Fall
2016

Haptic Ophtalmic

AN INSTRUMENT TO HELP THOSE WHO ARE VISUALLY IMPAIRED

TEAM DAREDEVIL | CSIS 2810-001

Table of Contents

Abstract

According to the World Health Organization, an estimated 285 million people have visual impairments worldwide. Of those 285 million, over 39 million of them are blind. Perkins School for the Blind states that of those numbers only 2 to 8 percent of the people with visual impairments use a white cane, a walking cane for the blind or visually impaired. White canes originated in England in 1930 and have been helping blind and visually impaired people since. The National Federation for the Blind offers free white canes to any person who proves to be blind or have extreme visual impairments.

Why Haptic Ophthalmic?

The white cane continues to be a very helpful gadget in our society, but with all of the new technology advances coming out, is there a better way than the white cane? Our project, "The Haptic Ophthalmic", helps people who have visual impairments and are blind. The Haptic Ophthalmic is a new technology that helps people to maneuver around their environment without the use of a white cane.

How Haptic Ophthalmic Works

The name, "Haptic Ophthalmic" itself relates how the device works – in short, vibration feedback (*haptic)* is provided through data that is inputted at a set of "eyes" (*ophthalmic)*.

We will refer to the device from now on as, *HO*. Our final version of the HO is designed to be affixed to the user's head, but can easily be scaled down for users wanting a less intrusive version that can be placed elsewhere on the user's body such as his or her hand/wrist.

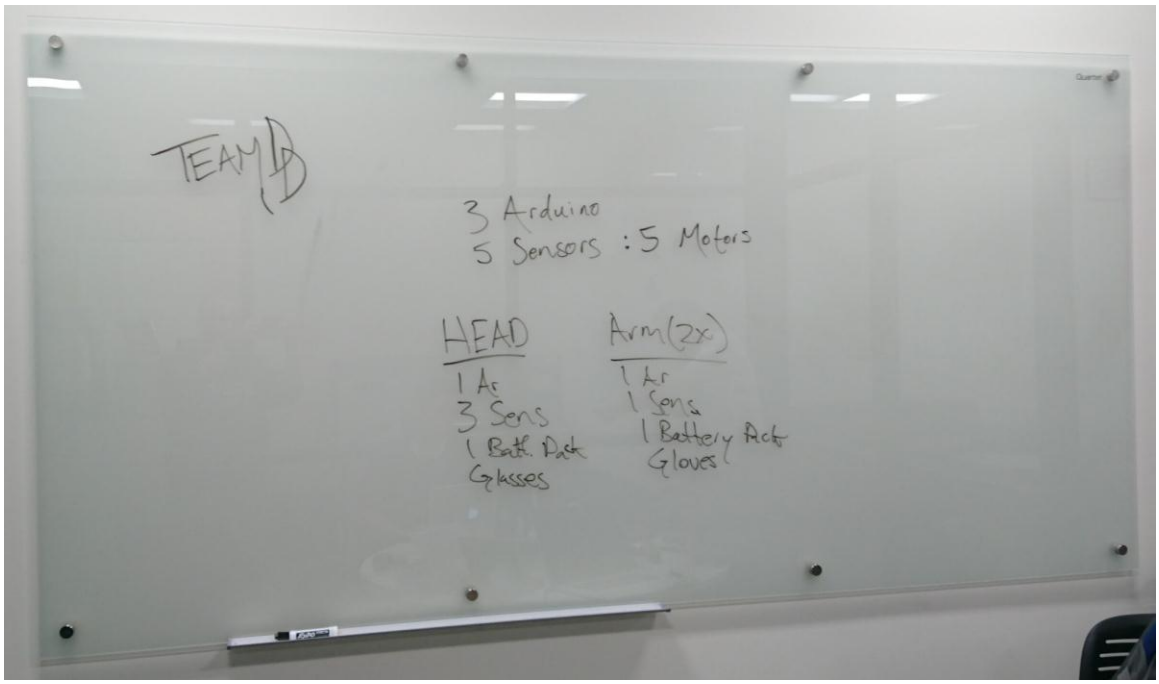
The HO uses existing technologies to interpret the distances of objects, and depending on how close an object is then feedback is provided to the user. The cranial version of the HO uses an Arduino Nano in combination with a total of three ultrasonic sensors and three small vibrating disc motors. Each sensor is programmed to work with its own motor. One sensor/motor combination is placed in front, along the brow of the user to provide feedback for forward facing objects, while the other two sensor/motor combinations are placed near the temples of the user to provide feedback for lateral objects.

Each sensor is programmed to measure distances as large as 300cm (approx. 9.5 feet) and as little as 10 cm, minimum. As the range between the maximum and minimum distances decreases, the motor will begin to vibrate with a stronger intensity, reaching its full strength at 10cm. Using a potentiometer, the user is enabled to either lessen the strength of the motors or can turn them off completely when not wanting to receive any feedback at all.

Background/Problems

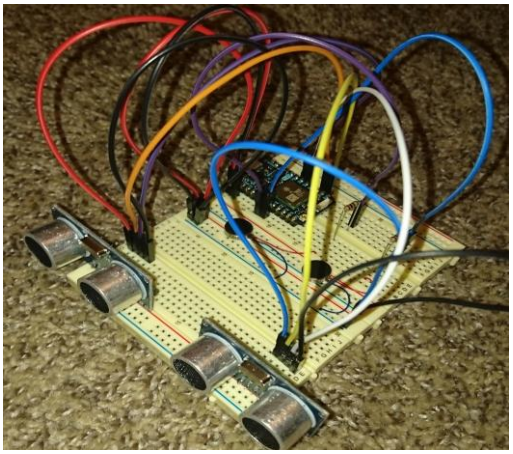
Through multiple brainstorming sessions and learning of Bryce's work with the visually impaired, we decided to pursue a project that could potentially help those looking for other methods to help them get around the world without the use of a white cane.

Michael's experience outside the classroom with microcontrollers and other electronic components led us to quickly being able to put together a parts list.



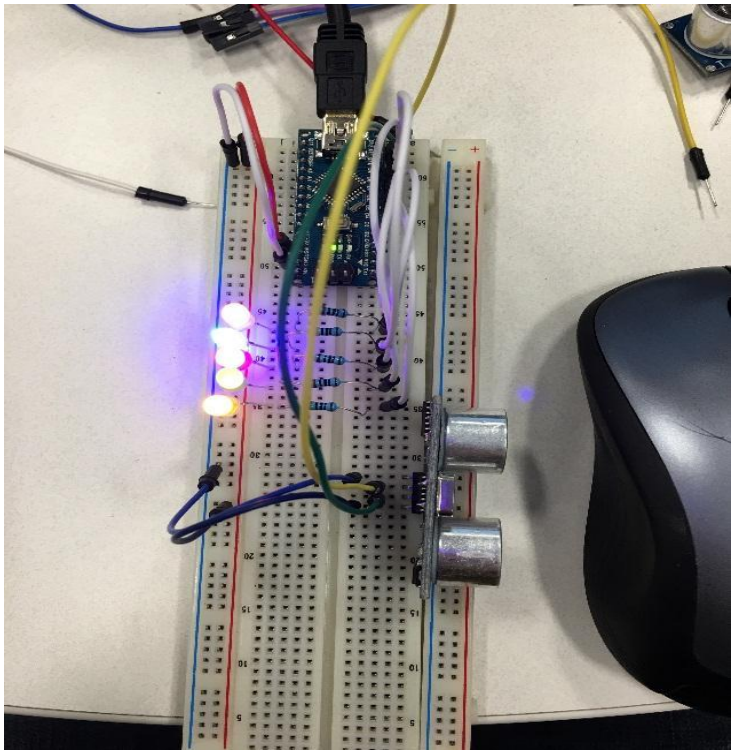
Initial Part List

Prototype Vo: Pseudo Photon Prototype



The initial version created was a photon microcontroller. It was very simple to prototype, however, despite the fact it was simple to wire and link components together using very simple and well documented C++ conventions (we had essentially our V3 prototype equivalent) we decided that for our needs we should use the classic arduino to keep our project cost effective. This was not only a decision made not only for our own benefit, but for the benefit of the maker community that is decidedly more dedicated to using the Arduino.

Prototype V1: A Single Sensor with LEDs



We needed a proof of concept before we attempted to program multiple sensors and motors. Prototype V1 (Figure 1) is comprised of a single sensor and five LEDs to provide visual feedback of specific distance thresholds being passed. As the minimum distance is reached, all LEDs are turned on.

It was at this time we decided that to formulate potential add-ons that we may want to provide.

The intensity of the LEDs provided us with the idea to add a potentiometer to lower their output. We thought that this idea could easily lend itself to being applied towards the motors, but decided to not incorporate it until we had the essentials in place.

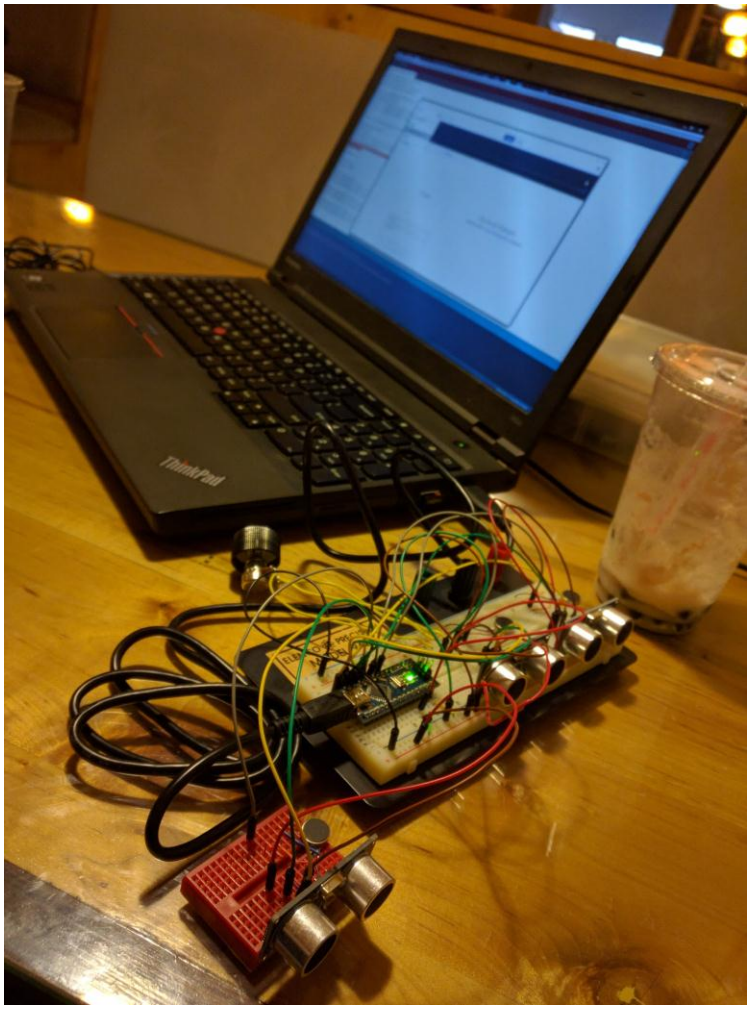
Prototype V2: Write a Motor

One of the challenges here was not only wiring in the motor in place of the LEDs, but we also had to be mindful of the versions to come. We knew we would be adding two more motors to the main aperture. We made the architectural decision to not hard-code singular fields in the code header, instead we placed these values into an array of arrays. Initially this was a single sub-array that only addressed the one motor-sensor pair, however, this would allow us the flexibility to scale out to more sensor-motor pairs in our incremental versions

Prototype V3: Multi-sensor with multi-motors

This was, from a code-standpoint, simple due to our decision from our previous version. We could scale more device pairs arbitrarily, however, we were introduced to a physical complexity we hadn't initially perceived. The original setup pinged all sensors and then checked each one for responses. This created anomalous resulting distances without an immediately obvious reason why. What was discovered was that the ping to the device results in an immediate response in hardware, however, our code was checking after the response was returned to the Arduino. We had to rewrite to save the distance immediately after we queried the device.

Prototype V4: Integrating the potentiometer



Prototype V4: Success! Plus, a bit of brain fuel courtesy of Watchtower Café.

Here we actually ran into the greatest trouble. This seemingly simple component of code resulted in behavior that appeared polynomial instead of linear. The reason appeared to be that we were querying the potentiometer more frequently than we had intended. The solution was to refactor into simpler methods. Once we had simplified our code base the querying the potentiometer no longer reflected this unwanted behavior.

Solution

We now put the pieces together and needed to place them into a single housing. The using the simplest items at our disposal to not only make the project easy for us, but for the maker community at large, we used safety goggles for their cost-effective easy accessibility. Using a simple dremel we created the spaces necessary for the sensors.

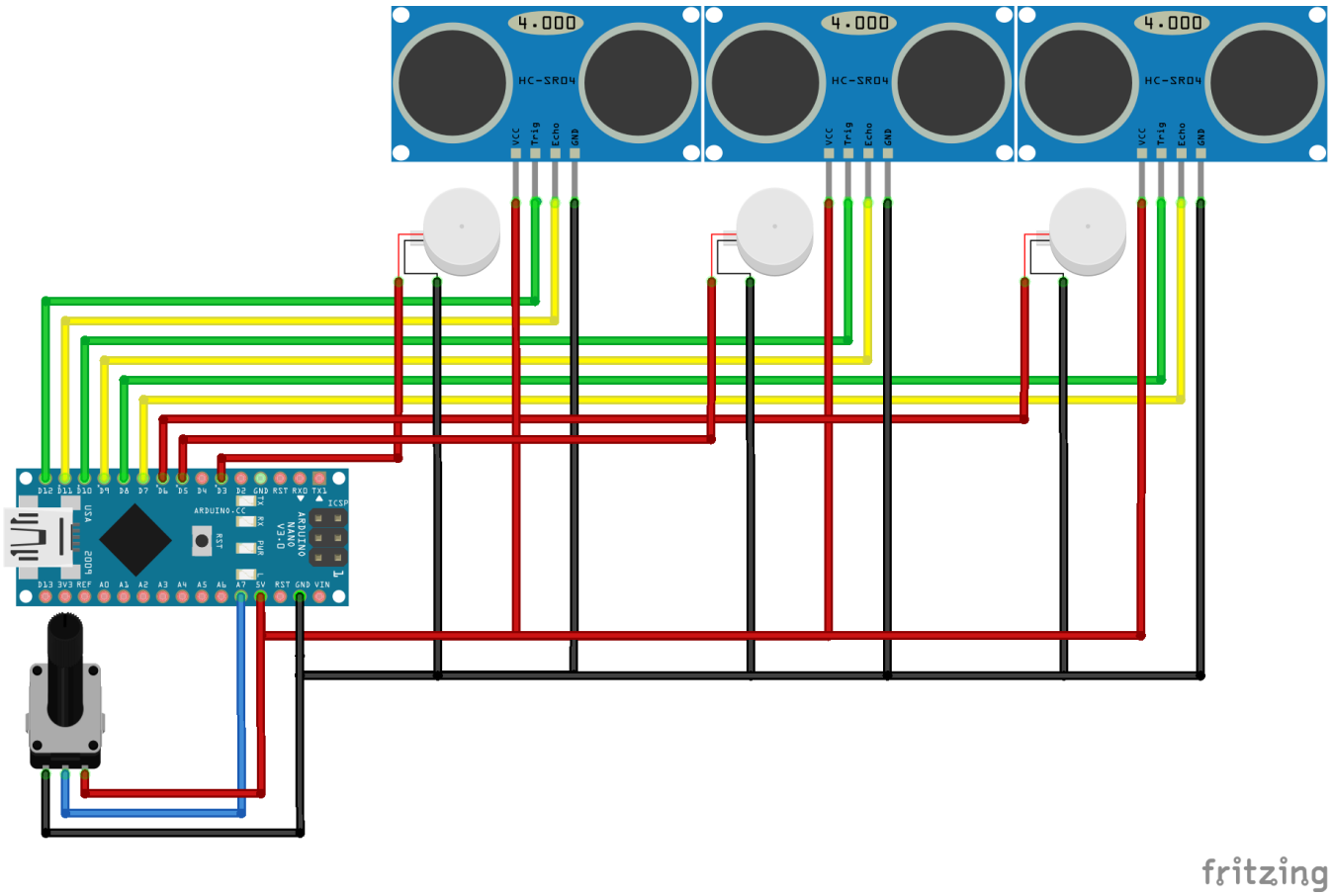
One thing that needed to be addressed is aesthetics. While this particular set doesn't require painting in any way functionally, people want to be able to express their creativity and flair. To demonstrate that in this proof of concept as utility device, the goggles were painted black and given yellow stripes to mimic utility equipment, but customization is limited only by imagination.

Conclusion

Our goal is clear and simple, to help those individuals who are blind or visually impaired to experience a better life. For years, the white cane has been the solution to assist these individuals, but many technological advances have changed and continue to change the way we live. It is time that we use our modern inventions like the Haptic Ophthalmic to better assist the blind and visually impaired. This way is the future for them, and we hope that you see our vision after reading this white paper.

Final Device w/ Fruit Banquet Still Life

Wiring Diagram



Power Supply Excluded. Diagram made with Fritzing.

Source Code

```
//red wire connects 5v on Nano to VCC on SR04
//brown wire connects SR04 to Ground
//Trigger pin of the SR04 sensor - green Jumper
//Echo pin of the SR04 sensor- yellow Jumper

//the nano can analogWrite to pins 3,5,6,9,10,11
//set up distance sensor pins
int sensorMotorCombo[3][3]={
  {12,  11, 3},  // Trigger pin for sensor0 = 11, echo pin for sensor0 = 12, motor
for sensor0 = 3
  {10, 9, 5},    // Trigger pin for sensor1 = 10, echo pin for sensor1 = 9, motor
for sensor1 = 5
  {8, 7, 6}     // Trigger pin for sensor2 = 8, echo pin for sensor2 = 7, motor for
sensor2 = 6
};

//data manipulation variables
long distances[3];
const int MAX_DISTANCE = 300; //this is in cm - we will use this distance to map
the values for analogWrite
const int MIN_DISTANCE = 10; //this is in cm - this is the closest we want to read
from the sensor
const int MAX_SPEED = 200;
unsigned long NEXT_TIME = millis();

void setup() {
  //Serial Port begin
  Serial.begin(9600);

  //Define inputs and outputs
  pinMode(sensorMotorCombo[0][0], OUTPUT); //pin 11 (trigger pin) is output
  pinMode(sensorMotorCombo[0][1], INPUT); //pin 12 (echo pin) is input
  pinMode(sensorMotorCombo[0][2], OUTPUT); //pin 11 (trigger pin) is output

  pinMode(sensorMotorCombo[1][0], OUTPUT); //pin 10 (trigger pin) is output
  pinMode(sensorMotorCombo[1][1], INPUT); //pin 9 (echo pin) is input
  pinMode(sensorMotorCombo[1][2], OUTPUT); //pin 11 (trigger pin) is output

  pinMode(sensorMotorCombo[2][0], OUTPUT); //pin 8 (trigger pin) is output
  pinMode(sensorMotorCombo[2][1], INPUT); //pin 7 (echo pin) is input
  pinMode(sensorMotorCombo[2][2], OUTPUT); //pin 11 (trigger pin) is output}

void loop() {
  // If it is too soon, we will not run the loop again.
  if (millis() < NEXT_TIME) {
    return;
  }
  // Read values from all sensors.
  // Store them into array
```

```
  for (int i = 0; i < 3; i++) {
    int triggerPin = sensorMotorCombo[i][0];
    int echoPin = sensorMotorCombo[i][1];
    distances[i] = readSensor(triggerPin, echoPin);
  }
  // Read value from potentiometer
  double potValue = readPot();
  // Take distances, calculate how much they should be vibrating with relation
  // to distance. Multiply that by the potentiometer value (which is between 0 and
1)
  // as a strength multiplier.
  for (int i = 0; i < 3; i++) {
    int strength = (map(distances[i], MIN_DISTANCE, MAX_DISTANCE, 200, 0) *
potValue);
    writeMotor(sensorMotorCombo[i][2], strength);
  }
  // Set the next time the loop will be allowed to execute again.
  NEXT_TIME = millis() + 60;
}//end of loop

/***
 * Reads potentiometer and returns percentage
 */
double readPot() {
  // Read the value from the pin, it will be between 0 and 1024, we then convert
this to a percentage from this range.
  // This is done with map and then a double divide/cast.
  int pinValue = analogRead(A7);
  return map(pinValue, 0, 1024, 0, 100) / (double) 100;
}

/***
 * Reads a value from a sensor by taking a trigger and echo pin.
 */
long readSensor(int triggerPin, int echoPin) {
  digitalWrite(triggerPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(triggerPin, LOW);
  long duration = pulseIn(echoPin, HIGH);
  long cm = duration / 2 / 29.1;
  if (cm < MIN_DISTANCE) {
    return MIN_DISTANCE;
  } else if (cm > MAX_DISTANCE) {
    return MAX_DISTANCE;
  }
  return cm;
}



/***
 * Writes a value to our motor, this value will be how much it vibrates.
```

```
 */
void writeMotor(int motorPin, int vibrateStr) {
  analogWrite(motorPin, vibrateStr);
}
```

About Team Daredevil

We gave ourselves the name, *Team Daredevil,* due to the nature of the project we pursued. In Marvel Comics lore, there exists a character named Matthew Murdock. He fights crime during the day as a lawyer and at night as a vigilante, using the moniker, *Daredevil*. We felt it appropriate to adopt his vigilante name considering he has a form of blindness, and offers hope to those who are oppressed by the injustices of the world.

Baker, Paul
Paul has been interested in writing code his entire life and since his childhood days wanted to be an engineer. He's currently pursuing a Computer Science degree through the SLCC-Weber partnership program and working as a SDET professionally. After he finishes his degree he hopes to move on to a master's degree and then a Ph.D studying machine learning in the pursuit of artificial intelligence. He's probably the only one that reads the documentation.

Dey, Michael
Mike has studied Computer Science at SLCC for 3 years. He's hoping to get his Associates Degree in two more semesters. He currently works as a video editor, but hopes to change career very soon. He enjoys computer programming and creating physical computing projects with the arduino and other microprocessor devices. He would love to get a career working in that field**.**

Edward, Bryce
Bryce is graduating from SLCC Fall of 2016 with an associate's of CSIS degree. He will be transferring to Utah State University pursuing a Bachelor's degree in Speech and Communication Disorders. He will pursue a career in becoming a speech therapist.
Bryce is currently working as a paraprofessional at Hillside Middle School in Salt Lake. He teaches children with special needs daily life functioning skills

Hardesty, Joshua

Joshua is currently works for Harman Professional as a technical trainer in the Technology and Services Experts division. He provides training for open architecture DSP platforms offered through BSS Audio and amplifier solutions available from Crown. He will be transferring from Salt Lake Community College to Weber State University to finish his Bachelors of Science in Computer Science and Information Systems. He will pursue a career in web development.