4,A3,A2,A1,A0} ,NUM\_SENSORS, TIMEOUT, EMITTER\_PIN);

unsigned int sensorValues[NUM\_SENSORS];

void setup()

{ //Mert Arduino https://bit.ly/MertArduino

delay(1500);

manual\_calibration();

set\_motors(0,0);

}

int lastError = 0;

int last\_proportional = 0;

int integral = 0;

void loop()

{ //Mert Arduino https://bit.ly/MertArduino

unsigned int sensors[5];

int position = qtrrc.readLine(sensors); //get calibrated readings along with the line position, refer to the QTR Sensors Arduino Library for more details on line position.

int error = position - 2000;

int motorSpeed = KP \* error + KD \* (error - lastError);

lastError = error;

int leftMotorSpeed = M1\_minumum\_speed + motorSpeed;

int rightMotorSpeed = M2\_minumum\_speed - motorSpeed;

// set motor speeds using the two motor speed variables above

set\_motors(leftMotorSpeed, rightMotorSpeed);

}

void set\_motors(int motor1speed, int motor2speed)

{ //Mert Arduino https://bit.ly/MertArduino

if (motor1speed > M1\_maksimum\_speed ) motor1speed = M1\_maksimum\_speed;

if (motor2speed > M2\_maksimum\_speed ) motor2speed = M2\_maksimum\_speed;

if (motor1speed < 0) motor1speed = 0;

if (motor2speed < 0) motor2speed = 0;

motor1.setSpeed(motor1speed);

motor2.setSpeed(motor2speed);

motor1.run(FORWARD);

motor2.run(FORWARD);

}

//calibrate for sometime by sliding the sensors across the line, or you may use auto-calibration instead

void manual\_calibration() {

int i;

for (i = 0; i < 250; i++)

{

qtrrc.calibrate(QTR\_EMITTERS\_ON);

delay(20);

}

if (DEBUG) { //Mert Arduino https://bit.ly/MertArduino

Serial.begin(9600);

for (int i = 0; i < NUM\_SENSORS; i++)

{

Serial.print(qtrrc.calibratedMinimumOn[i]);

Serial.print(' ');

}

Serial.println();

for (int i = 0; i < NUM\_SENSORS; i++)

{

Serial.print(qtrrc.calibratedMaximumOn[i]);

Serial.print(' ');

}

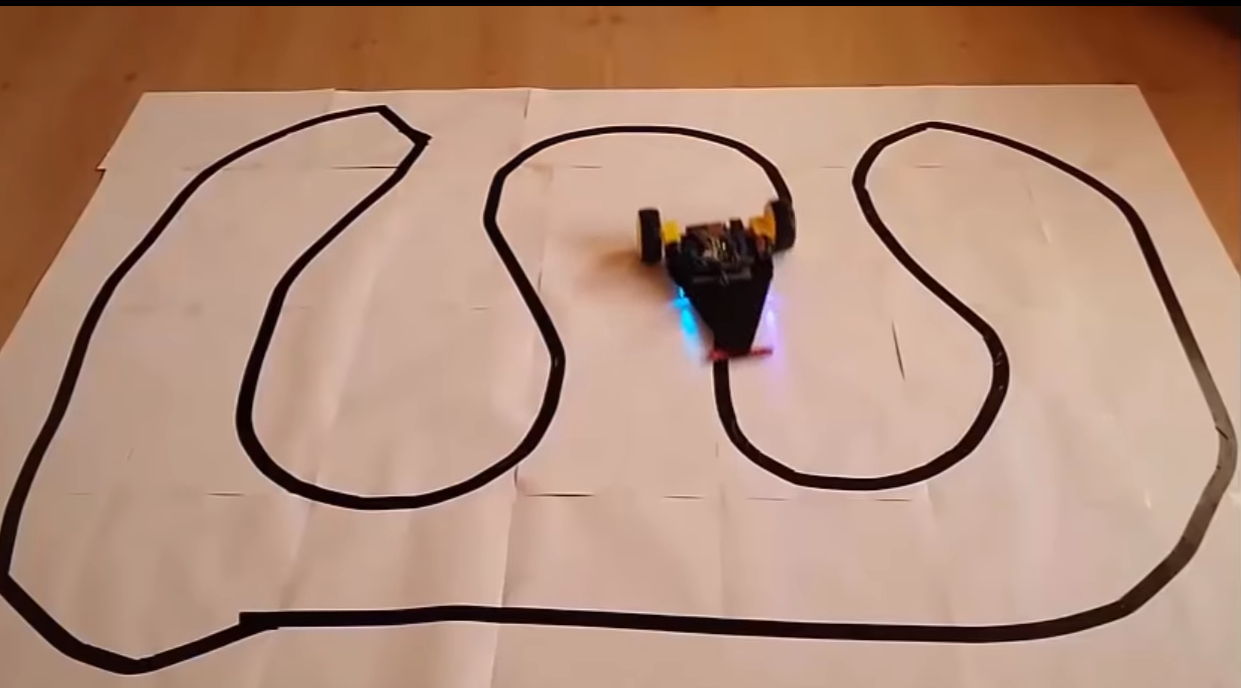
Serial.println();

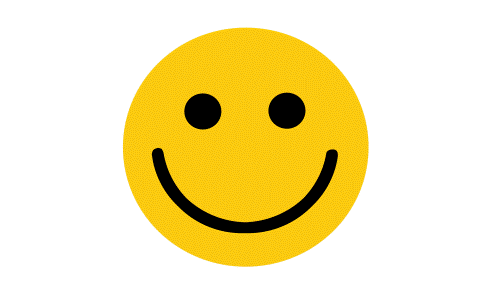
Serial.println();

//Mert Arduino https://bit.ly/MertArduino

}}

**Circuit prototype**

****

THANK YOU