cs281: Introduction to Computer Systems

Lab04 Report: Matt Kretchmar

In this lab we describe the design and operation of an audible distance detector.

Problem Definition

Our goal is to build an electronic circuit on a device that implements a distance detector. A distance detector measures the physical distance from our device to a nearby object. Our device produces an audible tone whose frequency indicates the proximity to the nearest object. A lower frequency indicates an object that is further away while a higher frequency tone indicates an object that is nearby. This kind of device could be used in an automobile, for example; the sensor located in the rear bumper signals to the driver how close they are to the nearest object behind the car as they back up.

Our distance device is sensitive to the range of immediate proximity to about 3 meters away (0cm to 300cm).

Hardware Design

To measure distances we use an HC-SR04 Paralax ultrasonic distance device. This device operates by sonar location. It emits an ultrasonic wave (frequency too high for people to hear). Those waves echo off of the first object in the device's path and then return to the device. The device captures the return wave. By measuring the amount of time elapse between the signal and its reflection, we can measure the distance to an object. This is very similar to how a bat echo-locates.

To produce an audible tone, we use a standard buzzer. Applying a waveform to the buzzer produces a tone; the tone's pitch is proportional to the waveform frequency.

The controller for our lab is the Arduino Uno micro-controller. It is an open-source hardware design that is economical, reliable and widely available in the industry.

We construct our device according to the hardware diagram in Figure 1. The ultra sonic device has two controlling pins. The Trigger pin starts the echo pulse; it is connected to Arduino Pin 2. The Echo pin on the sonic device receives the rebound signal; it is connected to Pin 3 on the Arduino. Arduino Pin 2 is configured as an output so that we can start and stop the sonic transmission (trigger the sonic device). Arduino Pin 3 is configured as an input; we can measure the reception of an echo signal from the device.

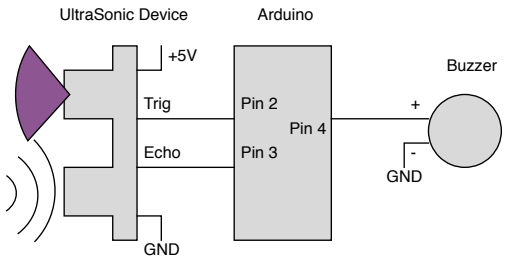

Figure 1: Audible Distance Detector Design

The Arduino Pin 4 is connected as an output to the power supply of the buzzer (+ pin). By toggling Arduino Pin 4 at a certain frequency, we can produce a desired tone on the buzzer. The `tone()` function on the Arduino supplies the necessary square wave to toggle the buzzer.

The whole device is photographed in Figure 2. It is powered by a 9volt battery so that it is portable. The tether cable (blue) shown in the photo is only for downloading the program; it is removed once the program is loaded so that the device is fully portable.

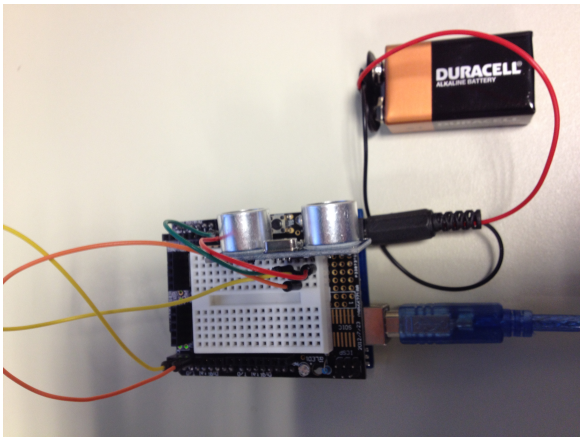

Figure 2: Photograph of Device

Distance Function

Careful attention must be paid to the function relating distance and tone so as to achieve the desired effect. A simple linear relationship is unlikely to produce a

good function. First, people hear tones in a logarithmic scale; a jump from one octave to the next on a piano *doubles* the frequency. Second, we want the tone to increase rapidly as the distance to the nearest object decreases. As we approach an object, there should be a "ramping up" effect in frequency to signal the final proximity to the object. The desired functionality is pictured in Figure 3.

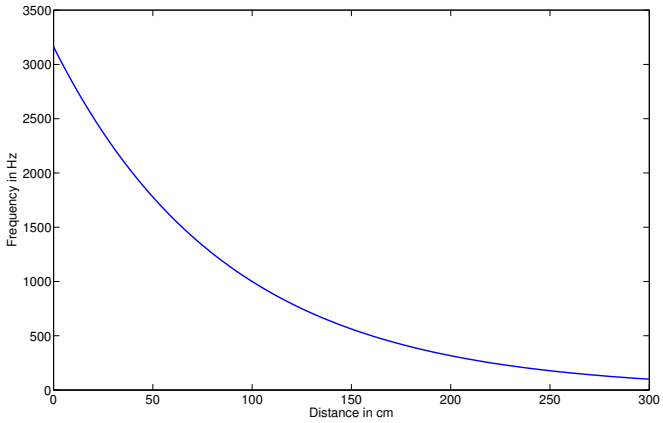

Figure 3: Desired Distance/Tone Function

To achieve this effect, we compute the frequency as follows. Let x be the measured distance to the nearest object (in cm). We compute an exponent e as

$$e = 3.5 - 1.5\frac{d}{300}$$

Then we compute frequency f (in Hz) as

$$f = 10^e$$

This gives us a frequency of approximately 100 Hz (low sound) for longer distances and a frequency of about 3kHz for very near distances. This function was settled on after experimenting with a number of alternative functions and parameters within the function; it produced the most desirable tone characteristics of those we tried.

Software Design

The software design is a straight-forward implementation of two main components, the distance acquisition and the tone production. The main loop of the program

implements these two phases. Because the sonic device often produces noisy input (distances can jump around a bit), we use a smoothing function to take ten distance readings in close succession and average over them. The distance is ”capped” at 300cm so that longer distances produce the same tone as the maximum range of 300cm. The program then computes the desired frequency and uses the built-in `tone()` function to produce the output square wave for the buzzer.

```
// Matt Kretchmar Design
#define TRIG 2
#define ECHO 3
#define BUZZER 4

void setup () {
  pinMode(TRIG,OUTPUT);
  pinMode(ECHO,INPUT);
  pinMode(BUZZER,OUTPUT);
}

void loop () {
  int distance, i;
  int total = 0;
  // take 10 distance readings
  for ( i = 0; i < 10; i++ )
  {
    digitalWrite(TRIG,HIGH);
    delay(10);
    digitalWrite(TRIG,LOW);
    distance = pulseIn(ECHO,HIGH)/2;
    total = total + distance;
  }
  distance = total / (10*29);
  // cap distance at 300 max
  if (distance > 300 ) {
    distance = 300; }

  float f = 3.5 - 0.005 * float(distance);
  int freq = int(pow(10,f));
  noTone(BUZZER);
  tone(BUZZER,freq);
}
```