

Assignment 2 report

Arya Javani 2402679

Introduction

In this report I will explain the steps taken to do the second assignment along with figures.

Microphone setup

For the microphone I installed droid camera but when I ran the program, I realized that my laptop already has two mics.

Showing the volume

For this part only one line of code had to be changed.

```
label.setText(str(index)+ f" : {volume}" )
```

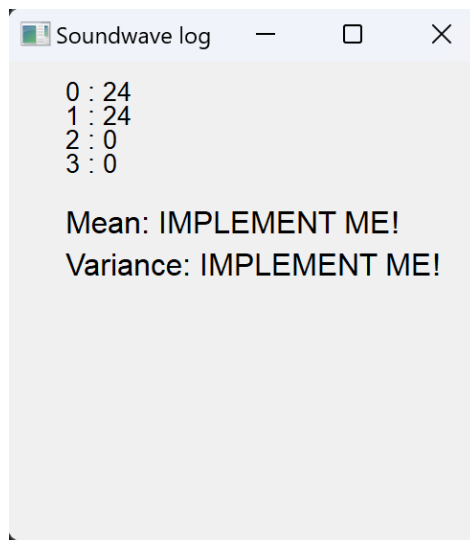
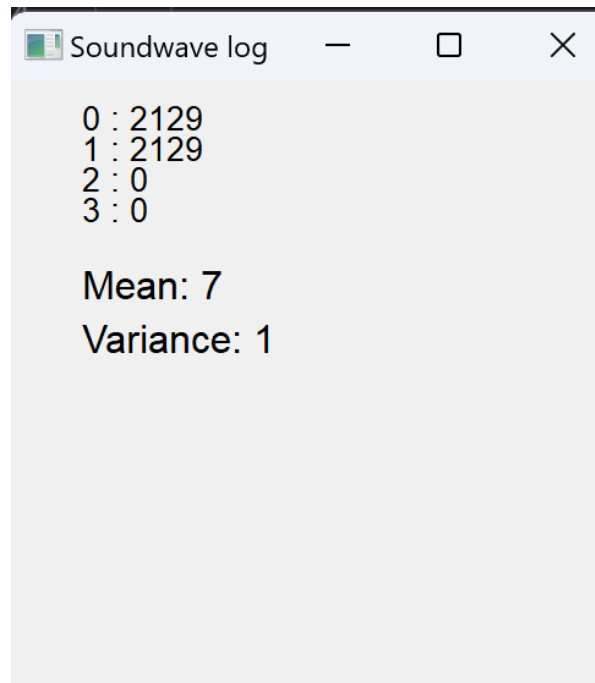


Figure 1: The first step, showing the volume captured by microphones

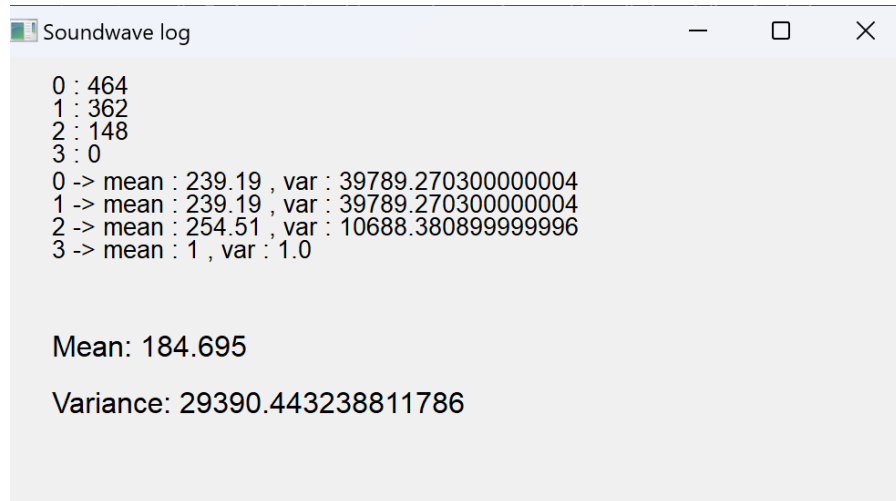
Overall mean and variance

For this part, set the text for mean and variance label in the volume loop. I also implemented the functions for calculating the variance and meaning.



Measuring the volumes for each sound source

In this part, I defined new labels for each device mean and variance. Then changed the buffer size to 100 and also defined a new set of mean and variance version for 1D lists and this is the result. (I got the mobile source to be working but there was an output that wasn't connected to anywhere but was affecting the mean and variance that was why the mean and variance were lower than what they should be). It is also worth mentioning that there was a piece of very strange code in which the main thread was manipulating the buffer. I'm not sure what happens on the low level but there's a high possibility that there will be a data race here between main and devices threads so I moved buffer modification to devices.



```
Soundwave log

0 : 464
1 : 362
2 : 148
3 : 0
0 -> mean : 239.19 , var : 39789.270300000004
1 -> mean : 239.19 , var : 39789.270300000004
2 -> mean : 254.51 , var : 10688.380899999996
3 -> mean : 1 , var : 1.0

Mean: 184.695

Variance: 29390.443238811786
```

Detecting the faulty microphone

For this part, we assume that a faulty microphone is one that doesn't capture output properly, so its input is either zero or significantly lower than other output. For checking this, we will use the overall variance and divide it by 200 and see if any of the devices still has lower value and identify it as faulty. (this is just a naïve detection as the microphone might have low variance because of a constant input, as I was testing values on songs this approach works, but for a general solution, many scenarios and inputs should be considered there might be so many scenarios that we will need a machine learning model to capture all the patterns)

In this part, first from my computer settings I closed the not working virtual input to help the values be more accurate. For testing, since I didn't have any extra device, I decided to keep the computer inputs as two values and cover my microphone to simulate a faulty microphone. (instead of covering my microphone, I adjusted the input level from droid app and reduced it to a very low level)

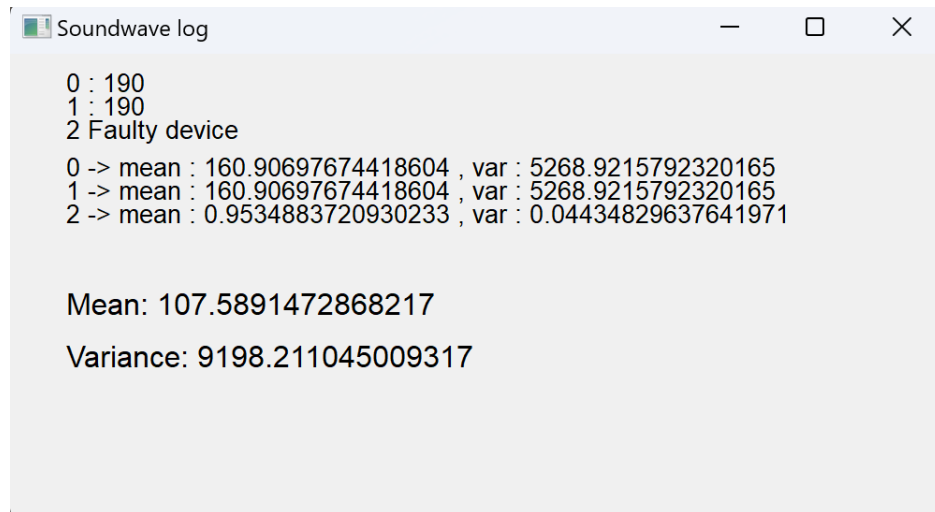


Figure 2: You can see that the mean and variance are very small because of the weak input that simulated a faulty device

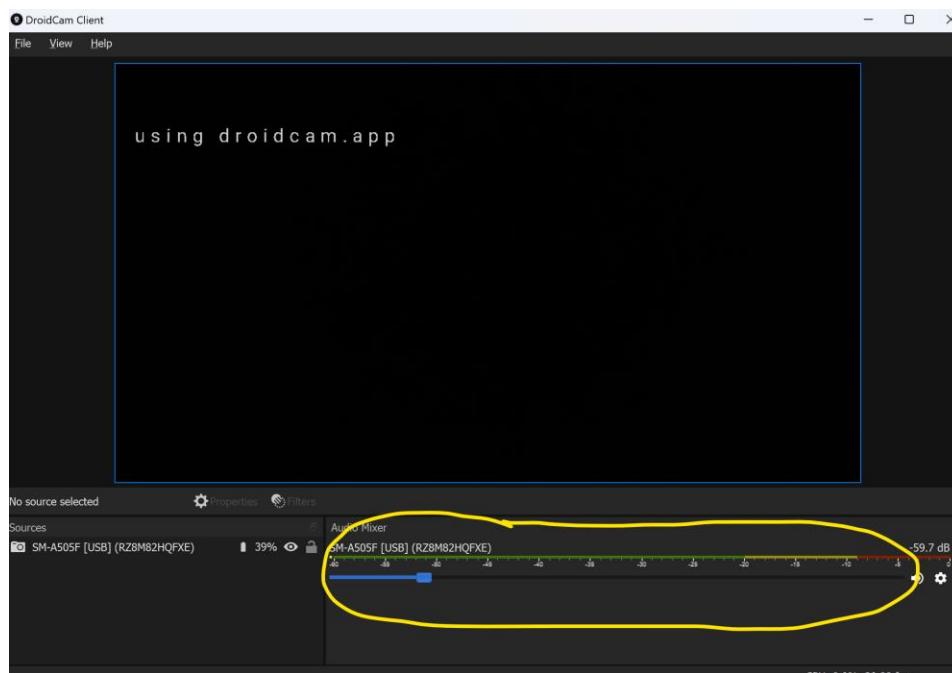


Figure 3: in this image it is shown that we limited the input level for the microphone to simulate faulte situation