Robotics101

Sergio Azizi

September 23, 2016

Foreword

The objective of this project was to build a low-cost robot, which can (a) detect and avoid obstacles, and (b) follow a source of light.
I completed this project with the UCLU Robotics Society.
This document is just a brief, informal summary of the project
(and my first ever attempt at using latex...have mercy on me and the amateur formatting).

Mission

Build a robot that that will follow a light source while avoiding obstacles...

...complete

Journey

There are three parts to this:

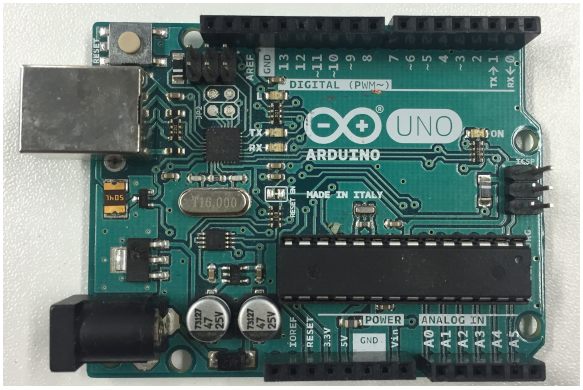
1. Assembly
2. Obstacle Avoidance
3. Light Follower

Overall, pretty simple.

1 Assembly

1.1 The Microcontroller

I used an **Arduino Uno**. Here is a quick overview:



Pin's The top row are the digital pins (used with digitalRead(), digitalWrite and analogWrite).
In the bottom right, we have the analog pins (used with analogWrite).
Next to it, (bottom central), we have GND and 5V pins, to connect arduino to ground and provide +5V power.

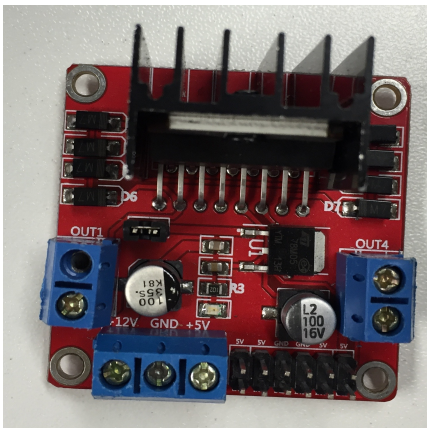
LED's LED 13, driven by pin 13.
Underneath are TX and RX LED's, indicating communication computer and Arduino.
To the right of the label, is the power pin (indicats if arduino receives power).

The actual microprocessor (an ATmega) is positioned underneath the label.
In the bottom left is the power connector where the batteries will be attached to.

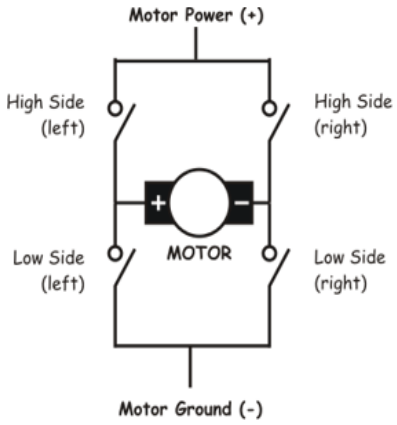
1.2 controls

The body of my "robot" is a wooden board with three wheels attached to it, two at the front and one in the back. The front wheels are powered via two brushless DC motors.

Each motor requires a current of 0.5 amps and a voltage of at least 6V. However, the **Arduino Uno** is only capable to providing 0.4 amps and 5 volts.
Solution: With a motor driver board and a switch that operates at the Arduino current and voltage level, I control the necessary amps and volts to change directions of the rotation. Operating at Arduino level means when you supply 5V it is "turned on", and when you supply 0V, it's "turned off". Here is the concept: The Driver is is arranged like an H-Bridge: When the switch is turned on, the circuit will close at high-left and low-right, causing the motor to rotate clockwise. Similarly, when the switch is turned off, the motor will turn in the anti-clockwise direction.



(a) Driver Board.



(b) h-Bridge.

We can then define a function like the one below and simply call left() whenever we want to do so.

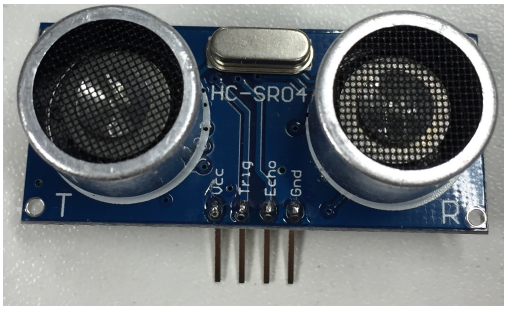
```
int lMotorPin1 = 6;
int lMotorPin2 = 9;
int rMotorPin1 = 3;
int rMotorPin2 = 5;

void setup() {
  pinMode(lMotorPin1, OUTPUT);
  pinMode(lMotorPin2, OUTPUT);
   pinMode(rMotorPin1, OUTPUT);
  pinMode(rMotorPin2, OUTPUT);
}
```

```
void left(){
    analogWrite(lMotorPin1, 255);
  analogWrite(lMotorPin2, 0);

   analogWrite(rMotorPin1, 0);
  analogWrite(rMotorPin2, 255);
}
```

2 Obstacle Avoidance



The robot uses a sonar to detect obstacles. The sonar consists of a transmitter that sends out ultrasonic waves and a receiver which detects the reflection of these waves.

```
int trigPin = 10;
int echoPin = 11;
void setup() {
  // put your setup code here, to run once:
  Serial.begin(9600);
  pinMode(trigPin, OUTPUT);
  pinMode(echoPin, INPUT);
}
void loop() {
  // put your main code here, to run repeatedly:
  digitalWrite(trigPin, LOW);
  delayMicroseconds(2);
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(2);
  digitalWrite(trigPin, LOW);
  delayMicroseconds(5);

  signed long duration = pulseIn(echoPin,HIGH);
  int distance = microsecondsToCentimeters(duration);
  Serial.print("Distance: ");
  Serial.println(distance);
  delay(100);
}
```

The setup part is to "set up" the communication between sonar and Arduino. In the Loop part, we tell the sonar to send out a low signal followed by a high signal followed by a low signal. Using the pulseIn function we get the time it took (in microseconds) for the receiver to detect the reflection if the high signal.

Finally, using the (self-defined) "microsecondsToCentimeters" function, we get the distanceat which the wave has been reflected.

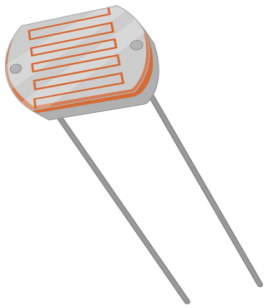
```
Complete
int microsecondsToCentimeters(signed long duration){
  return (int) (340.29*(duration)/10000/2);
}
```

This next part of the code are the commands in case it does come near an obsticacle. It basically

```
  if ((distance < 10) &&(distance != 0)) {
    backwards();
    delay(1000);
    turnright();
    delay(1000);
  }
```

says: "when you get within 10cm of an object, go back a little and change direction".

3 Light Follower



To detect changes in Light intensity, I used two light dependent resistors (LDR's).

```
int lefteye = A0;
int righteye = A1;

int tol = 100;

void setup() {
  // put your setup code here, to run once:
  pinMode(lefteye, INPUT);
  pinMode(righteye, INPUT);

  Serial.begin(115200);
}
void loop() {
  int lefteyeValue =analogRead(lefteye)-100;
  int righteyeValue =analogRead(righteye);

  Serial.print("left: ");
  Serial.println(lefteyeValue);

  Serial.print("right: ");
  Serial.println(righteyeValue);

  if (int(righteyeValue - lefteyeValue) > tol) {
    Serial.println("Turn Right");
    turnright();
  } else if (int(righteyeValue - lefteyeValue) < -tol) {
    Serial.println("Turn Left");
    turnleft();
  } else{
    Serial.println("go forward");
    goforward(255);
  }
  delay(300);
}
```

In simple terms, this is how an LDR works: Change in light intensity (arrow) change in resistance (arrow) change in voltage The Arduino has a builtin analog-to-digital converter (ADC), with a 10 bit resolution. This means at maximum light intensity, it will read the value 2pow(10)-1 (1023), and 0 when there is no light. Decreasing the light intensity, increases the resistance and according to Ohm's law, also the voltage. Therefore the side that reads reads a lower value will be the brighter one. A tolerance of 50 seemed to work fine in a dark room. In a brighter room, you'll have decrease the tolerance (unless you have a ridiculously strong flash light). Unfortunately, this made the robot more susceptible to unrelated sources of light.