New Mexico

SUPERCOMPUTING CHALLENGE

ROBOTIC GUNSHOT DETECTION SYSTEM: SCOOBY

Capital High School

Team: SCOOBY (CHS 1)

Team members:

Raul Alvarado Villalobos

Craig Andree Abajo

Jesse Burch

Haize Christiansen

Jordan Lam

Britny Marquez

Edward Scott

Teacher Sponsor:

Barbara Teterycz

Mentor:

David Ritter

April, 9 2024

Table of Content

Abstract/Executive Summary

Our country has been challenged with an ongoing and escalating issue of mass shootings in schools across the United States. According to the Center for Homeland Defense and Security, there have been 2069 school shootings in our country between 1970 and June 2022 with 684 killed. The list of schools is long: Virginia Tech, Sandy Hook Elementary, Robb Elementary, University of Texas-Austin, Parkland High, Columbine High, Texas Santa Fe High, Umpqua Community College, Red Lake Senior High, and hundreds of other schools. The deadliest year was in 2018, where 51 people were killed. In response to this critical need our team has been developing a robotic gunshot detection system, called SCOOBY, that can recognize the sound of a gunshot, precisely locate a shooter on the first shot, and set off appropriate alarms and distractions.

The project first started off with an Arduino Nano, but over time, the team learned that the Nano just did not have enough memory to complete the calculations needed for precise timing and location. The project was moved onto a Raspberry Pi microcontroller, which contained more memory and much higher processing speed. The peripherals connected to the microcontroller are: 4 directional microphones, 1 waveform microphone, an LCD display, SD drive, real time clock, mic amplifiers, comparators and a printed circuit board. Scooby's software was written using C++ and Python and has been expanded to test new ideas for the system. After the project was built and programmed, it was tested in various scenarios/experiments, and the data was collected and analyzed. As a safer and more reasonable way of testing the project, the team decided to use popped balloons at higher pressure thresholds than normal, as their sound waves are very similar to a gunshot.

After testing SCOOBY, it was able to correctly point towards the balloon as well as discern the difference between a balloon and an everyday sound. The key features (characteristics of the waveform) identified during data exploration and analysis include: leading edge time, peak, peak duration, and the zero crossing time. By installing a system of SCOOBY units across a school, and having them communicate to a main hub via wires, schools can determine if a loud sound was a gunshot and where it was located. This will prevent the future deaths of students and staff.

Research and Background

School shootings in our country have become an everyday occurrence. In 2022, there were 240 non-active and active shooter events. This was the deadliest year here according to the Center of Homeland Defense and Security School Safety Compendium. Here in New Mexico, there have been 13 school shootings which led to 6 deaths and 6 injuries. Some of the deadliest school shooting events in the US include Virginia Tech (33 killed in 2007), Sandy Hook Elementary (28 killed in 2012), Robb Elementary (21 killed in 2022), University of Texas-Austin (18 killed in 1966), Parkland High (17 killed in 2018), Columbine High (15 killed in 1999), Texas Santa Fe High (10 killed in 2018), Umpqua Community College (10 killed in 2015), Red Lake Senior High (10 killed in 2005). The statistics of the number of students and staff killed during these events are grim. It's important that schools have security systems that can save people's lives in such events.

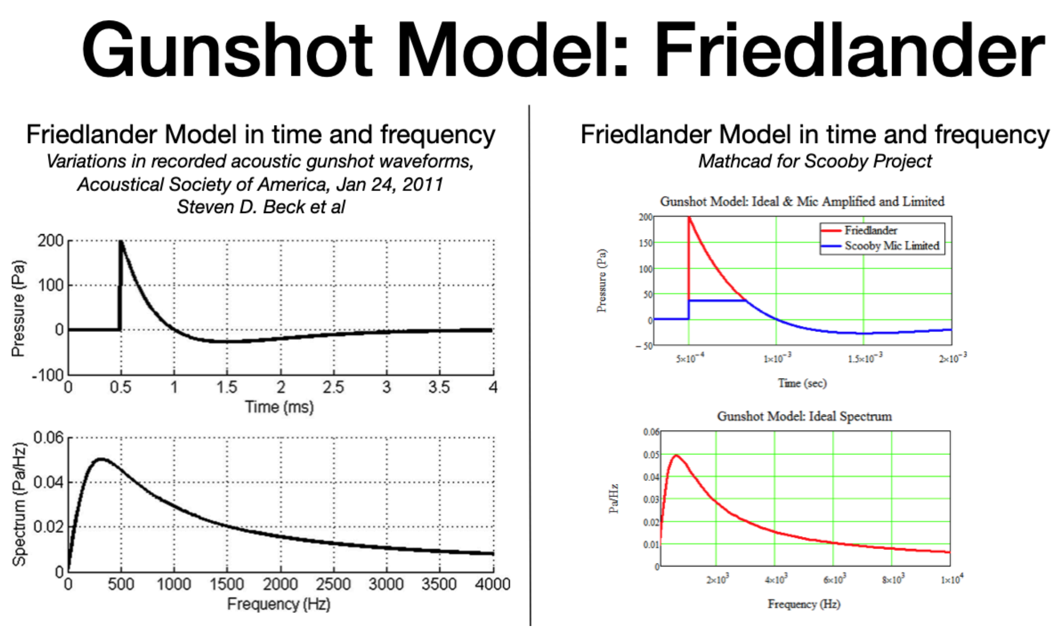
The inspiration for this project came from a gunshot detection system which was implemented in Chicago in August 2018 by a company called Shotspotter. This system was implemented over 117 square miles with 15-20 sensors placed per square mile. When their sensors detected a loud sound, it determined whether the sound was a gunshot. After it verified the sound as a gunshot, the system alerted local authorities and informed where the gunshot was located. This system contained many flaws which led other cities to drop contracts with Shotspotter. Safety officials questioned the efficacy of the system, since only 1 out 9 alerts included a gun. This system was costly, the city of Chicago signed a 3 year, $33 million contract for Shotspotter to implement this system. Also, a judge in Manchester, New York dismissed evidence provided by the Shotspotter system in weapon conviction since it “was not reliable scientific proof”. Shotspotter is also selling its system to school districts and individual schools, but there are many privacy concerns. People are afraid it could pick up people's conversations, leading it to become a spying device. SCOOBY takes all these factors into consideration. For example, it cannot record a human voice, it is bandwidth (1kHz) and time limited (2msec).

The Acoustical Society of America used the Friedlander Model to study gunshot characteristics in their paper “Variations in recorded acoustic gunshot waveforms generated by small firearms”. This model produces the ideal gunshot pressure waveform. The Friedlander model is given by this equation for the pressure wave after ‘tstart’:

$$p(t) := \left[P0 + Ps \cdot \left(1 - \frac{t - tstart}{T0}\right) \cdot e^{-b \cdot \frac{t - tstart}{T0}} \right]$$

Where:

- p(t) is the pressure at time t,
- P0 is atmospheric pressure, set to zero here to reflect gauge pressure (re atmosphere),
- Ps is the peak pressure, set to 200 Pa (pascals) to reflect a sound level peak of 140dB (decibels) SPL (sound pressure level)*,
- T0 is the positive duration of the pulse, set to 500 µsec,
- e is Euler's number ≅ 2.718281828459045,
- b is the exponential decay constant, set to 1 (typical),
- tstart is the time that positive pressure (time effectively zero) starts.

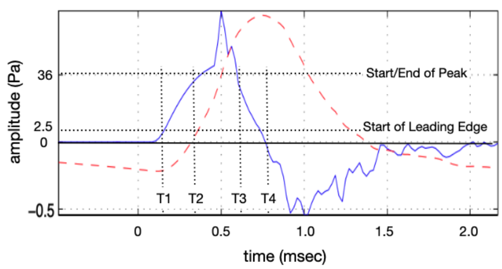


The waveform of a gunshot is characterized by the sudden rise of pressure at start then followed by a peak and exponential decay. Major features are the length in time of the gunshot (~2msec), and the frequency content of the waveform (substantially below 1000Hz). The

Friedlander model was coded into Mathcad so that the limitations of SCOOBY's microphone and signal processing can be included. The blue curve shows the waveform that arrives at SCOOBY's ADC (Analog to Digital Converter). Although the absolute peak is lost in this implementation, its nature is still reflected in the duration of the peak which is a key feature used in SCOOBY's code.

However, due to safety protocol, gunshots have been substituted for the balloon pops (since their acoustic properties are so similar). Both sounds have a SPL (sound pressure level) peak of about 140dB (decibel) and a peak duration of about 500 μsec.

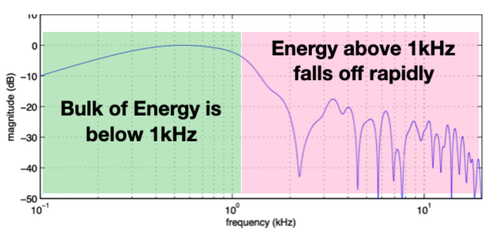
Balloon Pop Waveforms and Spectrum



Waveform Properties

1. Leading Edge: is less steep than a gunshot but still in reasonable range for Scooby. We don't use the slope of leading edge, but just look for the 50% point to identify the start of the event relative to the trigger point from other microphones.
2. Peak: length of the peak from balloon pop is similar to gunshot and falls in the range around 350usec. This a key feature of both balloons and guns and they are very similar.
3. Zero Crossing: for a valid waveform it must cross zero after the peak. This is also a key distinguishing feature of both balloon pops and gunshots.

Note: red line is the integral of the waveform (used in source paper) which we don't use here.



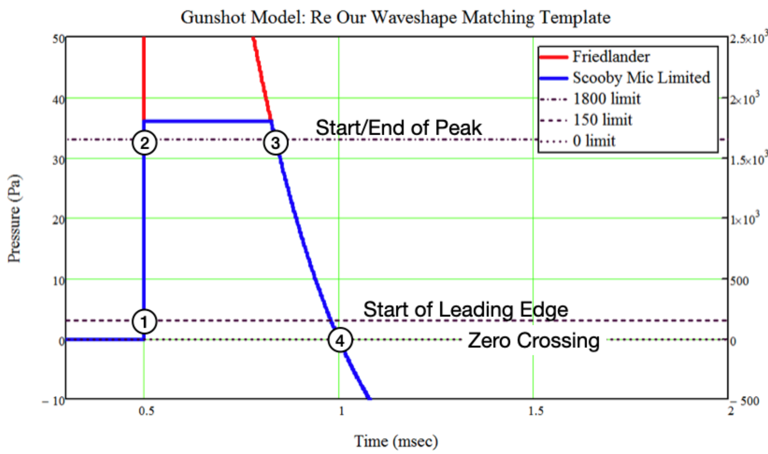
Spectrum Properties

1. Most of the energy is below 1kHz, so we can apply filters to ignore higher frequencies. This is very similar to gunshots.
2. Energy above 1kHz falls off rapidly. We can ignore this content.
3. Our trigger microphones have filters included to remove frequencies above 1kHz and emphasize those below. This helps prevent false triggering on random loud sounds.

Balloon pops (balloon inflated consistently to bursting pressure using an automated inflator fixture) are used as surrogates for gunshots because they have similar characteristics. A balloon pop has a rapid leading edge waveform, followed by a peak and a zero crossing with timing similar to a gunshot. Studies show that a balloon inflated until it bursts has the same acoustic energy as a gunshot.

The figure below demonstrates the details of the relevant part of the waveform (less than 2 msec). In this region only, the direct sound from the source reaches the microphone. There are no reflections to distort the waveform. In this plot the important amplitudes of SCOOBY's discriminator are shown.

Gunshot Model: Detail and Matching



Key Features

1. Leading Edge: characterized by start of leading edge (point 1) and start of peak (point 2).
2. Peak: characterized by start (point 2) and end (point 3) of peak.
3. Zero Crossing: characterized by waveform crossing zero (point 4).

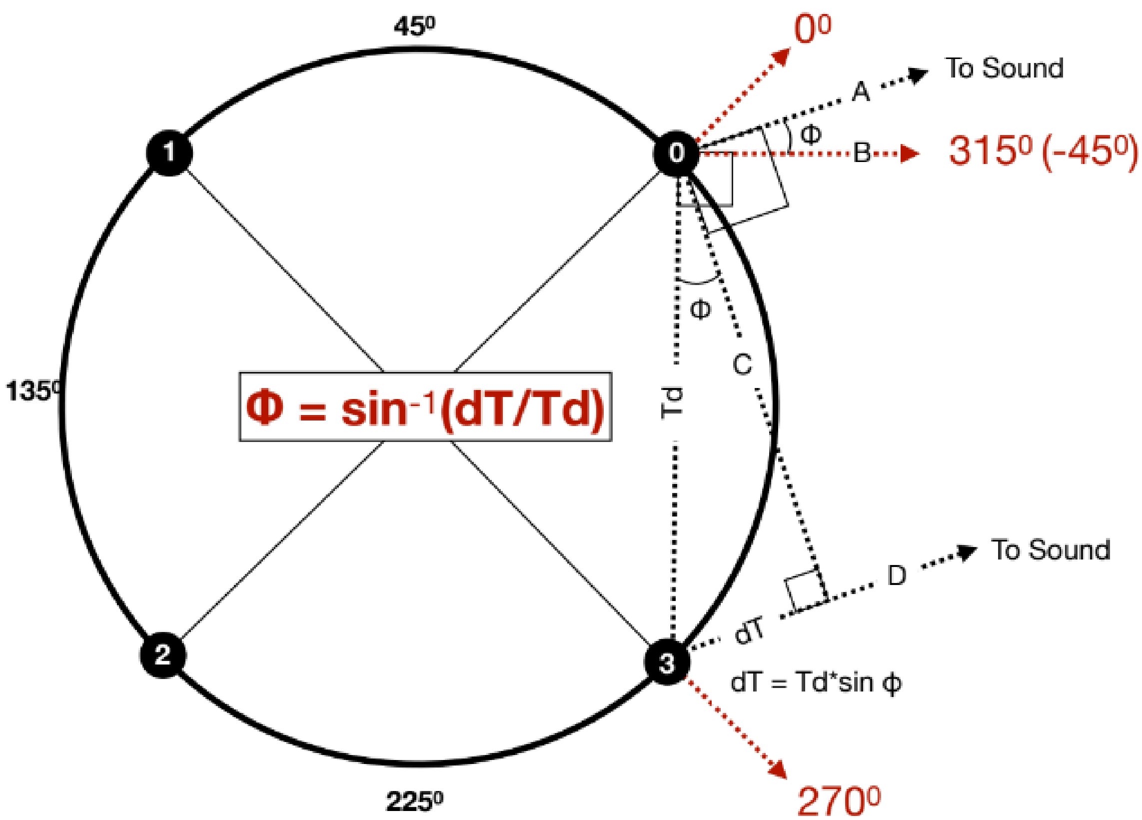
Main Goal

The main goal of this project is to develop a robotic gunshot detection system called SCOOBY that can recognize the sound of a gunshot, precisely locate a shooter on its first shot, set off appropriate alarms and distractions. The plan is to make multiple units of SCOOBY and make it as cheap and accessible as possible, so schools can easily implement it into their respective security systems.

Selected Approach

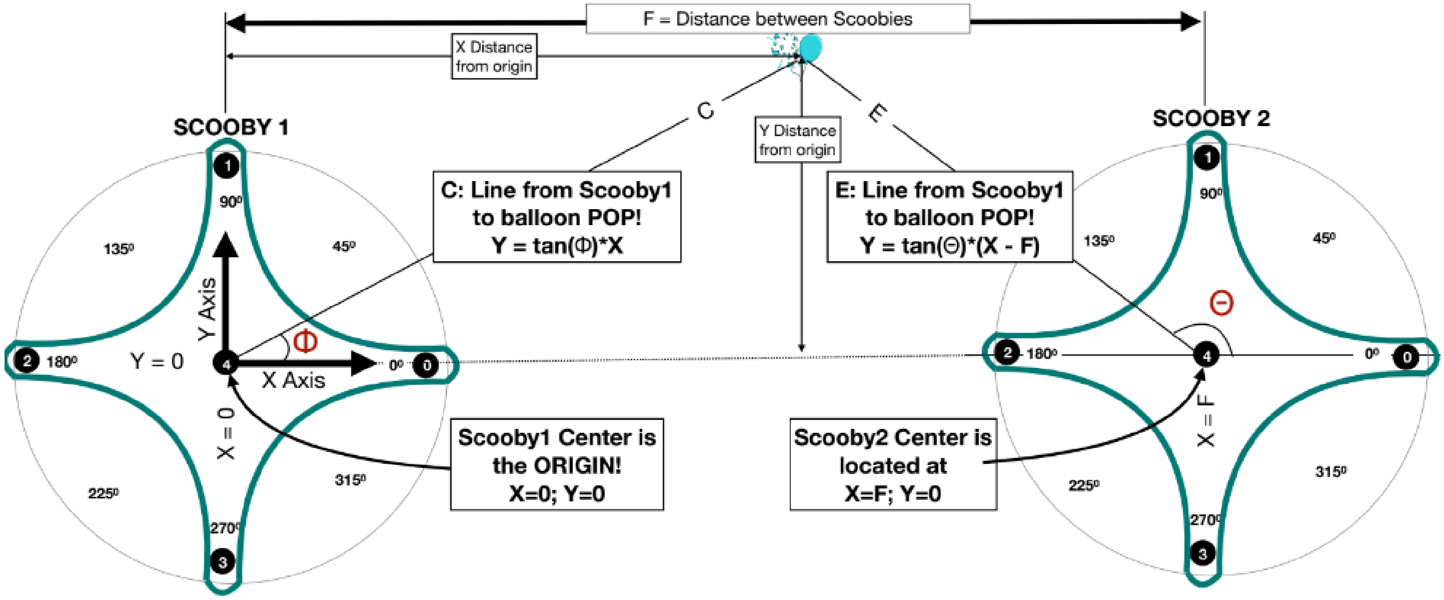
Angle Calculation

SCOOBY calculates the angle of a gunshot relative to its microphones using geometry and trigonometric functions. All of the precise timing to accomplish this is included in the SCOOBY unit. It does this with the use of the first two microphones on the unit being triggered. In the diagram (below) lengths are labeled as times by applying the speed of sound. So dT (difference time), while it is a time, is proportional to the extra distance a sound has to travel between microphones (in this case Mic3 to Mic0). The distance between mics at Td (timed distance) is calibrated as a distance which again is a time proportional to distance. C is simply the distance between microphones calibrated in delay time between them. This defines a triangle [Td, dT, C] and an angle Φ. Given these assumptions, the sin of Φ is simply dT/Td, and Φ is computed as the inverse sin of dT/Td, shown in the diagram below. This method has the advantage of keeping all critical dimensions and timing inside a single SCOOBY. SCOOBY reports events and angles, therefore multiple SCOOBIES can triangulate without requiring critical synchronization.



Triangulation

SCOOBY would be connected into a main hub for multiple units to send data at once. This data will then be used to triangulate a gunshot within a room or other space. For basic triangulation two units are needed. Each unit reports the angle to the detected sound. These angles (along with the position and separation of the units) will then be used to calculate the coordinates of the detected sound.



Two lines are drawn from each unit to the location of the detected sound: C for the first unit and E for the second. They intersect at the location of the detected sound. The center of the first unit defines the origin of the coordinate plane, (0,0). The second unit is at a known distance of F, with its center at (F,0). The angle of the first unit to the detected sound is phi and the angle from the second unit is theta. The equation of each line is y = mx + b, which can be rewritten as y = m (X-X0), where X0 is the X intercept of the line. The slope of line C is m = tan(Φ) and for line E is m = tan(Θ). The equation for line C is, y = tan(Φ) * X, while for line E it is, y = tan(Θ) * (X - F). These two equations are then used to calculate the X coordinate.

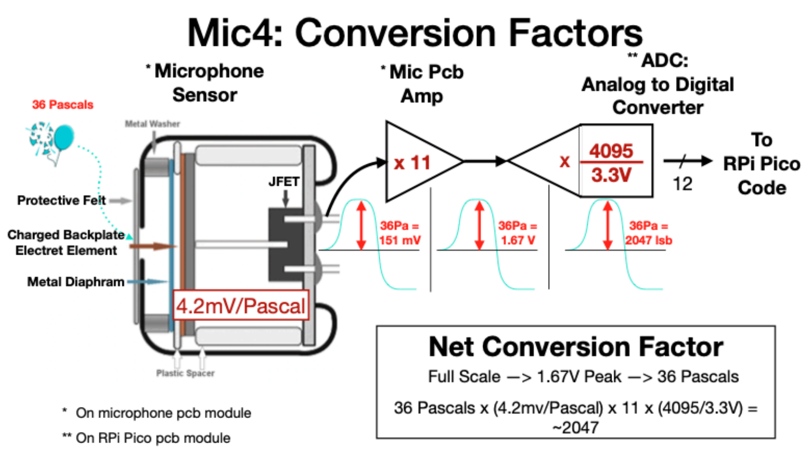
1. Slope of C is $\tan(\Phi)$, Slope of E is $\tan(\Theta)$
2. Equation of C: $Y = [\tan(\Phi) * X]$
3. Equation of E: $Y = [\tan(\Theta) * (X - F)]$
4. $[\tan(\Phi) * X] = [\tan(\Theta) * (X - F)]$
5. $[\tan(\Phi) * X] = [\tan(\Theta) * X] - [\tan(\Theta) * F]$
6. $[\tan(\Theta) * X] -]\tan(\Phi) * X] = [\tan(\Theta) * F]$
7. $[\tan(\Theta) - \tan(\Phi)] * X = [\tan(\Theta) * F]$
8. $X = \tan(\Theta) * F / (\tan(\Theta) - \tan(\Phi))$
9. $Y = \tan(\Phi) * X$ ... from (2)

The formula to calculate the X coordinate will always be x = tan(Θ) * F / (tan (Θ) - tan(Φ). There are some instances when triangulation could fail. If tan(Φ) = tan(Θ) then the angles to the detected sound are equal and the sound is too far away for a meaningful result. By the time the wave hits the microphones, the curvature of the wavefront is gone and the sound hits both microphones at the same time. This is a parallel error: the sound is so distant that C and E are effectively parallel. It also fails anywhere along the X axis where both tan(Φ) = 0 and tan (Θ) = 0.The equation for X results in a 0/0 indeterminate form..

In the remaining cases there are two places where the calculation becomes less reliable. Small errors in angle can make huge differences. To avoid these cases there are two ways to calculate Y from the X solution. The first one is when $|\tan(\Theta)| < |\tan(\Phi)|$ or when the detected sound is closer to the first unit. In this case it is advantageous to calculate the Y coordinate using: $Y = \tan(\Theta) * (X - F)$. While for the second case, when $|\tan(\Theta)| > |\tan(\Phi)|$ or when the detected sound is closer to the second unit it is better to use $Y = \tan(\Phi) * X$. If, for example, the wrong solution is used, say, $Y = \tan(\Phi) * X$ when X is very small, $\tan(\Phi)$ will be very large and small variations in X will make a huge difference in the result. The alternative is $Y = \tan(\Theta) * (X - F)$, where $\tan(\Theta)$ will be F times smaller than $\tan(\Phi)$ and $(X - F) = \sim F$. This results in a much more stable solution.

Microphones

SCOOBY contains two types of microphones, four microphones for direction (0-3 on the top of the unit) and one microphone for waveform capture (in the center of the circle, detail below). The four directional microphones are used to determine the direction of a gunshot relative to Mic0, as demonstrated in the diagram above. The fifth microphone in the center collects samples of the waveform in a smoothly changing analog signal which is then converted to digital. This arrangement guarantees that a trigger mic will be triggered before the pressure wave reaches the center mic. That way the entire waveform is captured. Conversion factors (Pascals to digital code) are shown below.



Main Code

SCOOBY's Code is written in C++ language. There are two main parts of SCOOBY's code: the first part being the Main Code where all the Setup functions and Loop functions are contained. The second part of the code being the library, SCOOBY's library is where all the necessary functions and variables are defined. The Setup function in the code is where serial communication is set up and assigns the pins that are being used. It also creates a file for SCOOBY to record its data in.

```cpp
void setup() {
  analogReadResolution(12);    //
  USBIn=analogRead(pinUSBIn);


  setupSerial();              //set up of serial
  Wire.setClock(780000);
  Wire.begin();


  setupMicPins();     // Assign GPIOs to Mics, LEDs, LCD, SPI Bus and LED Ring.
  setupLedPins();
  setupAnalogPins();
  setupSdSpi();
  setupLCD();
  splashLCD();
  setupRing();


  sprintf(datafileName, "Scby%02d%02d.txt", myClock.getMonth(century),myClock.getDate());
  wait4trigger();
```

While the Loop function activates after the setup is complete, the loop runs till it detects a loud sound/trigger. When SCOOBY hears a trigger, it will immediately record that data and calculate the angle of the sound. Then proceed to run the loop function all over again, looking for another loud sound to detect.

```cpp
void loop() {
  inPort = getPort();   // Sample ports
  if(getData()) {       // if inPort != 0 grab Data (samples, bits plus Mic4 waveform)
    clearRing();        // turn off all LEDs in ring
    clearTrigTimes();   // Set trigger bits to zero.
    openSDFile();       // If SD connected open the data file
    printLCD(0,0,lcdString0 );
    getDataRecord();    // Read 'samples' from mics
    printMics();        // Print mics in index order, not triggered order
    getMicOrder();      // Put mics in order of trigger times
    print1stMic();      // Print first mic
    getQuad();          // Figure out quadrant/angle of trigger
    reverbTime = wait4Echos(); // Listen and wait for echos to stop
    updateLCD();        // Print out to LCD
    updateRing();       // Light up appropriate LED on ring.
    printParams();      // Print results to monitor page (Serial) and SD
```

Waveform Matching Code

SCOOBY has code that discriminates between everyday sounds and balloon pops. The code looks at key features of the soundwave and does some calculations to determine if a sound is a balloon pop. The code was first developed in Python to test and develop a working algorithm for analyzing key points in a soundwave. Later it was translated into C++ and further optimized to run faster.

Waveform Matching Python Model Code

```
def leadingEdgeTimeTest(wfm, pk, time):
  leStart = lookForLES(wfm)
  leStartTime = time[wfm.index(leStart)]
  pkTime = time[wfm.index(pk)]
  leAvgTime = (leStartTime + pkTime)//2
  leAvgTimes.append(leAvgTime)
  if leAvgTime >= leadingEdgeTimeConstraint1 and leAvgTime <= leadingEdgeTimeConstraint2:
      return True
  else:
    return False
```

Scan for Leading Edge Time and test for valid leading edge

```
def testDurationPeak(time, start, end):
  durationOfPeak=(time[end]-time[start])
  if durationOfPeak > 100 and durationOfPeak < 700:
    return True
  else:
    return False
```

Scan for the Duration of the Peak and test for valid duration

```
def testForZeroCrossing(wfm, endIndex):
  for n in wfm:
    if n < 0:
      negIndex = wfm.index(n)
      if negIndex > endIndex:
        return True
  return False
```

Scan for Zero Crossing

Waveform Matching: Scooby C++ Code

```
// -------------------- Point 1 Search: start of leading edge ------------
  while( (Waveform[wfmCursor] ) <= ampLimit1) { // Search for point 1.
    wfmCursor += 1;
    if(wfmCursor >= samples) { wfmError = 0x10; return FALSE; } // if search fails
  }                                                              // Set error flag to 8
  timePoint1 = sampTime[wfmCursor]; // get the time of point 1
// -------------------- Point 2 Search: start of peak --------------------
  while( (Waveform[wfmCursor] ) <= ampLimit2) {  // Search for point 2.
    wfmCursor += 1;
    if(wfmCursor >= samples) { wfmError = 0x20; return FALSE; } // if search fails
  }                                                              // Set error flag to 16
  timePoint2 = sampTime[wfmCursor]; // get the time of point 2
// -------------------- Point 3 Search: end of peak ----------------------
  while( (Waveform[wfmCursor] ) >= ampLimit3) {  // Search for point 3.
    wfmCursor += 1;
    if(wfmCursor >= samples) { wfmError = 0x40; return FALSE; } // if search fails
  }                                                              // Set error flag to 32
  timePoint3 = sampTime[wfmCursor]; // get the time of point 3
// -------------------- Point 4 Search: zero crossing ---------------------
  while( (Waveform[wfmCursor] ) >= ampLimit4) {  // Search for point 4.
    wfmCursor += 1;
    if(wfmCursor >= samples) { wfmError = 0x80; return FALSE; } // if search fails
  }                                                              // Set error flag to 64
  timePoint4 = sampTime[wfmCursor]; // get the time of point 4
// --------------Calculate Critical Features ----------------------
  leadingEdge = (timePoint1 + timePoint2)/2;  // Calc midpoint of leading edge
  peakLength = (timePoint3 - timePoint2);     // Calc length in time of peak
  zeroCrossing = timePoint4;              // This will be 0 if point 4 search fails.
// ---------------Test and Generate Error Flag ---------------------
// Waveform error flag: 0 means Okay, lower 3 bits flag following errors
  if ( (leadingEdge < leTime[0]) || (leadingEdge > leTime[1]) ) wfmError = 1; // check leading edge
  if ( (peakLength < pkTime[0]) || (peakLength > pkTime[1]) ) wfmError += 2;  // check peak duration
  if ( (zeroCrossing < zeroTime[0]) || (zeroCrossing > zeroTime[1]) ) wfmError += 4; // check zero crossing

  if(micFail || badSeq || (wfmError !=0) ) return FALSE; // Return False if any error.
  else return TRUE;
```

SCAN FOR START OF LEADING EDGE
Point 1: start of leading edge

SCAN FOR START OF PEAK
Point 2: start of peak (max pressure)

SCAN FOR END OF PEAK
Point 3: end of peak (max pressure)

SCAN FOR ZERO CROSSING AFTER PEAK
Point 4: zero crossing of trailing edge

CALCULATE KEY FEATURES
Leading Edge time, Peak Length, and Zero Crossing time

COMPARE TO LIMITS
Apply maxes and mins to test validity

Return FALSE if any test fails

Triangulation Code

SCOOBY's triangulation code is written in Python, the code will be in the main hub. The code reads the serial ports from each unit to get phi and theta. The code code first looks for errors. If there are no errors with the angles, it determines which case is true to calculate the location of the detected sound. Next the calculation of the location of the detected sound is done.

```
if errorDetector(phi, theta) == True:
  print("Error, please try again")
elif abs(math.tan(theta)) < abs(math.tan(phi)):
  calculateXandYcase1(phi, theta, Fdis, units)
elif abs(math.tan(theta)) > abs(math.tan(phi)):
  calculateXandYcase2(phi, theta, Fdis, units)
```

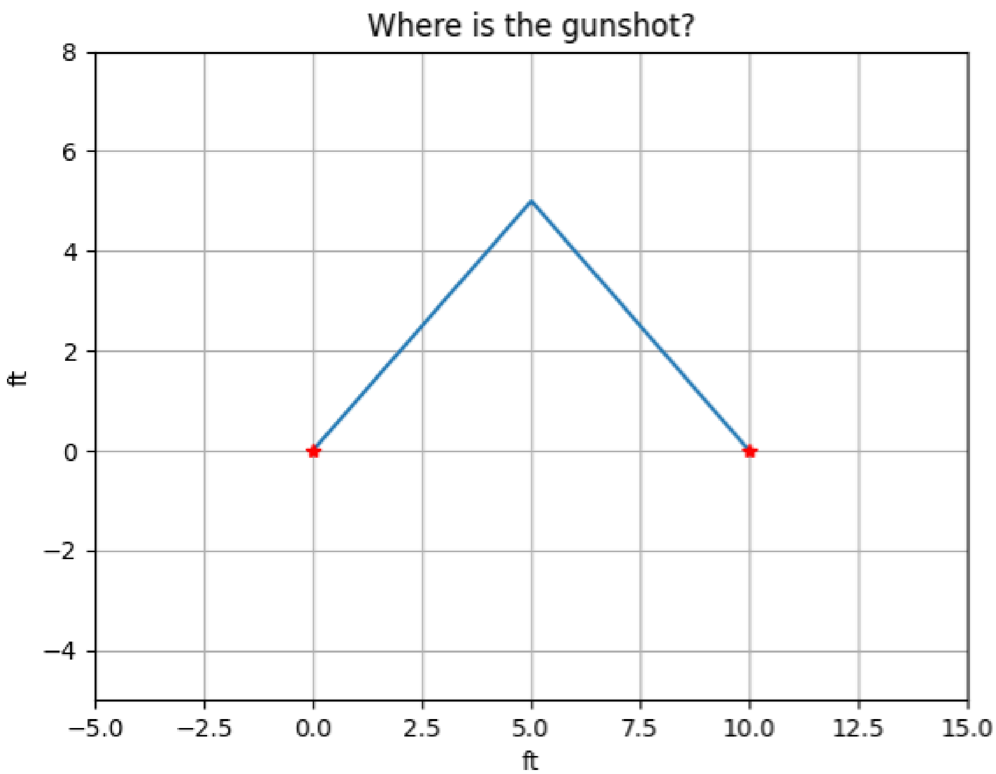
Determines if errors are present and which case of calculation will be done

```
def calculateXandYcase1(phi, theta, Fdis, units):
  print("Case 1 Calcuation:")
  x = math.tan(theta) * Fdis / (math.tan(theta) - math.tan(phi))
  y = math.tan(theta) * (x - Fdis)

  x = round(x, 2)
  y = round(y, 2)
```

Calculation of the location is done

After the code calculates the location of the detected sound. It is then plotted in a graph. The graph represents a room where two units would be placed. There are two red stars on the graph, these are the two units in the room. The two lines intersect at the location of the detected sound. In this case the angle of Phi is 45 and Theta is 145. The graph is produced below.



Machine Learning Algorithm and Data Collection

The system will also have a Machine Learning Algorithm in the future, which will allow predictions to be made with incredible accuracy. To achieve this and triangulation, parsing the data using a main hub is necessary. The hub has its own data collection code, in which it collects, parses and sanitizes data collected from SCOOBY's COM port, it can also plot data using PyPlot, which can generate graphs of the waveform.

Collection Code

Collecting Data:

```
serialCom = serial.Serial('/dev/<SCOOBY_COM_PORT>', 115200, timeout=0.2)
data = ""

incomingData = unit1.readline().decode('utf-8').replace('\n', '')
data = data + incomingData
```

Program initializes link to COM port and starts recording data.

Parsing Data:

```
waveformArray = []; dataArray = []

initialArr = data.split('\r')

for i in range(len(initialArr)):
        if "   " in initialArr[i]:.
                waveformArray.append(int(initialArr[i].split("   ")[2]))
                dataArray.append(int(initialArr[i].split("   ")[0]))
```

Empty arrays are created and a temporary array is filled with data,

This data is then ran in a for loop to parse the data into its plottable form.

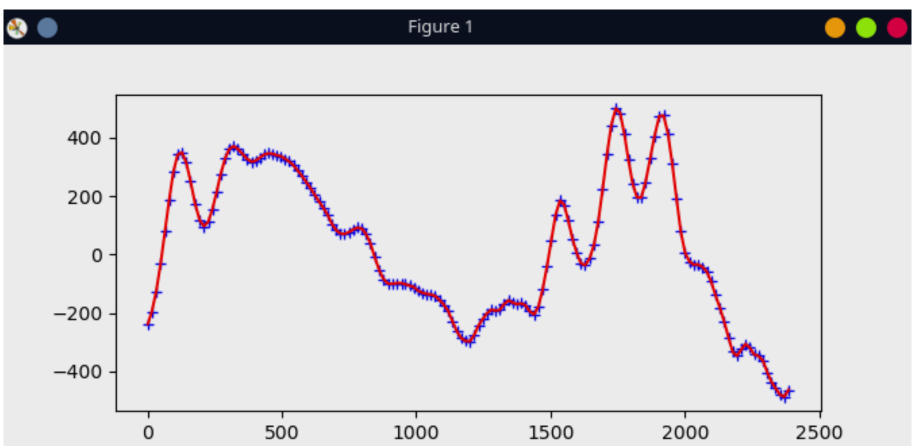
```
f = open('./data/' + rand() + '.csv', 'w');
f.write('Time,Vec,Wfm\n' + data.replace('  ', ',').replace('   ', ',').replace('    ', ',').replace(',,', ','));
f.close()
```

Data is saved onto a CSV file.

```
plt.clf()
plt.plot(dataArray, waveformArray, 'b+')
plt.plot(dataArray, waveformArray, 'r')
```

plot the data using PyPlot.

Below is a graph of a waveform that the code generated.



Import Required Libraries

Self-explainatory, imports the decision trees that we need, and Pandas, a data library.

In [ ]:
```
import tensorflow_decision_forests as tfdf
import tensorflow as tf
from tensorflow_decision_forests.keras import core
from datetime import date
import pandas as pd
```

Add training data, convert data to TF Dataset alongside specifying Data Labels.

Simple, We just load the data as a Pandas DataFrame, and then convert said DataFrame into a TF Dataset.

-- Future Improvements --

-Currently there is not need to worry about the data size since the training data is less than 1MB. Once the data reaches >512kb, consideration in reworking data loading must be taken in account so it loads chuncks of data all at once instead.

In [ ]:
```
# Load a dataset in a Pandas dataframe.
train_df = pd.read_csv("data/train.csv")
test_df = pd.read_csv("data/test.csv")

# Convert the dataset into a TensorFlow dataset.
train_ds = tfdf.keras.pd_dataframe_to_tf_dataset(train_df, label="Balloon")
test_ds = tfdf.keras.pd_dataframe_to_tf_dataset(test_df, label="Balloon")
```

Second (Final) Model.

The Second Model focuses on the first model's weaknesses, it uses GradientBoosted Trees to improve training Performance and Accuracy.

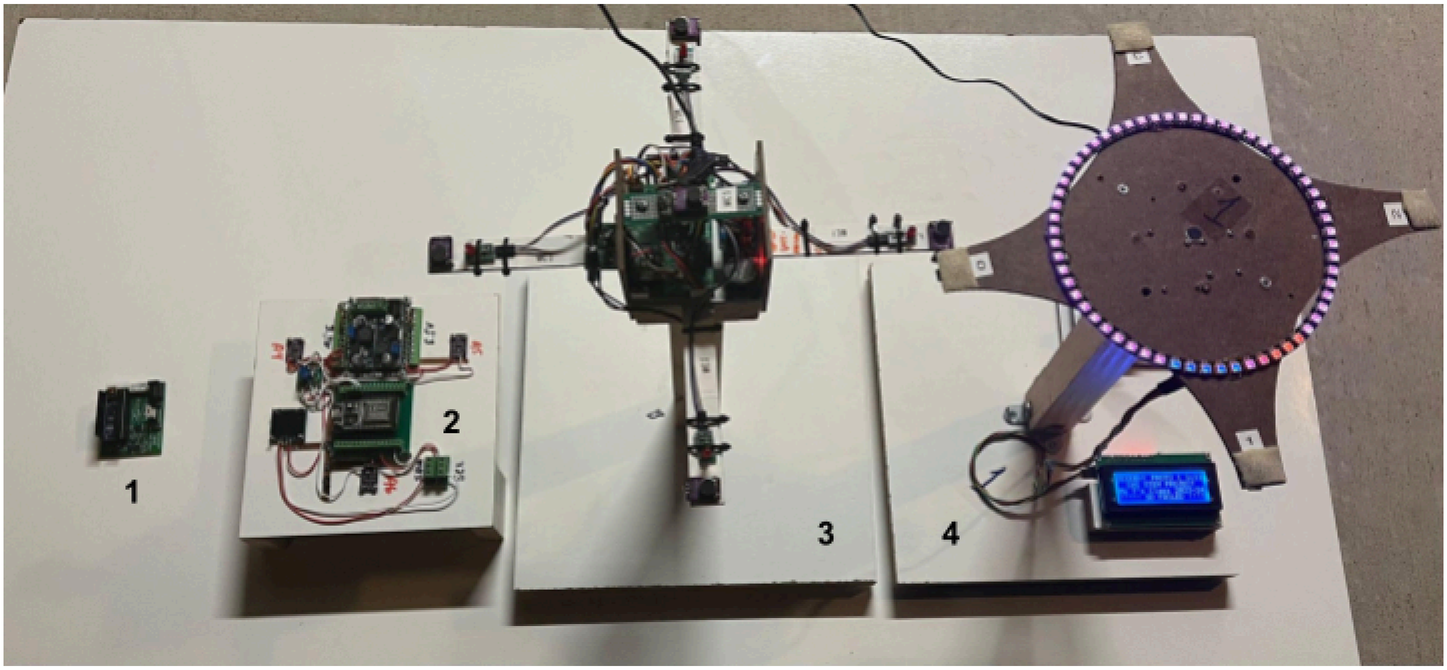
In [ ]:
```
# Configure the tuner.
tuner = tfdf.tuner.RandomSearch(num_trials=1_000_000)
tuner.choice("num_candidate_attributes_ratio", [1.0, 0.95, 0.9])
tuner.choice("use_hessian_gain", [True, False])

tuner.choice("growing_strategy", ["BEST_FIRST_GLOBAL"])
tuner.choice("max_num_nodes", [16, 32, 64, 128, 256, 512, 1024])


model2 = tfdf.keras.GradientBoostedTreesModel(
    task=core.Task.CLASSIFICATION,
    tuner=tuner,
    max_depth=100
)

model2.fit(train_ds)
model2.compile(metrics=["accuracy",tf.keras.metrics.Precision(),tf.keras.metrics.Recall()])
```

Prototypes



The project has gone through four different prototypes:

1. SCOOBY originally was on a Arduino Nano board, but it did not have enough memory or speed to complete calculations needed.
2. The second prototype was based on an ESP32. This processor was much faster and had enough memory, but the footprint was too big to fit on the Scooby board.
3. The first full SCOOBY's structure was built with directional microphones out on 'wings' that gave the purest waveform data (as tested against a laboratory grade microphone). This prototype worked, but after many modifications to add features (LCD, Ring Led display, Com port) the unit became unreliable due to constant rework.
4. The fourth prototype was a cleaned up version with minor modifications to the circuit board, a new one piece baffle for the microphones, a box to contain the PCB and wires, and windscreen added over microphones. The new version of the circuit board accommodates the LCD screen (added for more information) and the LED ring (added for easy directional display of the results).

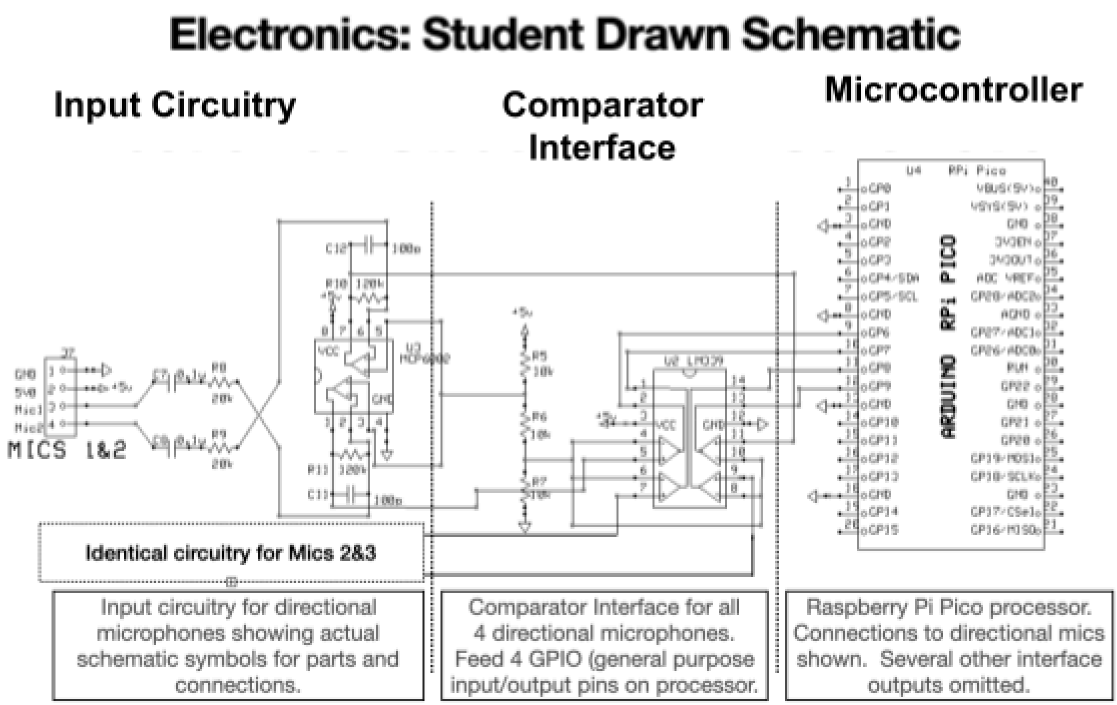
Cost

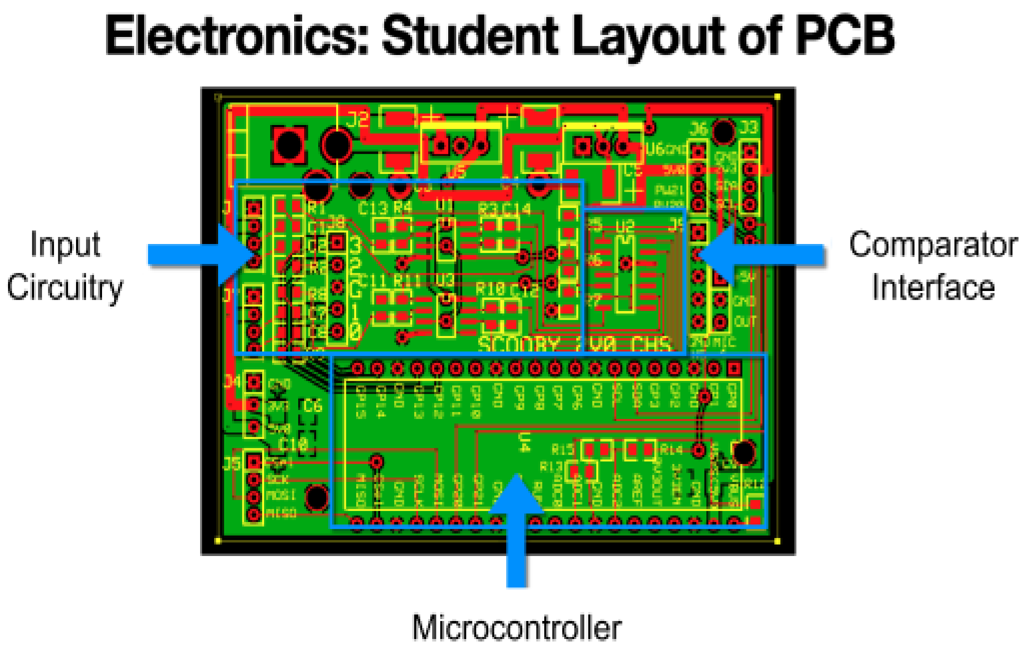
Electronics	Number Used	Cost each	Total Cost
Mic	15	$1.30	$19.50
SD Drive	3	$6.50	$19.50
Power Input	3	$1.10	$3.30
Regulators	6	$0.65	$3.90
Misc Resistors & Caps	12	$1.24	$14.94
Amplifiers	6	$0.70	$4.20
Comparators	3	$0.50	$1.50
Ring LEDs	3	$12.99	$38.97
LCD Display	3	$9.99	$29.97
Chassis	3	$2.25	$6.75
Real Time Clock	3	$2.38	$7.14
RPi Controller	3	$6.66	$19.98
Printed Circuit Board	3	$35.00	$105
Batteries	3	$1.85	$11.10
Battery Holder	3	$2.25	$6.75
Total Cost: $495.00			

The purpose of this project is to build units as cheaply as possible to keep the cost down for schools so they could implement a system of multiple units of SCOOBY. For this reason, the following inexpensive components were obtained from low cost distributors such as Adafruit, PCBWay, Elegoo, and others.

Circuitry

Inside of SCOOBY is a circuit board that the team designed during the summer of 2023. There are three main parts: the input circuitry, the comparator interface, and the microcontroller. The schematic was used as an input to the circuit board layout tool.





Test Model and Evaluation

Safety and Protocol

The testing process included popping large balloons that produced a loud sound that could potentially damage hearing with repeated exposure. To prevent hearing loss the team used ear plugs and stood 20 feet away as an automated system inflated and burst the balloon (BBB and CHAMP). Separate team members were responsible for setting up the layout, managing the automated balloon system and recording data. After each test the SD drive was checked for data integrity.

Testing

The system was tested by popping many balloons in a school's courtyard, hallways and classrooms. At first, the balloons were pumped up manually and then popped with a sharp object such as a screw. Afterwards, the team invested in an automated system which filled the balloons up with air until they blew up. Balloons were used as a substitute for the sounds produced by an actual gun. This is not only for safety reasons, but also because balloons produce a similar decibel range. There was a study done on this subject by University of Alberta Researchers. Heuser Hearing Institute wrote the article “Hearing Loss from Balloons?” using the data from gunshots and balloons inflated until they burst (as done in our tests). The sounds are within 1dB (~10%) of each other.

SCOOBY was placed in the middle of the courtyard. Two team members were responsible for popping the balloons. The tests were conducted at various distances (0-120ft). All the outcomes of each experiment displayed by SCOOBY were read and recorded, and the records of data on the SD card were verified.

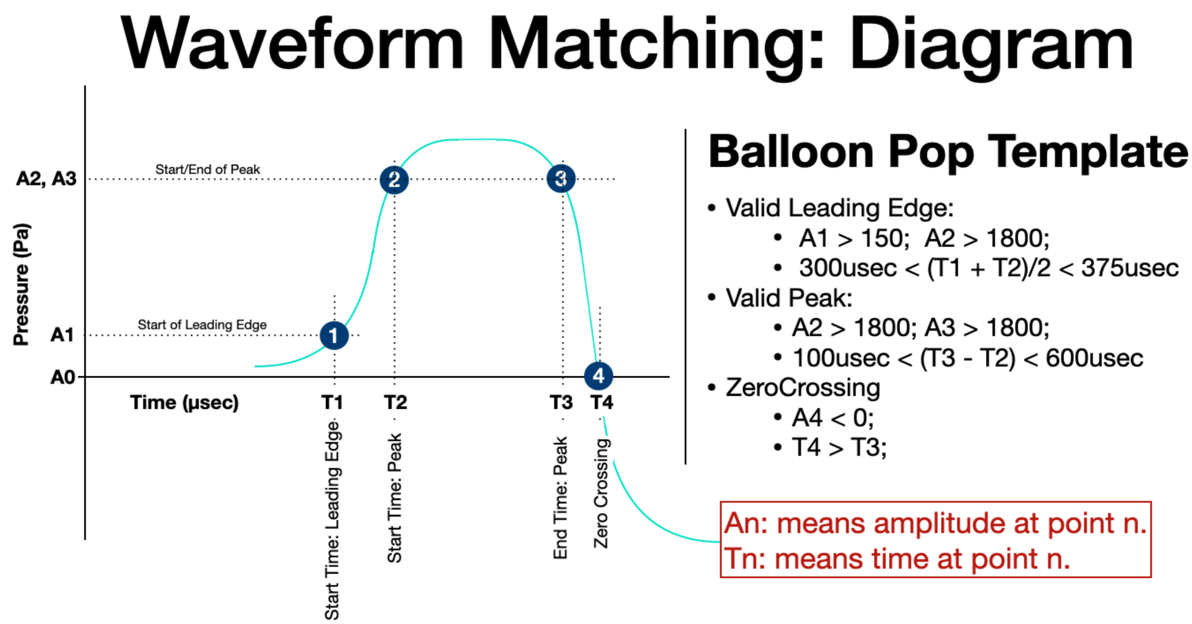
While SCOOBY was being tested, a hypothesis about the balloons was formed. As observed, the balloons dyed in darker colors, such as green and red, took longer to expand. When they finally filled up, they produced a louder sound. Throughout the experiments one thing was mostly consistent. After they “popped”, the way the balloons tore remained relatively similar. It mostly produced two main pieces. One of these pieces consisted of strands of rubber similar to strings. The other was the neck with strands of rubber (the same as in the other piece).

Data Collection

Data collected by SCOOBY was stored into an SD in a TXT file. The file contained three columns of data: Time (µsec), Mic Flags, and Amplitude. The raw data was then imported into a Google Sheet for the purpose of data cleaning and their graphical representation.

Test Results

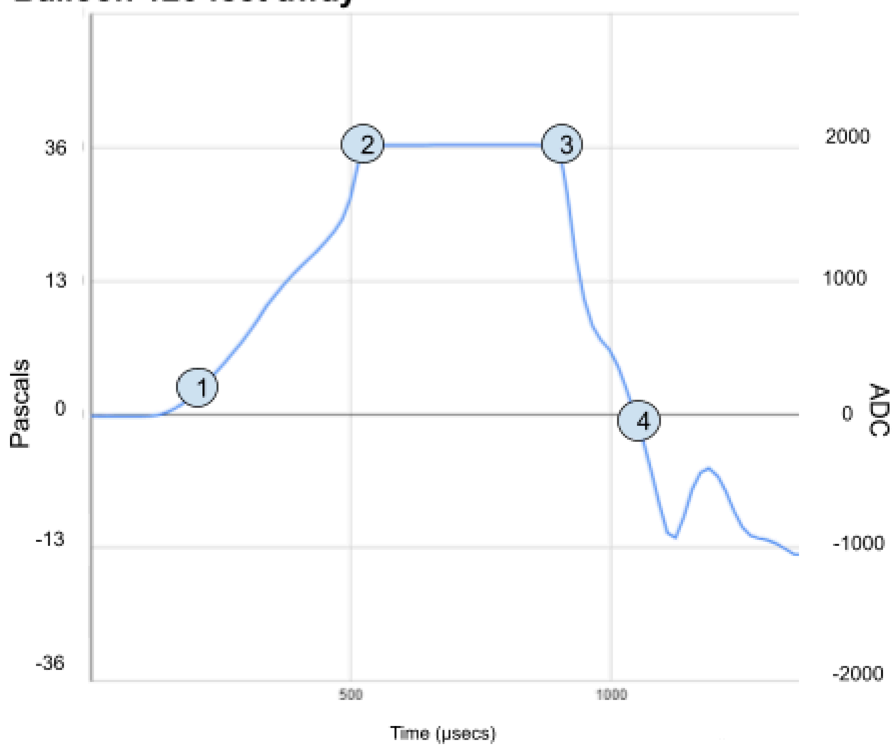
There are 4 points along the waveform that best characterize the waveform. The points are identified by their amplitude and the time at which that amplitude occurred. For example, point 1 is the first time that the waveform exceeds 150, and point 2 is the first time it exceeds 1800. These limits were tested against the collected data so that actual balloon bursts passed the test, but other sounds like door slams, chair falling, book falling, etc did not.



SCOOBY determines four key characteristics of the waveform: leading edge time, peak, peak duration, and the zero crossing time. Between points 1 and 2 is the leading edge of the soundwave, this is calculated with the average times for point 1 and point 2. It must be between 300 µsecs and 375 µsecs. The peak is located at points 2 and 3, the peak must last between 100 µsecs and 600 µsecs. The zero crossing is when the sound has first negative pressure and is at point 4.

After analyzing the data, the conclusion is that as the distance increases the waveform spreads out and gets wider but at extreme distances, the amplitude starts to fall and makes the pulse appear shorter. Down below are examples of data collected:

Balloon 120 feet away

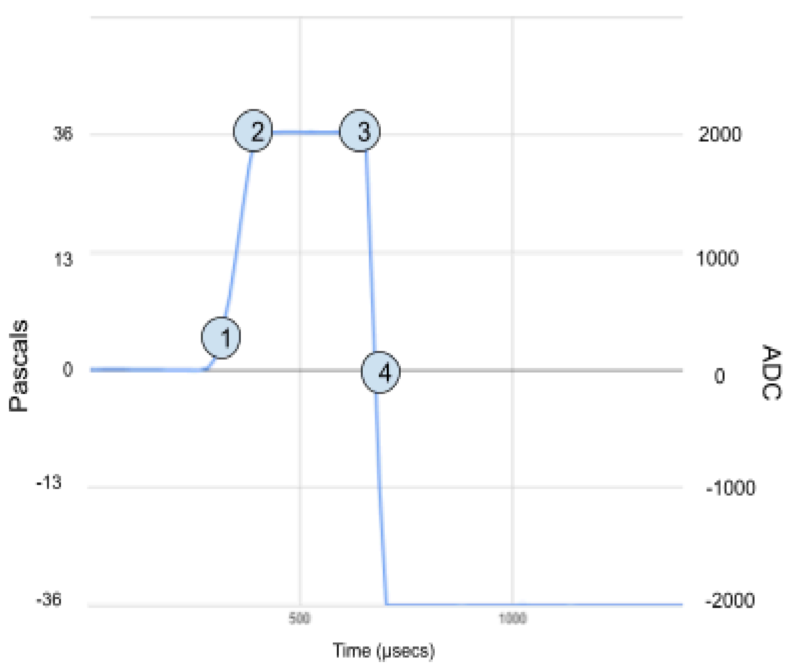


Valid Leading Edge:
A1 = 192
T1= 210 usecs
A2 = 1894
T2 = 515 usecs
300 usecs < (210 usecs + 515 usecs)/2 < 600 usecs
300 usecs < 362 usecs < 600 usecs

Valid Peak:
A2 = 1894
T2 = 515 usecs
A3 = 1661
T3 = 915 usecs
100 usecs < 915 usecs - 515 usecs < 700 usecs
100 usecs < 400 usecs < 700 usecs

Zero crossing:
A4 = -189
T4 = 1061 usecs
-189 < 0
T4 > T3

Balloon 20 feet away



Valid Leading Edge:
A1 = 304
T1= 316 usecs
A2 = 2020
T2 = 398 usecs
300 usecs < (398 usecs + 316 usecs)/2 < 600 usecs
300 usecs < 357 usecs < 600 usecs

Valid Peak:
A2 = 2020
T2 = 398 usecs
A3 = 551
T3 = 672 usecs
100 usecs < 672 usecs - 398 usecs < 700 usecs
100 usecs < 274 usecs < 700 usecs

Zero crossing:
A4 = -987
T4 = 688 usecs
-987 < 0
T4 > T3

Triangulation Testing

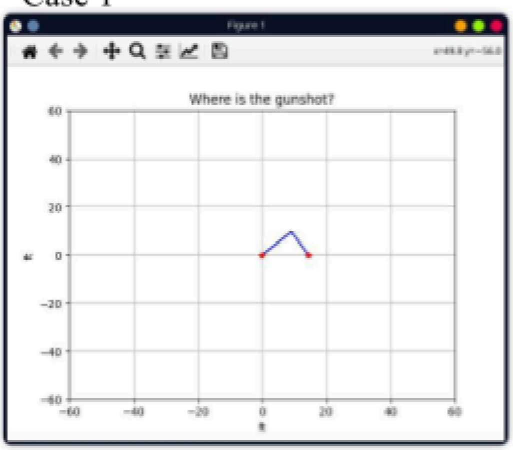
Some functional testing was done on triangulation. The goal of these tests was to see if triangulation is possible with two SCOOBY units connected to a computer running Linux.

There was three tests cases were done for the functional testing:

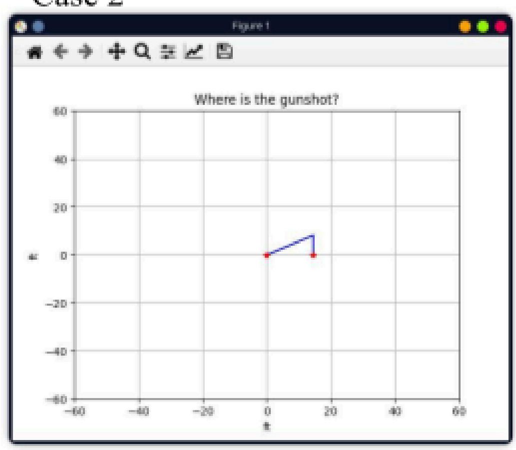
- Case 1: Balloon was popped near between both units, SCOOBY 1's angle was about 45 degrees and SCOOBY 2's angle was about 135 degrees
- Case 2: Balloon was popped near SCOOBY 2, SCOOBY 1's angle was about 45 degrees and SCOOBY 2's angle was about 90
- Case 3: Balloon was popped near SCOOBY 1, SCOOBY 1's angle was about 90 degrees and SCOOBY 2's angle was about 135.

The screenshots produced below are the results of each test case:

Case 1



Case 2



Case 3

Triangulation with two units is in its preliminary stages, it is important to further test its limit to identify possible errors in calculations.

Strengths

Some strengths that SCOOBY has is that it does accurate calculation of the originating point of a loud sound relative to its microphones. With the use of two units, it will be able to triangulate a loud sound within a room. Also it accurately recognizes a balloon pop, implicating recognition of gunshots in a real event simulation. Each unit of SCOOBY is cost effective and is simple to assemble. Finally it completes all of its completion of all calculations in microseconds (runs 127 thousand times per second). Meaning that in the future it will be able to detect consecutive gunshots.

Weaknesses

Some weaknesses identified while testing is that with the current configuration visually impaired or hearing impaired might not notice device triggering as there is no loud alarm or bright light. There is also a possible vulnerability of false positive detection. More data must be collected to minimize false positives. Also many guns in the market have sound proofing modifications, this may lead to them not being detected by SCOOBY. Only one unit can tell the direction , not the exact location. For triangulation to work there must be at least two units. At this time SCOOBY can’t tell the difference between balloon pops and gunshots (due to safety reasons, SCOOBY has not been tested with actual gunshots). Also SCOOBY currently depends on electricity, meaning if its batteries die the unit is not functional.

Future plans

The future plan with SCOOBY is to test it in its intended configuration. Which is attached to a ceiling panel. Also to help hearing impaired people to know that SCOOBY was triggered, a bright display will be added to inform them. While for the visually impaired a distinct sound will be used to inform them of a trigger. Currently SCOOBY can have a filter so it can only trigger when a balloon is being popped but sometimes it gets triggered with a loud sound similar to a gunshot. To help SCOOBY different balloon pops, gunshots or loud sounds (such as balloon pops, door slams, students dropping items, etc.), it is planned to use a Machine Learning algorithm to differentiate this, preferably using low-cost hardware such as a Raspberry Pi. To be able to save the data that the system collects, a unit will be connected to a central hub via Bluetooth or wired connection. In addition, the hub would have multiple units connected to be able to triangulate sounds to get much more accurate results on where gunshots came from.

Collaboration

The team met regularly on Tuesdays and Fridays to progress on the project. On Tuesdays the team worked on reports/presentations for the competitions. On Fridays the team worked on the project by: collecting and analyzing data, brainstorming new ideas, and gaining experience with the system. During the meetings, the team met with the engineering mentor who guided the development of the project.

Bibliography

"2015 Umpqua Community College Shooting." *Wikipedia*, Wikimedia Foundation, 7 Nov. 2023,en.wikipedia.org/wiki/2015_Umpqua_Community_College_shooting. Date Accessed: November 14 2023

"2018 Santa Fe High School Shooting." *Wikipedia*, Wikimedia Foundation, 4 Nov. 2023, en.wikipedia.org/wiki/2018_Santa_Fe_High_School_shooting. Date Accessed: November 14 2023

Chapman, Isabelle, et al. "Uvalde School Shooter Was in School for up to an Hour before Law Enforcement Broke into Room Where He Was Barricaded and Killed Him." *CNN*, Cable News Network, 26 May 2022, www.cnn.com/2022/05/25/us/uvalde-texas-elementary-school-shooting-wednesday/index.html. Date Accessed: November 14 2023

"Charts & Graphs." *CHDS School Shooting Safety Compendium*, 2 Mar. 2023, www.chds.us/sssc/charts-graphs/. Date Accessed: November 2 2023

"Columbine High School Shootings." *Encyclopædia Britannica*, Encyclopædia Britannica, inc., 18 Sept. 2023, www.britannica.com/event/Columbine-High-School-shootings . Date Accessed: November 14 2023

Craig, Gary. "Rochester Man Shot by Police Sues Cops, City, and Shotspotter." *Democrat and Chronicle*, Democrat and Chronicle, 29 Mar. 2019, www.democratandchronicle.com/story/news/2018/08/30/silvon-simmons-rochester-police-officer-joseph-ferrigno-gun-lawsuit/1119014002/. Date Accessed: November 14 2023

"Estimating Room Impulse Responses from Recorded Balloon Pops", Audio Engineering Society, Conf paper, November 4-7 2010, Jonathan S. Abel et al. Date Accessed: July 21, 2023

"Gun Shot Detection." *Police Surveillance in Chicago*, chicagopolicesurveillance.com/tactics/gun-shot-detectors.html. Date Accessed: October 3 2023

Hauser, Christine, and Anahad O'connor. "Virginia Tech Shooting Leaves 33 Dead." *The New York Times*, The New York Times, 16 Apr. 2007, www.nytimes.com/2007/04/16/us/16cnd-shooting.html. Date Accessed: November 14 2023

Fuel-Admin. "Hearing Loss from Balloons?: Heuser Hearing Institute: Blog." *Heuser Hearing Institute*, 29 Dec. 2017, thehearinginstitute.org/hearing-loss-balloons/. Date Accessed: December 2 2023

Radio, Minnesota Public. "MPR: What Happened at Red Lake?" *News & Features*, 22 Mar. 2005, news.minnesota.publicradio.org/projects/2005/03/redlake/. Date Accessed: November 14 2023

"Sandy Hook School Shooting." *History.Com*, A&E Television Networks, www.history.com/this-day-in-history/gunman-kills-students-and-adults-at-newtown-connecticut-elementary-school. Date Accessed: November 14 2023

"Teen Gunman Kills 17, Injures 17 at Parkland, Florida High School | February 14, 2018." *History.Com*, A&E Television Networks, www.history.com/this-day-in-history/parkland-marjory-stoneman-douglas-school-shooting. Date Accessed: November 15 2023.

"Texas Tower Shooting of 1966." *Encyclopædia Britannica*, Encyclopædia Britannica, inc., 18 Sept. 2023, www.britannica.com/event/Texas-Tower-shooting-of-1966. Date Accessed: November 14 2023

"The Long, Shameful List of Gunfire on School Grounds in America." *Everytown Research & Policy*, 28 Feb. 2023, everytownresearch.org/maps/gunfire-on-school-grounds/. Date Accessed: November 15 2023

“Variations in recorded acoustic gunshot waveforms generated by small firearms”, Acoustical Society of America, January 24 2011, Steven D. Beck et al. Date Accessed: July 20 2023