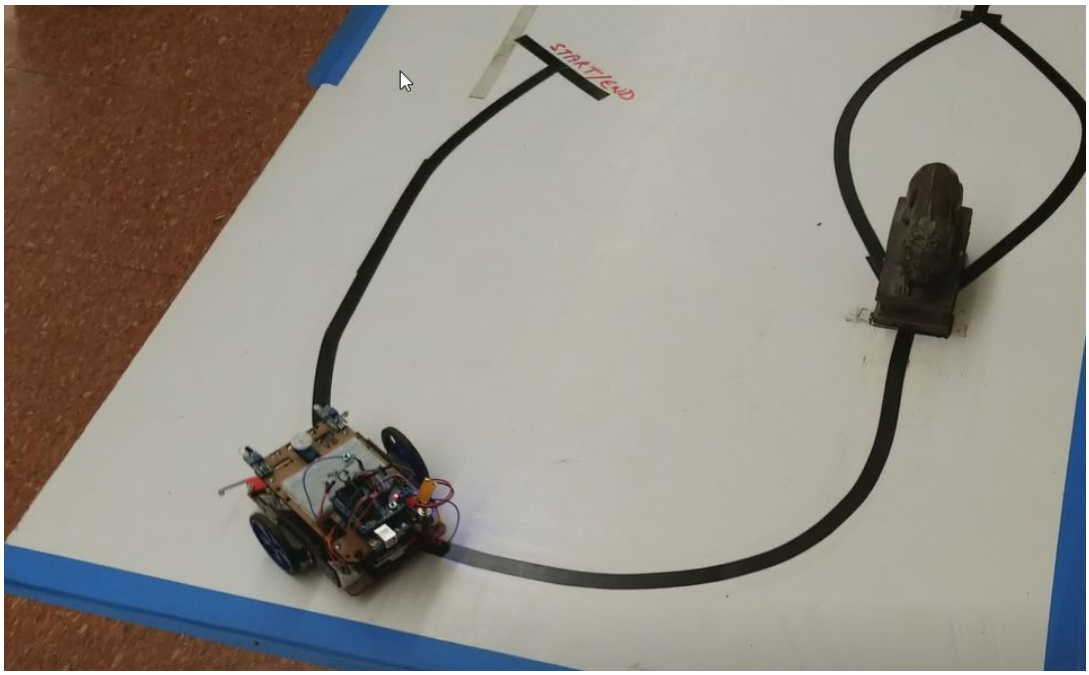
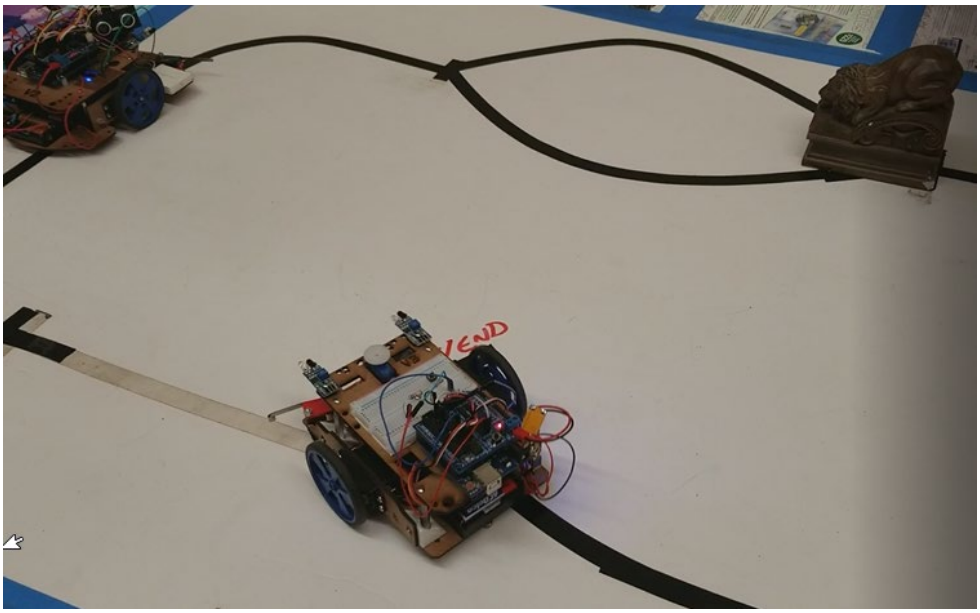
Experiment 5: IR Line Following with a "Twist"

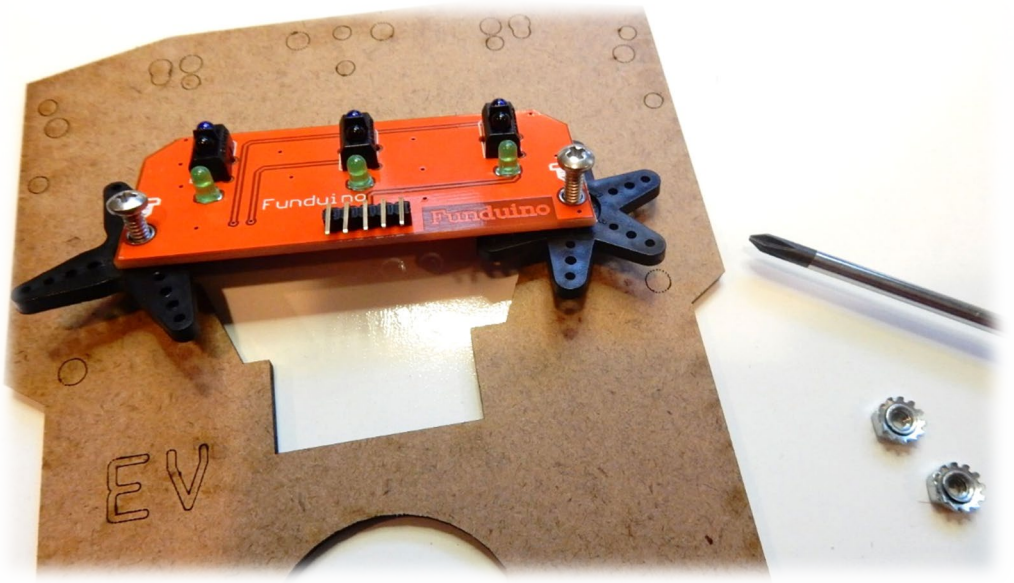
Purpose: Install the IR line following sensor and have the robot follow a black line on a white surface. Typically the black line is made from 1" wide black electrical tape. The twist is that there will be an obstacle placed randomly on the line and the robot must go around it. You are given code to follow the line and must figure the algorithm and pick a sensor/s based on your experience that will work the best to go around the randomly place object. The robot will start at a specific place and must return to the starting place once it has gone around the loop and avoided the obstacle.

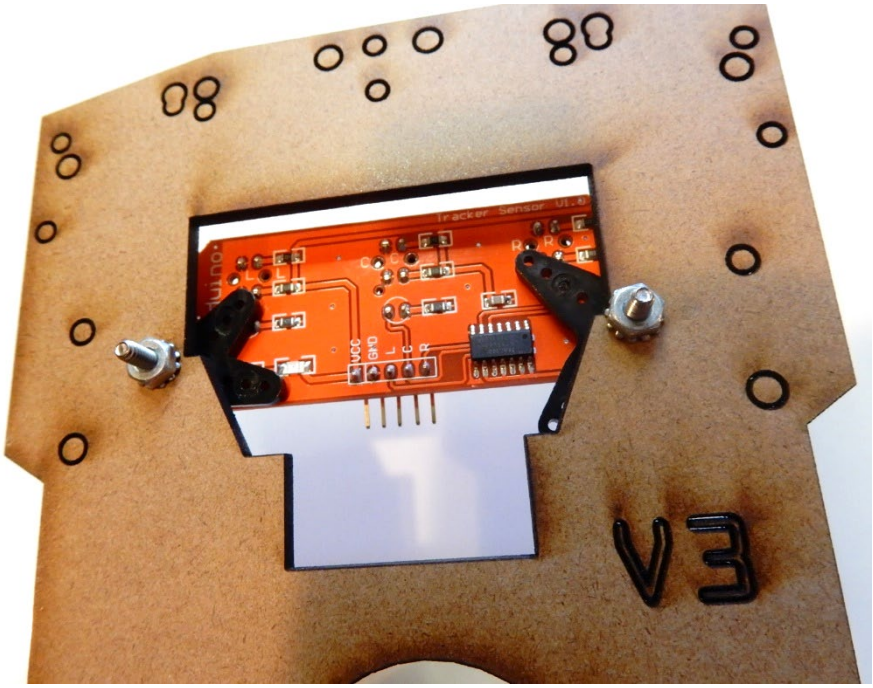


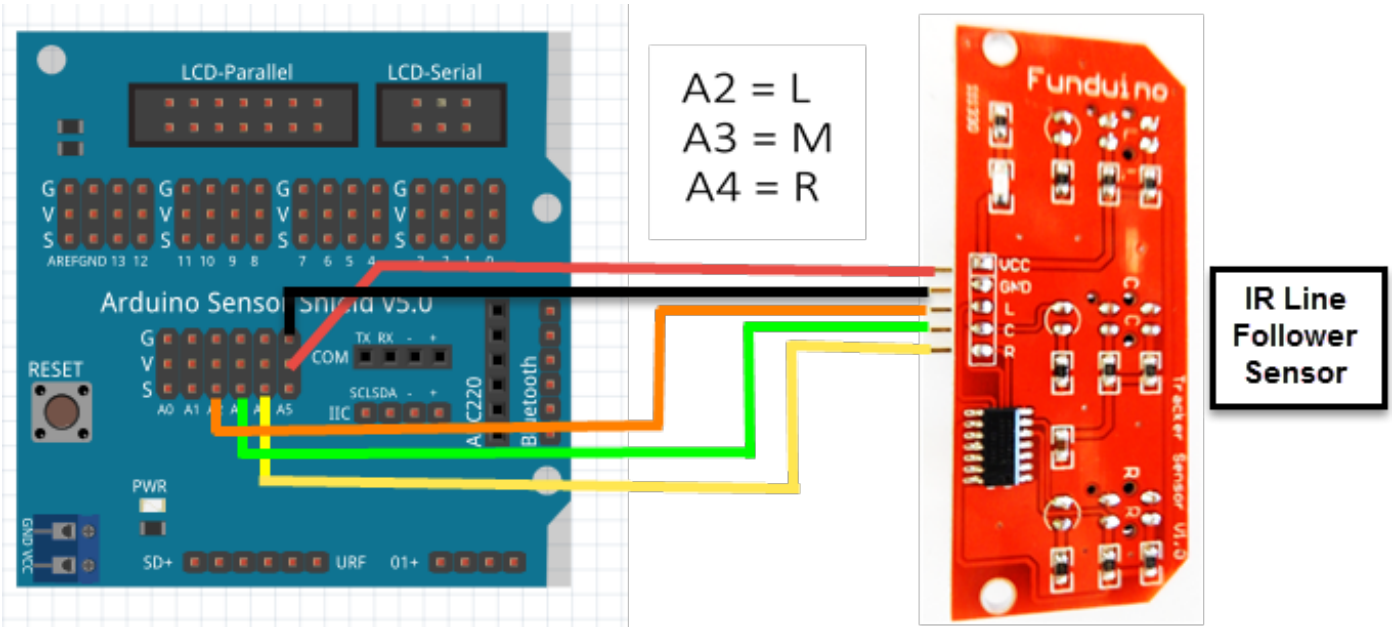


Line following with a twist. Using a line following sensor follow a black line on a white surface floor and have the robot return to its starting point. The twist is that there will be an obstacle on the path so your robot must be able to go around the obstacle and continue going. The robot that can perform this task the fastest wins this competition. You can use IR , Feelers or sonar to avoid the object and go around it. Video showing an example of what it will look like: https://www.youtube.com/watch?v=zod5qlQf3xk

Make sure your line following sensor has been installed as shown below.







This example below will provide you with a signal from all three TCRT5000 IR sensors found on the line following sensor. Use the Serial Monitor to test the sensors and write down the values you get when they see a white area or black area. You will use these numbers in the line following program and you can reuse this program to calibrate the correct values at the time of the competition.

```
// Testing the three IR sensors found in the line following sensor.
void setup()
{
Serial.begin(9600);              // baud Rate :9600
Serial.print("Sensor value");
}
 void loop()
{
Serial.println(analogRead(A2));     // Reading analog value of sensor and printing on serial monitor
Serial.println(analogRead(A3));
Serial.println(analogRead(A4));
delay(1000);    }
```

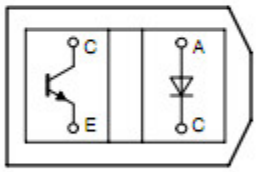
Another Example created by modifying the Arduino AnalogInput sensor:

```
// Reads the Funduino line following sensors - you can add the third sensor for your practice

int sensorPin = A2;    // select the input pin for the potentiometer
int sensorValue = 0;  // variable to store the value coming from the sensor
int sensorPin3 = A3;    // select the input pin for the potentiometer
int sensorValue3 = 0;  // variable to store the value coming from the sensor

void setup() {
Serial.begin(9600);
}

void loop() {
  // read the value from the sensor:
  sensorValue = analogRead(sensorPin);
  sensorValue3 = analogRead(sensorPin3);
  Serial.print("A2=");
 Serial.println(sensorValue);
  delay(1000);
    Serial.print("A3=");
 Serial.println(sensorValue3);
   delay(1000);
}
```

Technical specs about the sensor:

Reflective Optical Sensor with Transistor Output

19156_2



Top view 19156_1

FEATURES

- Package type: leaded
- Detector type: phototransistor
- Dimensions (L x W x H in mm): 10.2 x 5.8 x 7
- Peak operating distance: 2.5 mm
- Operating range within > 20 % relative collector current: 0.2 mm to 15 mm
- Typical output current under test: I_C = 1 mA
- Daylight blocking filter
- Emitter wavelength: 950 nm
- Lead (Pb)-free soldering released
- Compliant to RoHS directive 2002/95/EC and in accordance to WEEE 2002/96/EC

Pb-free

e2

RoHS COMPLIANT

DESCRIPTION

The TCRT5000 and TCRT5000L are reflective sensors which include an infrared emitter and phototransistor in a leaded package which blocks visible light. The package includes two mounting clips. TCRT5000L is the long lead version.

APPLICATIONS

- Position sensor for shaft encoder
- Detection of reflective material such as paper, IBM cards, magnetic tapes etc.
- Limit switch for mechanical motions in VCR
- General purpose - wherever the space is limited

Sample code to get you started on the line following sensor program. You can use one, two or all three IR sensors of the line following sensor. It is up to you and your algorithm.

```
/*
This line following demo uses three of the analog reflective sensors on the Funduino line following sensor to follow a line.
*/

#include <Servo.h>

Servo servoLeft; // Define left servo
Servo servoRight; // Define right servo

const int lineLSense = A2;
const int lineMSense =A3;
const int lineRSense = A4;

int irReflectL = 0;
int irReflectM = 0;
int irReflectR = 0;

int thresh = 650;
```

```
void setup() {
  servoLeft.attach(11);
  servoRight.attach(10);
}


void loop() {
// Read reflective sensors
  irReflectL = analogRead(lineLSense);
  irReflectM = analogRead(lineMSense);
  irReflectR = analogRead(lineRSense);

  if (irReflectL <= thresh && irReflectM <= thresh && irReflectR <= thresh) {
    line_stop();
  }
  if (irReflectL <= thresh && irReflectM <= thresh && irReflectR >= thresh) {
    line_spinRight();
  }
  if (irReflectL <= thresh && irReflectM >= thresh && irReflectR <= thresh) {
    line_forward();
  }
  if (irReflectL <= thresh && irReflectM >= thresh && irReflectR >= thresh) {
    line_slipRight();
  }
  if (irReflectL >= thresh && irReflectM <= thresh && irReflectR <= thresh) {
    line_spinLeft();
  }
  if (irReflectL >= thresh && irReflectM <= thresh && irReflectR >= thresh) {
    line_forward();
  }
  if (irReflectL >= thresh && irReflectM >= thresh && irReflectR <= thresh) {
    line_slipLeft();
  }
  if (irReflectL >= thresh && irReflectM >= thresh && irReflectR >= thresh) {
    line_forward();
  }
}


// Motion routines for line following
void line_forward() {
  servoLeft.write(180);
  servoRight.write(0);
}

void line_slipRight() {
```

```
  servoLeft.write(180);
  servoRight.write(90);
}

void line_slipLeft() {
  servoLeft.write(90);
  servoRight.write(0);
}

void line_spinRight() {
  servoLeft.write(180);
  servoRight.write(180);
}

void line_spinLeft() {
  servoLeft.write(0);
  servoRight.write(0);
}

void line_stop() {
  servoLeft.write(90);
  servoRight.write(90);
}
```

You already know how to use infrared, feelers and sonar sensors for navigation. You should be able to incorporate what you learned before for the obstacle avoidance part of this competition.

Obstacle is to be smaller or similar size as your robot.

Let's get ready to compete now! (we usually have a very competitive competition!)

You should also try using a single sensor for line following. You might find that it works better for you. Here is one I did with a single sensor.

```
/*
This line following demo uses one of the analog reflective sensors on the Funduino line following sensor to follow a line.
*/

#include <Servo.h>

Servo servoLeft; // Define left servo
Servo servoRight; // Define right servo



//const int lineLSense = A2;
  const int lineMSense = A0;
//const int lineRSense = A4;
```

```
//int irReflectL = 0;
int irReflectM = 0;
//int irReflectR = 0;

int thresh = 900;


void setup() {
  servoLeft.attach(11);
  servoRight.attach(10);
}


void loop() {
// Read reflective sensors
//  irReflectL = analogRead(lineLSense);
  irReflectM = analogRead(lineMSense);
 // irReflectR = analogRead(lineRSense);

line_forward();

  if (irReflectM <= thresh) {
    line_slipRight();
  }

  if (irReflectM >= thresh ) {
    line_slipLeft();
    delay (10);
  }

//  if (irReflectM <= thresh) {
  //  line_spinRight();
 // }

}


// Motion routines for line following
void line_forward() {
  servoLeft.write(180);
  servoRight.write(0);
}

void line_slipRight() {
  servoLeft.write(180);
  servoRight.write(90);
```

```
}

void line_slipLeft() {
 servoLeft.write(90);
 servoRight.write(0);
}

void line_spinRight() {
 servoLeft.write(180);
 servoRight.write(180);
}

void line_spinLeft() {
 servoLeft.write(0);
 servoRight.write(0);
}

void line_stop() {
 servoLeft.write(90);
 servoRight.write(90);
}
```

Demo and report: provide code and demo that the robot can follow a black line on a white surface, avoid hitting an obstacle as large or smaller than the robot and that it continues following the line after it avoids the object. If using touch sensors such as the feelers then is fine if the robot touches the obstacle.

End of Experiment

Solution with single sensor

```
#include <Servo.h>

Servo servoLeft; // Define left servo
Servo servoRight; // Define right servo
const int lineMSense = A0;
int irReflectM = 0;
int thresh = 900;
void setup() {
  servoLeft.attach(11);
  servoRight.attach(10);
  pinMode(7, INPUT);
}

void loop() {
irReflectM = analogRead(lineMSense);
line_forward();

byte wRight = digitalRead(7);
//these values are dependent of the obstacle and varies by obstacle so is just
an idea
if(wRight == 0)
{
  line_backward();
  delay (500);
```

```
  line_stop ();
  delay (800);
  line_left();
  delay(750);
  line_stop ();
  delay (800);
   line_forward();
  delay(1000);
 line_stop ();
  delay (800);

  line_right();
  delay(1100);
  line_stop ();
  delay (800);
  line_forward();
  delay(3000);


 }
  if (irReflectM <= thresh) {
    line_slipRight();
  }

  if (irReflectM >= thresh ) {
    line_slipLeft();
    delay (10);
  }

//  if (irReflectM <= thresh) {
  //  line_spinRight();
 // }

}


// Motion routines for line following
void line_forward() {
  servoLeft.write(180);
  servoRight.write(0);
}

void line_forward2() {
  servoLeft.write(150);
  servoRight.write(60);
}

void line_left() {
//  servoLeft.write(90);
//  servoRight.write(120);
  servoLeft.writeMicroseconds(1700);
servoRight.writeMicroseconds(1700);
}

void line_right() {
 // servoLeft.write(120);
  //servoRight.write(90);
    servoLeft.writeMicroseconds(1300);
servoRight.writeMicroseconds(1300);

}

void line_backward() {
  servoLeft.write(0);
  servoRight.write(180);
}
void line_slipRight() {
  servoLeft.write(180);
  servoRight.write(90);
}

void line_slipLeft() {
  servoLeft.write(90);
```

```
  servoRight.write(0);
}

void line_spinRight() {
  servoLeft.write(180);
  servoRight.write(180);
}

void line_spinLeft() {
  servoLeft.write(0);
  servoRight.write(0);
}

void line_stop() {
  servoLeft.write(90);
  servoRight.write(90);
}
```