Final Report

Eric Dong (ericd3) Sarah Hashash (hashash2)

December 2018

1 Introduction

1.1 The Problem

In our experiences with waterjet machines, the water tank doesn't automatically fill itself, leaving the user to have to manually refill the tank when the water becomes polluted. Our project provides a solution that automates this process of refilling and draining, using an ultrasonic sensor, pump, h-bridge, photoresistor, buttons, rgb LED, and laser module. The combination of the laser module and photoresistor act as water quality sensor, with the rgb led indicating the pollution status through a gradient of color. The rest of the components aid in sensing the water levels in the tank, signaling when the water must be drained or pumped to reach the desired water level.

1.2 Design Concept

As mentioned above our solution contains two main blocks of components that execute different roles in automation: water quality and water level sensing. In the latter block, the inputs include the ultrasonic sensor which continuously runs, collecting data that is printed out onto the serial monitor, and the buttons, which when one is pressed but not both will set the desired water level. The output of this block is the pump that will fill or drain the water until the desired water level if met. The button does not need to be pressed for the duration of the fill up and drain process.

In water quality block of the component, the data read from the photoresistor, an input, will trigger the pump, an output, to drain of the entire tank.. The data read from the photoresistor will also affect the color gradient of the rbg LED, an output. The rbg LED will range from green to yellow to red reflecting the lower to higher data values that will be printed on the serial monitor.

2 Analysis of Components

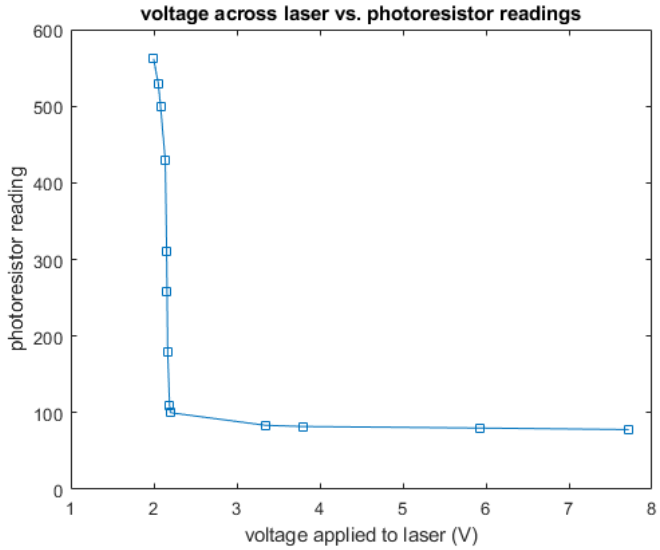
2.1 Characterization of each Sensor

- Ultrasonic sensor
  - This sensor determines the distance of the closest object that is in front of it and feeds this signal to the Arduino.
  - The graphs we sketched for the module 7D graph demonstrates how the ultrasonic sensor works. It first sends out a ultrasonic wave, and when it bounces back to the echo microphone, it creates a PWM signal to the echo pin.
- Photoresistor
  - This sensor determines how much light is hitting it and variates the resistance when a voltage is applied. We then measure the voltage drop across the photoresistor in series with another resistor and feed this analog signal to the arduino.

– To get the values for the graph, applied different voltages to the laser module which was aimed at the photoresistor and recorded the values that the arduino analogRead function outputted. This tells us how sensitive the photoresistor is and the range of it.

  * The graph below can be approximated to $y = 6.8814x^{-1.269}$

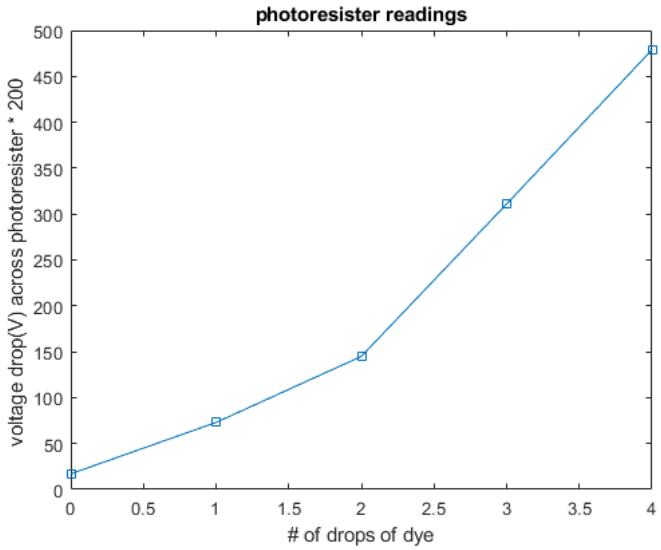
Figure 1: voltage plot and readings



– To get the values for the below graph, we varied the number of drops of food dye, aim the laser module at the photoresistor and recorded the values that the arduino analogRead function outputted. This tells us how sensitive the photoresistor is and the range of it.

  * The graph below can be approximated exponentially to $y = 23.977e^{0.8126x}$

Figure 2: drop plot and readings

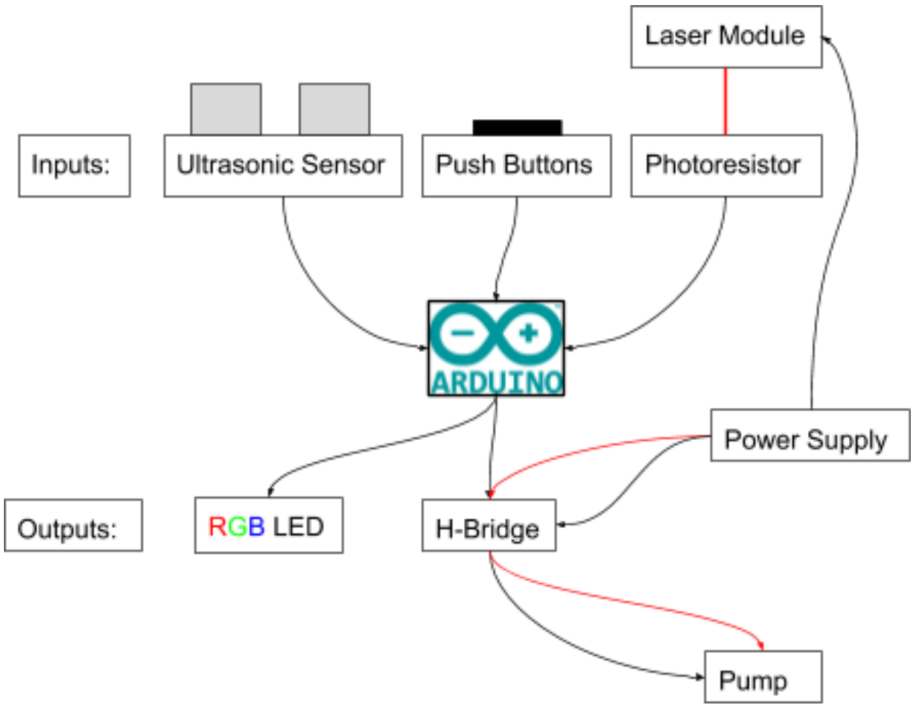


2.2 Design Considerations

- The ultrasonic sensor emits ultrasound at angle. Considering this property of the ultrasound, we had to make sure that nothing else but the target, which is the water surface, is in the immediate range of the sensor. Components such as wires should not get in the way of the ultrasound, otherwise there would be weird values that will be in the data set.
- The ultrasonic sensor would once in a while send extraneous values to the Arduino, thus we had to prevent these from value being used. In the code there is a recursive function where if the value is extraneous, then the function calls itself again until the values were actually valuable.
- The photoresistor has to be placed where the transparent container has a flat and vertical surface so that the laser does not deflect off the surface of the container. The laser should also not be close to the water surface since any small waves in the cup may cause inaccurate readings.
- We were first planning to do the project with a water quality sensor, We figured that it was too expensive, so we decided to do it with a laser and photoresistor setup. This is based on Beer's law where some of the light/laser shooting through a solution would scatter or absorbed by the solute which in our case is the food coloring that we add to the water solution.

3 Design Description

3.1 Block Diagram



3.2 Circuit Schematics

There are three inputs to the arduino, the ultrasonic sensor, push buttons, and the photoresistor. The ultrasonic sensor receives a signal from the arduino through the trig pin to send out a ultrasonic wave, this wave bounces back from a nearby object and into the echo pin where an PWM signal is then created and fed

back to the arduino. The Arduino measures the pulse width and then calculates the duration of the wave, we can then use the duration to calculate the distance to the object. The push buttons are used as logical switches in our circuit, when the push button is pressed, the digital pin that's connected to the Arduino receives a HIGH signal, otherwise it's LOW. We had to make sure that the push button is grounded so that without it being pressed it returns a LOW. The final input is the photoresistor which is connected to the analog pin of the Arduino. We have created a voltage divider, where the photoresistor is connected in series to another resistor, we then connect 5V and ground to either end of the circuit, and analog pin to where the two resistors are connected. The Arduino's analog pin reads the voltage in reference to ground of the photoresistor's voltage drop, and we process the signal accordingly. Now the outputs. The RGB LED is controlled by the Arduino, and is a common anode type LED, meaning the common pin of the LED is connected to 3.3V through a resistor. From the processed signal of the photoresistor, we connected the pins of the RGB LED to three digital pins of the Arduino and so if the voltage drop across the photoresistor is low, then the green LED lights up and as the voltage drop increases, the red LED begins to light up, creating a gradient of colors. The final output is the H-Bridge, which controls the motor/pump. We searched online for the pinouts of the IC and connected wires accordingly such that 1A and 2A pins are connected to digital pins on the Arduino. These are inputs of the IC where if one pin is HIGH and the other is LOW, the motor starts spinning, and if the signals are swapped, the motor reverses. The inputs of the H-Bridge is controlled by a combination of the push buttons and ultrasonic sensor. The 1Y and 2Y pins on the IC is connected to the motors with no directional preference. Finally the power supply is connected to the laser module and the H-Bridge in order to power each component. The laser uses 6V and the H-bridge uses 12V to run the motor.

Figure 3: Circuit

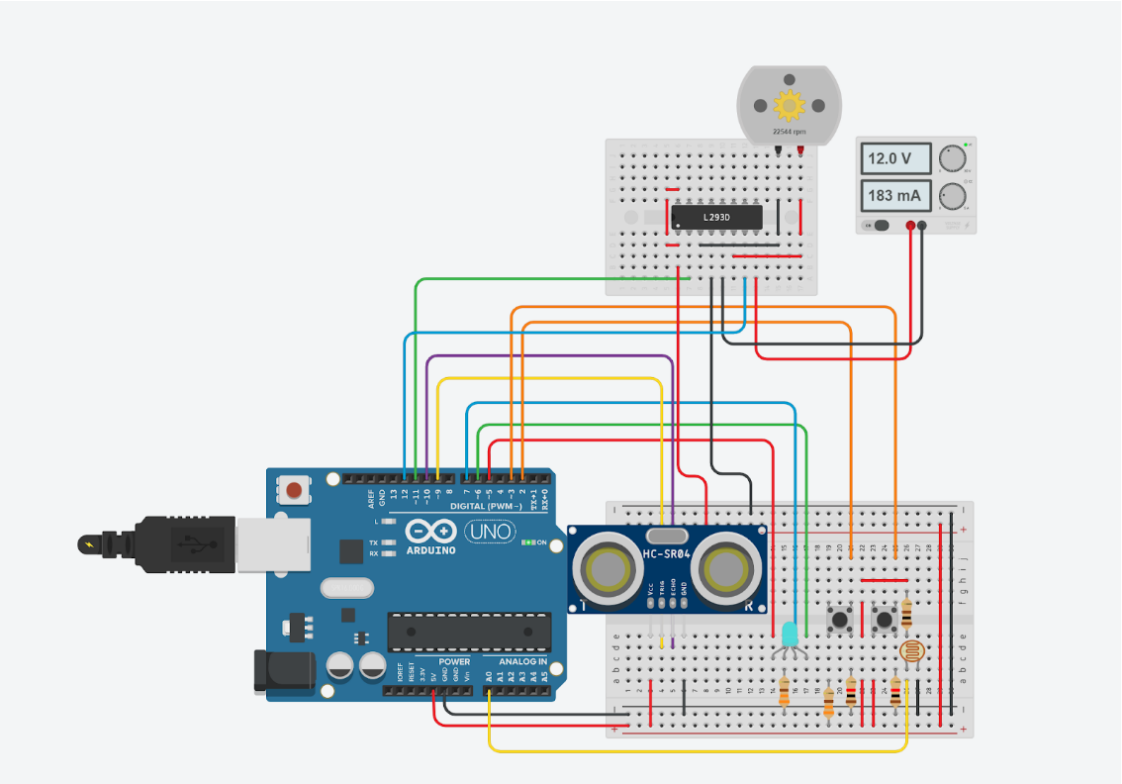
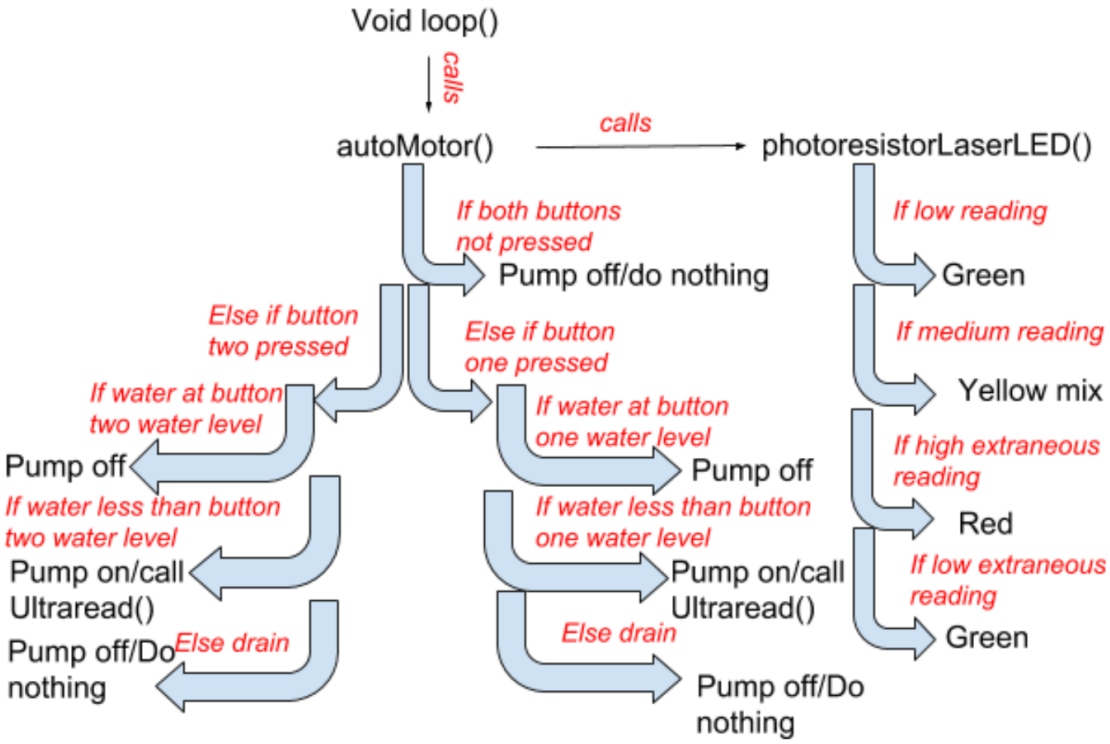


Figure 4: Code Diagram

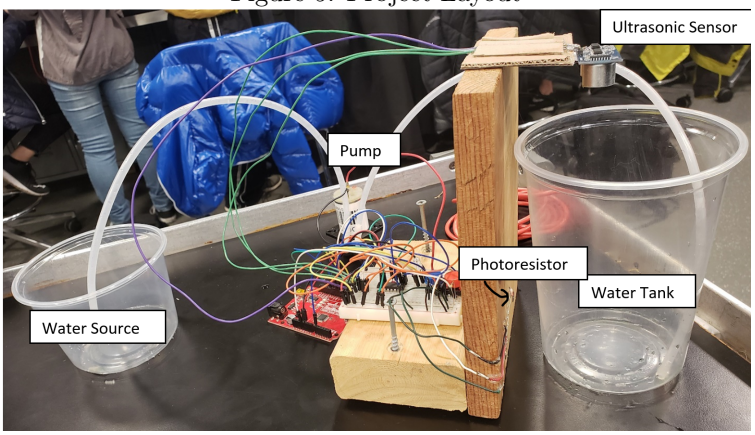


Note: Entire code is at the end of this report

3.3 Physical Construction

While there are no mechanical moving components of our product, we did have to mount various components. First, we had to mount the ultrasonic sensor with glue and cardboard above the tank to get accurate readings, then we had to place the laser module and photoresistor of optimally on the sides of the tank to minimize the effects of refraction. In addition the photoresistor had to be glued down as shifting during the process could make our pump never drain. Lastly we glued and secured the breadboard onto the wooden frame that was constructed with some scrap wood.

Figure 5: Project Layout



4 Conclusion

4.1 Lessons Learned

Surprisingly, one of the more unexpected obstacles we encountered was what we used as the tank for demonstration purposes. In the very beginning we wanted to find a tub with a physical drain that we could control with a servo. However, we could not find the tub in the hardware stores and therefore we used a clear plastic box the size of a shoe box for testing purposes intending to find a better substitute later during the project. Realizing we could just reverse the motor we looked for a smaller tub for demonstration practices. We then used a cd stack protector but realized the circle made it incredibly difficult to mount the ultrasonic sensor. In the end we settled on a transparent cylinder, which is not optimal for the photoresistor and laser module, but best demonstrated our product in the five minutes allotted. We also had some oversight when initially coming up with what components would be required to make our final product work, luckily using the power supply for both the laser module and motor was a quick fix. In addition for demo purposes we used food dye instead of actual pollutants, in real life it would be best to use a water quality sensor, however, we were constrained by cost and chose not to buy it. The laser and photoresistor does a sufficient job of detecting the food dye when it is dispersed evenly throughout the tank, however, pollutants do not act in a uniform manner.

4.2 Self-Assessment

Overall, we are content with our products as we feel it performs the goals we set. However, we realize that our product is more of a proof of concept due to its small scale rather than a marketable product. If it were more of a marketable product, instead of the pump reversing to drain there would be drain at the bottom of tank that would open and close. This mechanical drain is incredibly important because if the water were to be polluted it could potentially damage the pump. Otherwise, our finished product contains all the components we anticipated, including some extra(e.g h-bridge), and accomplishes the goal of automating of the water refill and drain process albeit a little slowly.

4.3 Code Attribution

All code is original except part of the code used for getting ultrasonic readings:
https://randomnerdtutorials.com/complete-guide-for-ultrasonic-sensor-hc-sr04/

Figure 6: Project code

```
#define buttR 2
#define buttB 3
#define m1 11
#define m2 12
#define trigPin 9
#define echoPin 10
#define red 5
#define grn 6
#define blu 7
long duration, cm, inches;
long redVal = 7;
long blueVal = 10;
long depth = 15; //depth of the container
// Red button = 6cm, Blue button = 12cm
void setup() {
  // put your setup code here, to run once:
  Serial.begin(9600);
  pinMode(buttR, INPUT);
  pinMode(buttB, INPUT);
  pinMode(m1, OUTPUT);
  pinMode(m2, OUTPUT);
  pinMode(trigPin, OUTPUT);
  pinMode(echoPin, INPUT);
  pinMode(grn, OUTPUT);
  pinMode(blu, OUTPUT);
  pinMode(red, OUTPUT);

  // pin for the lasermodule
  //pinMode(laserPin, INPUT);
}

void photoresistorLaserLED()
{
  if (analogRead(A0) <= 160 && analogRead(A0) > 12)
  {
    analogWrite(grn, map(analogRead(A0), 12, 160, 0, 255));
    analogWrite(red, map(analogRead(A0), 12, 160, 255, 0));
    digitalWrite(blu, HIGH);
  }
  else if (analogRead(A0) >= 161 && analogRead(A0) <= 250)
  {
    digitalWrite(blu, HIGH);
    digitalWrite(grn, HIGH);
    digitalWrite(red, LOW);
    while (ultraRead() < depth)
    {
      digitalWrite(m1, LOW);
      digitalWrite(m2, HIGH);
      delay(10);
    }
  }
  else if (analogRead(A0) <= 12)
  {
    digitalWrite(red, HIGH);
    digitalWrite(grn, HIGH);
    digitalWrite(blu, HIGH);
  }
  else if (analogRead(A0) > 250)
  {
    digitalWrite(blu, HIGH);
    digitalWrite(grn, HIGH);
    digitalWrite(red, HIGH);
  }
}

void manualMotor()
{
  photoresistorLaserLED();
  if ((digitalRead(buttR) == LOW && digitalRead(buttB) == LOW) || digitalRead(buttR) == HIGH &&
digitalRead(buttB) == HIGH)
  {
    digitalWrite(m1, LOW);
    digitalWrite(m2, LOW);
  }
  else if (digitalRead(buttR) == HIGH && digitalRead(buttB) == LOW)
  {
    digitalWrite(m1, HIGH);
    digitalWrite(m2, LOW);
  }
  else
  {
    digitalWrite(m2, HIGH);
    digitalWrite(m1, LOW);
  }
}
```

Figure 7: Project code cont.

```
void autoMotor()
{
  photoresistorLaserLED();
  if ((digitalRead(buttR) == LOW && digitalRead(buttB) == LOW) || digitalRead(buttR) == HIGH &&
digitalRead(buttB) == HIGH)
  {
    digitalWrite(m1, LOW);
    digitalWrite(m2, LOW);
  }
  else if (digitalRead(buttR) == HIGH && digitalRead(buttB) == LOW)
  {
    if (ultraRead() == redVal)
    {
      digitalWrite(m1, LOW);
      digitalWrite(m2, LOW);
      Serial.println("lvl at redVal");
      for (int i = 0; i <= 2; i++)
      {
        digitalWrite(13, HIGH);
        delay(50);
        digitalWrite(13, LOW);
        delay(100);
      }
    }
    else if (ultraRead() > redVal)
    {
      Serial.println("pumping to redVal");
      while (ultraRead() > redVal)
      {
        digitalWrite(m1, HIGH);
        digitalWrite(m2, LOW);
        delay(10);
      }
    }
    else
    {
      Serial.println("draining to redVal");
      while (ultraRead() < redVal)
      {
        digitalWrite(m1, LOW);
        digitalWrite(m2, HIGH);
        delay(10);
      }
    }
  }
  else
  {
    if (ultraRead() == blueVal)
    {
      Serial.println("lvl at blueVal");
      digitalWrite(m1, LOW);
      digitalWrite(m2, LOW);
      for (int i = 0; i <= 2; i++)
      {
        digitalWrite(13, HIGH);
        delay(50);
        digitalWrite(13, LOW);
        delay(100);
      }
    }
    else if (ultraRead() > blueVal)
    {
      Serial.println("pumping to blueVal");
      while (ultraRead() > blueVal)
      {
        digitalWrite(m1, HIGH);
        digitalWrite(m2, LOW);
        delay(10);
      }
    }
    else
    {
      Serial.println("draining to blueVal");
      while (ultraRead() < blueVal)
      {
        digitalWrite(m1, LOW);
        digitalWrite(m2, HIGH);
        delay(10);
      }
    }
  }
}
```

Figure 8: Project code cont.

```
long ultraRead() {
  // The sensor is triggered by a HIGH pulse of 10 or more microseconds.
  // Give a short LOW pulse beforehand to ensure a clean HIGH pulse:
  digitalWrite(trigPin, LOW);
  delayMicroseconds(2);
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin, LOW);

  // Read the signal from the sensor: a HIGH pulse whose
  // duration is the time (in microseconds) from the sending
  // of the ping to the reception of its echo off of an object.
  pinMode(echoPin, INPUT);
  duration = pulseIn(echoPin, HIGH);

  // Convert the time into a distance
  cm = (duration / 2) / 29.1;   // Divide by 29.1 or multiply by 0.0343
  inches = (duration / 2) / 74; // Divide by 74 or multiply by 0.0135
  //  Serial.print(inches);
  //  Serial.print("in, ");
  //  Serial.println();
  delayMicroseconds(10);
  if (cm <= 45)
  {
    Serial.print(cm);
    Serial.println("cm");
    return cm;
  }
  else
  {
    ultraRead();
  }
}

void loop() {
  // put your main code here, to run repeatedly:
  autoMotor();
  //  Serial.println(analogRead(A0));
}
```