Robotics101

Sergio Azizi

Abstract

I built this under the supervision of Chinemelu Ezeh, president of the UCLU Robotics Society, with he UCLU Robotics Soc. ...

A sensor detects changes in its environment (through through its physical properties, eg a thermistor changes its internal resistance with changes in temperature) and passes this information on to the microcontroller through changes in voltage. We can then program the microcontroller to interpret these voltages however we like (for example, when temperature becomes unusually high, light an LED to signal that something unusual is happening or trigger some other process).

1. Mission

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

2. Journey

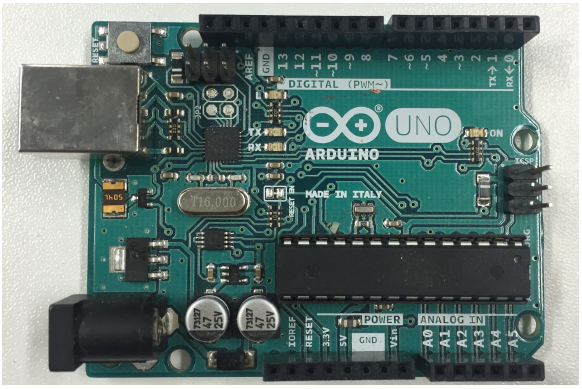
There are three parts to this:

1. Assembly and Controls
2. Obstacle Avoidance
3. Light Follower

Overall, pretty simple.

3. Microcontroller

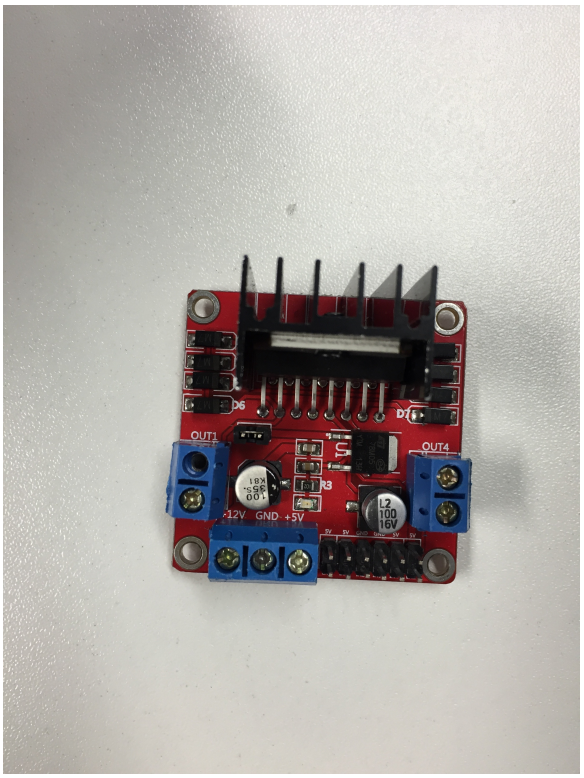
I used an Arduino Uno. Pins: The top row are the digital pins (used with digitalRead(), digitalWrite and analogWrite) In the bottom right, we have the analog pins (used with analogWrite). Next to it, (bottom central), we have GND and 5V pins, to connect arduino to ground and provide +5V power. LED: LED 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



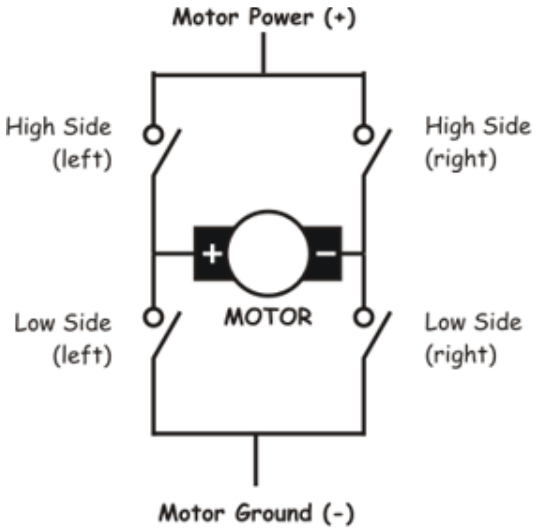
power). The microcontroller (an ATmega) is positioned underneath the label. In the bottom left is the power connector where the batteries will be attached to.

4. Building the thing

My "robot" is just a 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 circuit and a switch that operates at the Arduino current/voltage level (on means 5V, of means 0V), I control the necessary amps and volts to change directions of the rotation. Motor Driver: Basically, the Driver is is arranged like an H-Bridge. When the switch is turned on (supplying 5V to Arduino), pin 2



will close, causing the motor to rotate clockwise. Similarly, when the switch is turned off (by supplying 0V), pin 3 will close and the motor will rotate in the opposite direction. The configuration then looks something like this: Now all that's left to do is define some simple functions to make it move in a certain direction, eg.



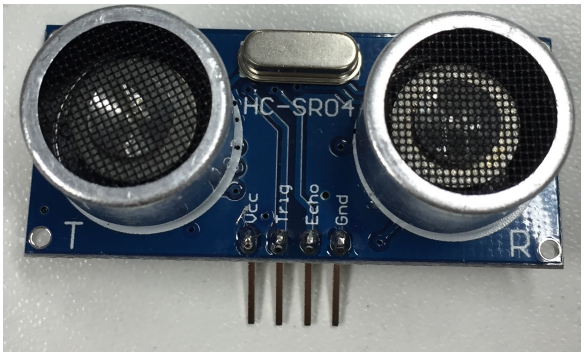
source: http://www.mcmanis.com/chuck/robotics/tutorial/h-bridge/images/basic-bridge.gif

```
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);
}
```

5. 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. The setup part is to ”set up” the communica-

```
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);
}
```

tion 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. This next part of the code is somewhat arbi-

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

trary. It basically says: ”when you get within 10cm of

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

an object, go back a little and change direction”.

6. Light Follower

/*Picture of LDR*/ To detect changes in Light intensity, I used two light dependent resistors (LDR’s). In simple terms: 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

```
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);
}
```

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.