Navigation Aid for the Visually Impaired: Utilization of Infrared Obstacle Detection Algorithms and Mobile Devices

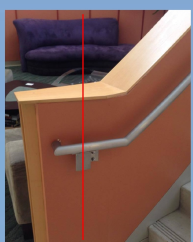
Anup Hiremath and Matthew Stallone
COSMOS UCSD 2015
Cluster 1: Computers in Everyday Life

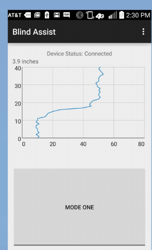
Abstract

In making this project, we aimed to create a system using Bluetooth and IR obstacle detection to make a smart navigational system for the blind. Instead of a system as low-tech as a cane, our system was designed to provide far more useful information about the surroundings. Past attempts to solve this problem either use expensive, complicated technology, or require extensive amounts of training to use. For our project, we had to get information from the Arduino to the phone and use that information to detect obstacles in the surroundings. We then had to convey this information to a blind user in a way that does not involve visual cues. The result is a smart obstacle detection system that can give a blind user a much greater grasp of their surroundings than a cane might.

Applications

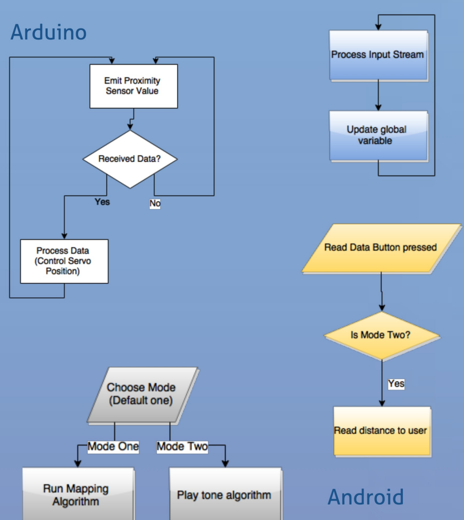
The device can be used by a blind user to navigate through an unfamiliar environment. They can be alerted to obstacles or to a doorway without having to physically tap around. The system of finding obstacles by graphing proximity sensor readings might also be used in mobile robotics to help robots navigate more effectively.





The readings the device might get in Mode 1

Procedure



Introduction

In today's rapidly changing and increasingly interconnected world, getting to where you need to go is more challenging than ever before. For the visually impaired this task is even more daunting. While technology advances at breakneck speed, the blind must rely on outdated technology such as the white cane or guide dog. While advances have been made in this field, all of the devices involve expensive equipment or hours of training to use.

Purpose

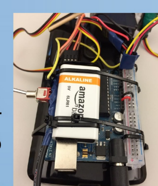
In creating this system, we aimed to create a device that not only worked, but also was easy to see and was implemented on readily available, affordable technology, such as android phones and Arduino microcontrollers.

Goals

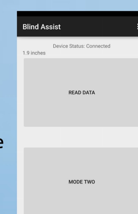
The goals were to connect to phone to the to the Arduino board such that information can be sent both ways, enabling proximity sensor data to get to the phone, calibrate obstacle sensor data to reliably give distance to an obstacle, and create a non-visual user interface that is easy to use for a blind user.

Our convenient wooden case with on/off switch

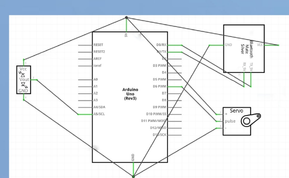
A close-up of our wiring and Arduino

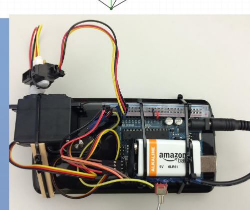


A screenshot of the easy-to-use user interface



Hardware





Above is the circuit diagram and wiring for the Arduino

Software

We used Java on Android Studio for the Android application and C for the Arduino

```java
private class InputHandlerThread extends Thread {
    public void run() {
        if (isConnected) {
            byte[] buffer = new byte[1024];
            int start = 0, end = 0;

            try {
                inputStream.read(buffer);
                for (int i = 0; i < buffer.length; i++) {
                    if (buffer[i] == "#".getBytes()[0]) {
                        if (start == 0) start = i;
                        else {
                            end = i;
                            break;
                        }
                    }
                }
            } catch (IOException e) {
                e.printStackTrace();
            }

            start += 3;
            String in = "";

            for (int i = start; i < end; i++) in += String.valueOf((char) ((int) buffer[i]));

            if (in.length() > 0) {
                proximityData = Integer.parseInt(in);
                double distance = calculateDistanceFromProximitySensor(proximityData);
                outputText.setText(distance == 100 ? "> 100 inches" : (double) ((int) (distance * 10)) / 10.0 + " inches");
            }
        }
    }

}

private double calculateDistanceFromProximitySensor(int data) {
    double distance = Math.pow(data/1064.9, 1/-1.1126);
    return distance > 100 ? 100 : distance;
}
```

The code above uses digital signal processing to process the bit stream from the Arduino. It checks for a hashtag before reading the data, letting us read each message only once.

sketch_jul28a | Arduino 1.6.5

sketch_jul28a

```c
#include <Servo.h>

Servo servo;
String message;
bool isModeOne = 1, dirIsPos = 1;
int pos = 0;

void setup(){
  Serial.begin(9600);
  pinMode(13, OUTPUT);
  servo.attach(6);
  servo.write(110);
}

void loop(){
  while(Serial.available()){
    delay(10);
    char c = Serial.read();
    message += c;
  }

  if(message.length() > 0){
    if(message == "LED"){
      if(digitalRead(13) == HIGH) digitalWrite(13, LOW);
      else digitalWrite(13, HIGH);
    }else if(message == "resetServo") {
      servo.write(110);
      isModeOne = 0;
    }else if(message == "startServo") {
      isModeOne = 1;
    }

    message = "";
  }

  if(isModeOne) {
    if(dirIsPos){
      pos++;
      if(pos > 180) dirIsPos = 0;
    }else {
      pos--;
      if(pos < 50) dirIsPos = 1;
    }
    servo.write(pos);

  }
  Serial.println(String(analogRead(5)) + "#");
}
```

Done uploading.

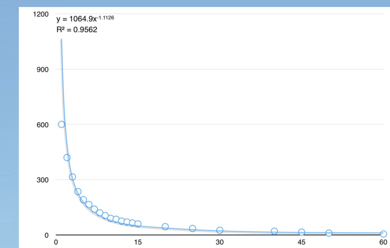
Sketch uses 6,264 bytes (19%) of program storage space. Maximum is 32,256 bytes.
Global variables use 293 bytes (14%) of dynamic memory, leaving 1,755 bytes for local variables. Maximum is 2,048 bytes.

47 Arduino Uno on /dev/cu.usbmodem14100000.1

Challenges

One challenge we faced was to get the Arduino to communicate with the android application. While Android to Arduino communication was relatively simple, the Arduino sends a continuous bit stream that is hard for the phone to decipher. To solve this, we used a form of digital signal processing to only get the data we needed.

We also had to turn obstacle sensor data into distance. We did this by measuring distance vs. sensor data at many intervals and then running a regression. Inverting the regression equation gave us a reliable way to turn sensor data into distance.



Future Improvements

-Put the system in a case to be more aesthetically pleasing

-Integrate pyro-electricity to sense human vs. non-human obstacles

- Give the sensor a 2-D field of view to sense more obstacles

Conclusion

Our system has two modes. One plays a tone depending on the distance the proximity sensor reads, acting like a virtual cane, giving the user an auditory view of the environment. This notifies the user of obstacles long before they hit it. It also can provide the user with the distance to the obstacle upon the click of a button. The other mode will do an upward sweep, tracking the proximity sensor reading at each position, creating a graph of objects directly in front of the sensor. As a result, the height and distance of obstacles can be determined and conveyed via audio output.The combination of both of these modes in a small, convenient arduino attachment makes this project a simple, viable alternative to other electronic obstacle detection devices.

Acknowledgements

Professor Ryan Kastner
Professor Curt Schurgers
Teacher Fellow Shirley Miranda
TAs: Luis Camal, Lucerito Gutierrez, Khelsey Gozum

COSMOS UC San Diego
California State Summer School for Mathematics and Science