

National University Of Computer and Emerging

Sciences



User Interfacing from

Tapping Motion on a Surface

(Final Report – FYP II)

Group Members:

|  |  |
| --- | --- |
| Rahim Shahid | L14-4137 |
| Omer Haroon | L14-4119 |
| Saad Mazhar | L14-4151 |
| Waheed Amir | L14-4135 |

**Project Advisor:**

Prof. Zeeshan Ali Rana

**Project Co-Advisor:**

Prof. Mubasher Baig

Department of Computer Science

FAST NU, Lahore, Pakistan

Anti-Plagiarism Declaration

This is to declare that the above publication produced under the:

**Title: User Interfacing from Tapping Motion on a Surface**

is the sole contribution of the authors and no part hereof has been reproduced on **as it is** basis (cut and paste) which can be considered as **Plagiarism.** All referenced parts have been used to argue the idea and have been cited properly. We will be responsible and liable for any consequence if violation of this declaration is determined.

Date: \_\_\_\_\_\_\_\_\_\_\_\_\_\_ Student 1

Name: \_\_\_\_Rahim Shahid\_\_\_\_\_\_

Signature: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Student 2

Name: \_\_\_\_Omer Haroon\_\_\_\_\_\_

Signature: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Student 3

Name: \_\_\_\_Saad Mazhar\_\_\_\_\_\_\_

Signature: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Student 4

Name: \_\_\_\_Waheed Amir\_\_\_\_\_\_

Signature: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Table of Contents

[Chapter 1: Introduction 8](#_Toc515318198)

[1.1 Overview 8](#_Toc515318199)

[1.2 Goals and Objectives 8](#_Toc515318200)

[1.3 Project Scope 9](#_Toc515318201)

[1.4 Problem Elaboration 9](#_Toc515318202)

[1.5 Document Outline 10](#_Toc515318203)

[Chapter 2: Literature survey / related work 11](#_Toc515318204)

[2.1 Technology Overview 11](#_Toc515318205)

[2.2 Techniques Used For Acoustic Localization 11](#_Toc515318206)

[2.2.1 TDOA 11](#_Toc515318207)

[2.2.2 Template Matching 12](#_Toc515318208)

[2.2.3 Acoustic Holography 13](#_Toc515318209)

[2.2.4 Vision 13](#_Toc515318210)

[2.3 Applications 14](#_Toc515318211)

[2.3.1 Virtual Keyboard 14](#_Toc515318212)

[2.3.2 Musical Instrument 14](#_Toc515318213)

[2.4 Literature Review Summary 14](#_Toc515318214)

[Chapter 3: Requirements and design 16](#_Toc515318215)

[3.1 Requirements 16](#_Toc515318216)

[3.1.1 Functional Requirements 16](#_Toc515318217)

[3.1.2 Non Functional Requirements 16](#_Toc515318218)

[3.2 Design 16](#_Toc515318219)

[3.2.1 Training 17](#_Toc515318220)

[3.2.2 Evaluation 18](#_Toc515318221)

[Chapter 4: Implementation & Test Cases 20](#_Toc515318222)

[4.1 Prototype Workflow 20](#_Toc515318223)

[4.2 Sensors Used 21](#_Toc515318224)

[4.3 Detail of Features 21](#_Toc515318225)

[4.3.1 Feature Extraction Process 22](#_Toc515318226)

[4.4 A.I. Models 24](#_Toc515318227)

[4.4.1 Decision Trees 24](#_Toc515318228)

[4.4.2 Random Forest 24](#_Toc515318229)

[4.4.3 Neural Network 24](#_Toc515318230)

[4.5 Test Cases 25](#_Toc515318231)

[4.6 Bug Report 28](#_Toc515318232)

[4.6.1 Single Channel Microphone Bug 28](#_Toc515318233)

[4.7 Conclusion 30](#_Toc515318234)

[Chapter 5: Experimental results and analysis 31](#_Toc515318235)

[5.1 Accelerometer Results 31](#_Toc515318236)

[5.2 Microphone Results 31](#_Toc515318237)

[5.3 Conclusion 32](#_Toc515318238)

[Chapter 6: Conclusions 33](#_Toc515318239)

[6.1 Challenges & Limitations 33](#_Toc515318240)

[6.2 Accelerometer 34](#_Toc515318241)

[6.3 Microphone 34](#_Toc515318242)

[6.4 Physical Location of Sensors 34](#_Toc515318243)

[6.5 Portability 34](#_Toc515318244)

[6.7 Future Work 34](#_Toc515318245)

[References 35](#_Toc515318246)

List of Figures

[Figure 1: TDOA working principle. Multiple sensors surrounding the tap window 11](#_Toc515316343)

[Figure 2: TM approach. Single sensor, Source [5] 12](#_Toc515316344)

[Figure 3: Training 16](#_Toc515316345)

[Figure 4: Evaluation 17](#_Toc515316346)

[Figure 5: A complex time-domain signal Source [3] 21](#_Toc515316347)

Figure 6: Magnitude spectrum of time-domain signal plotted above Source [3]…………..23

Figure 7: More filters in low frequency bands: A way to eliminate unnecessary information from signals Source [4]………………………………………………………………………23

**List of Tables**

[Table 1: Literature Review 15](#_Toc515317597)

Table 2: Sensor Analysis……………………………………………………………………..32

Abstract

The purpose of this report is to document the findings of our final year project: User interfacing by tapping motion on a surface. In our project we aim to create a new user interfacing system for smartphones in which users can give input commands to their phones by tapping on the resting surface of the phone. The phone can then discern certain properties of the wave, such as its intensity and the direction of the source to give inputs to an application. To accomplish this, we will make use of the accelerometer and microphones housed inside the body of the phone to detect disturbances near the phone, we will then collect the wave data of these disturbances from the sensors and train an A.I model using wave features extracted from this data and then compare subsequently generated taps on the phone in real time with this model in order to tell which direction the disturbance has originated from relative to the phone e.g. left, right, up or down etc.

After training our A.I models, we learned that wave features in the spectral domain such as spectral centroid, variability and skew are most useful in detecting the direction of taps. Several A.I models were used from which Decision Trees and Random Forests gave the highest accuracy of up to 90% and Artificial Neural Networks giving an accuracy of 76%. The project is limited by the non-portability among different kinds surfaces and requires vast amount of data to resolve this. However narrowing it down to a few surfaces and a few make of smartphones, we can successfully demonstrate the project.

# Chapter 1: Introduction

This section will provide a brief overview of the nature and purpose of our project as well define the goals, objectives and scope of the project.

## Overview

Today there are almost as many cell-phone subscriptions as there are people on the Earth and 70% are tied to smartphones. Smartphones are part of the day to day routine of billions of people around the globe and most of them utilize touchscreens for input. A lot of progress has been made in the realm of entering input into these devices by using touchscreens or through audio using voice commands. But relatively little thought has gone into using acoustic signals and vibrations generated by tapping on the resting surface (the surface the device is places on) to provide input to one’s phone, tablet or any other kind of device.

Thus the purpose of this project is to further research this relatively novel form of input and create an interfacing system that gives users the ability to generate acoustic signals through the environment to present as input to their device, or more simply put, it allows them to give inputs to their devices by tapping and creating vibrations on the surface that their device is resting on. This kind of input has many characteristics such as intensity of vibrations, the location and direction of the source of vibrations and distance of the source relative to the device etc. which can be interpreted to give commands to the device.

To achieve this, we will perform acoustic source localization using hardware that can be found easily on most modern phones that are commercially available to the general public i.e. an accelerometer and microphones.

This kind of interfacing mechanism will provide a wide range of possibilities for app development in the future and help drive innovation in many different kinds of applications including video games and interfacing systems and much more, the only limitations are the imaginations of the developers and what kind of uses they can come up with.

## 1.2 Goals and Objectives

The ultimate goal of the project is to create an interfacing mechanism which uses acoustic vibrations generated using the environment to give commands to a device, this can be divided into the following major sub goals:

* **Detect taps accurately**

Use the accelerometer to determine that the acoustic vibrations being detected by the system were actual “taps” made by the user on the surface on which the device is currently resting.

* **Acoustic Source Localization**

Determine the location of the source of the acoustic vibrations and disturbances created by the taps using readings from the device’s accelerometer. The microphones present on the device can also be used to aide in determining the direction.

* **Create basic library for use in development**

Create a small and basic library of code that can be used to do further research and develop simple apps using the system.

## Project Scope

* **System will be able to detect the direction of the source of vibrations**

The system should be able to determine the direction of the source of vibrations. For example, if the source is on the left of the device or on the right.

* **Demo the system**

We will demonstrate the basic functionality of the system using very simple apps or software that will use the system to interface with users.

## Problem Elaboration

There are two distinct problems that we need to solve. Detection of a Tap and Identification of its various Parameters such as direction, distance etc.

1. **Detection of Taps**

First problem we have at hand is detecting when a tap has occurred. Detection problem is precisely defined as:

***Tell whenever a user deliberately strikes the surface in the vicinity of the device with his finger tips to invoke some kind of behavior.***

Only then will the tap be considered for further processing. This is different from other user actions that also cause vibrations in the phone but are not intended to control the application e.g. picking some object off the table, or moving the elbow rest. The system should be able to effectively filter out these irrelevant vibrations and only pass those that resulted from actual finger taps.

1. **Identification of important Parameters**

Detection step tells you “when” a tap occurred. But we also need to answer the following questions about the tap in order to make a useful system:

1. Where did the tap come from (Direction)?
2. How far did the tap come from (Distance)?
3. How hard did the user strike the surface (Intensity)?

Distance is considered from the exact center of the rectangular area of the phone and angle is considered to be anti-clockwise. Intensity is yet to be defined.

Now there is an issue with this above description. We do not know what level of accuracy should be ensured by the system. In fact, at this stage we cannot really tell how much accuracy is even achievable with contemporary devices. Therefore, we have to describe this second problem more formally and precisely in terms of error measures. We define three metrics as Edist, Edir and Eint representing maximum errors in the calculations of distance, direction and intensity of a tap respectively. Now, we describe this problem more accurately as:

***Minimize the error metrics associated with the calculation of distance, direction and intensity of a tap.***

Viewing this problem as a minimization problem greatly simplifies our work. Instead of attempting to find the desired properties directly from a wave, we can start with some very trivial values for the above mentioned errors and progressively refine our system to minimize them. For example in the calculation of direction of a tap, initially we will strive for Edir to be at most 90 degrees i.e. the system should be able to tell only cardinal directions of the source of a tap ( whether it has come from left, right, up or down side of the phone ). If we succeed at this, then we will try to achieve Edir to be 45 degrees making the system capable of predicting extended directions (left, down, left-down and so on). This way we will continue refining errors till we have achieved a reasonable accuracy or hit the device bottleneck. Same goes for distance and Intensity.

## Document Outline

In chapter 2 we will mention all the relevant research previously conducted on the subject of acoustic source localization that we could and how it will be used in our system. Chapter 3 gives a quick overview of the system’s function specifications and design. Chapter 4 goes into the details of our implementation of the system so far and test cases used for validation. In Chapter 6 we present the conclusion and summary of the work done during our project.

# Chapter 2: Literature survey / related work

This sections provides a detailed review of the literature we studied to find a solution for our problem. We discuss the principles that can be used to locate taps on a surface by using either their acoustic properties or by computer vision. Different methods are discussed along with their advantages, disadvantages and suitability for our project. At the end some applications of these techniques are also discussed.

## Technology Overview

Our project revolves around detecting and catching vibrations that are propagating in a surface as a result of a strike or tap by the user. When physical contact is made with a solid surface (wall, table, etc.) the surfaces vibration pattern will change according to how energy distributes on the surfaces. Such a change can be caused by the energy generated at the point of contact when the objects surface is tapped. In majority of materials vibration proliferate easily so it insures that the data about the interaction of user’s finger with the surface can be conveyed to a foreign location. This will only utilize the materials structure itself as the transmission medium, smothering the requirement for any other layer or some nosy gadget over the range that is to be made touch sensitive.

Tangible Acoustic Interface technology is becoming advanced in deciding the location of vibration sources which originate as a result of user interaction with a surface. They are able to extract extra data from this interaction as well which portrays additional information e.g. nature of strike, properties of the medium. This data helps filter out the false positives. It will along these lines be conceivable to decide how and where the surface is touched or strike giving a total portrayal of the interaction. The following section give an outline of the different approaches used for acoustic source localization.

## Techniques Used For Acoustic Localization

Our study of different papers published in our relevant domain revealed a number of techniques that are used to tackle problems like ours. We found these techniques to be most commonly and effectively used for the purpose of acoustic vibration localization. Furthermore we also critically analyze another popular method incorporated to achieve our results, computer vision.

### TDOA

**Time Delay of Arrival technique** has close ties in the research area of acoustic source localization. The principal of this technique is to measure the time of arrival of a wave at different locations and thus estimate its point of origin. The sensors are placed at predetermined position, ideally surrounding the surface on which the waves are to be located. The following figure gives the general idea of this technique.

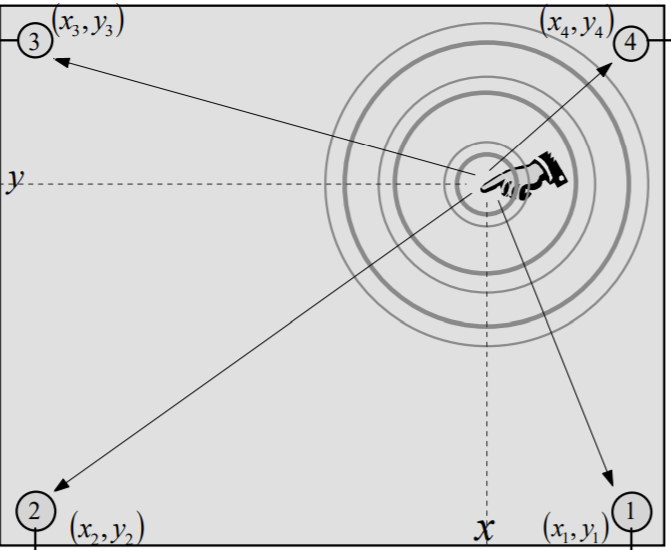


Figure : TDOA working principle. Multiple sensors surrounding the tap window, Source: [5]

TDOA type locators are all based on an arrangement of isolated receivers or an array of sensors. Delay in the signals arrival time is firstly calculated in the far off sensors. It is calculated for each pair of sensors and hyperbolic curves are drawn for them accordingly. Later they are cross matched to find the exact location.

We explain further with an example. In case of the figure above, borrowed from [5], the location of impact (x, y) can be located with some fairly simple calculations.

**(1)**

**&**

**(2)**

Here ∆t1 is the time difference between sensor 1 and 3. And ∆t2 is time difference between sensors 2 and 4.

Although it proposes a very simple and geometric solution to the problem, but it requires specialized hardware that will be used only for this application, it may be suitable for certain use cases, but we aim to provide this feature in smart phones which is rather counterproductive with this hardware.

### Template Matching

It is possible to reconstruct a wave that has been scattered through a medium by recording the received signals and simulating sending it back to its original location. This suggests that the information about its origin can be extracted from a wave without having to use multiple sensors to triangulate its location. It has been used by DT Pharma in paper “Tangible Acoustic Interface Approaches” [5]

This approach treats received signals as a function of its source location to pinpoint the location of impact. To be able to achieve this we need to collect some initial training or template data. Waves at different locations are generated manually and saved. Then the subject wave is compared to the available template data and the result is drawn using the closest match. The matching process may be different for different types of waves, different features of the wave have to be calculated and compared for accurate results. The following figure illustrates the working principle of this technique.

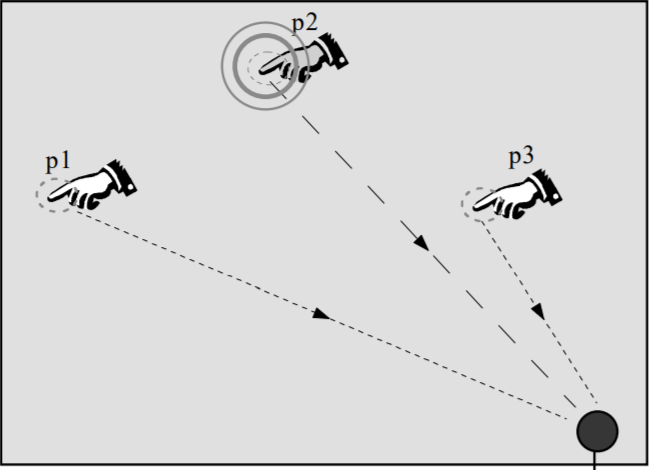


Figure : TM approach. Single sensor, Source [5]

When contact is made by the finger at the surface small waves are generated which spread in all directions. As they propagate their amplitude decreases with distance. As they hit the detector, the duration of the response depends on how much the wave is absorbed by the material, and using the amplitude, frequency, and multiple other features a computer can be used to determine a best match from the already available training data. This is the most advanced technique and has been successfully used by Alain Crevoisier in the paper “Tangible Acoustic Interfaces and their Applications for the Design of New Musical Instruments”, hence it seems to provide the most promising results for our project.

### Acoustic Holography

**Near Filed Acoustic Holography** (NAH) is a method by which sources of vibration in a 3D field can be visualized by taking acoustic pressure measurements at fixed and known points. Wave properties e.g. pressure, intensity and particle velocity can be measured as a function of position and time.

It is well used for acoustic source localization in mid-air. It visualizes audio source in a similar way an IR camera visualizes heat sources. This technique has only been used in one paper [6], which concluded Template matching to be more effective, hence its application will yet be dodged in our project.

### Vision

There are a number of projects done using cameras and computer vision to implement a virtual keyboard or any other input device. Many approaches have been followed for this purpose, some are training based, some are purely image processing based and some hybrid. For training based methods the system is trained beforehand by providing data of what each key press looks like and later the test data is compared. In Processing based approaches there are further many methods that have proven successful. One method used by Vishal D. Gate is worth discussing which uses shadow detection. The camera emits a light on the users hand and captures the shadow formed by it. From this shadow it calculates the probability of each key press and returns the key with the highest probability.

Although computer vision is a promising solution to our problem, but it is not the answer that we seek. Vision requires additional hardware and specific positioning. Although smart phones have cameras they need to be placed vertically so the camera can point on the surface. Furthermore this technique will not work in dark areas unless assisted with a light, which is again additional hardware. The purpose of this project is to make a very simple and effective input mechanism which the user expects to work without any extra hassle, hence we conclude that computer vision will not serve as a solution to our problem.

## Applications

In this section we discuss a few applications of the above techniques.

### Virtual Keyboard

Referring to a US patent [7], a sound based virtual keyboard design was registered by HP in 2006. It uses a number of sensors embedded in a device which detects user interaction by taps. It uses an array of sensors placed on a surface to detect location of sound or vibration transmitted by the medium surface. There is a signal processor to which all these sensors are connected. It is responsible to determine the point of origin. This device is then connected to a PC which only knows it is a simple keyboard. All the processing is done in the device and only the resultant signal is forwarded. Since it is an old implementation it uses the TDOA approach discussed above.

The patent request also registers a vision based virtual keyboard to be used with the portable device of the time PDAs. It uses a separate camera dock on which the PDA is to be placed. The camera records a window to be tapped and recognizes user interaction using principles of computer vision.

### Musical Instrument

Another interesting application of this idea was proposed by Alain Crevoisier in his paper “Tangible Acoustic Interfaces and their Applications for the Design of New Musical Instruments” [6]. This paper studies both TDOA and Time reversal techniques and their suitability for this application and concludes at the end which one is giving better results.

Unlike our project, this project transforms 3D surfaces into tangible interfaces by incorporating more than one sensors and to permanently transform them into these input devices.

Alain concludes his paper by stating how exciting and interesting application can be made out of TAI. Any surface can be made into an interactive device by using just a few sensors. All three TAI techniques were tried with Time reversing being the most successful one. It was able to tell difference between blocks of an A4 sheet divided in 9x6 parts. This promises successful result for our project as well.

## Literature Review Summary

The techniques used by different researchers are mentioned here along with their results. The most effective techniques was found to be template matching with TDOA being on the second position.

|  |  |  |
| --- | --- | --- |
| **Researchers** | **Method** | **Description** |
| D T Phama , M Al-Kutubia | TDOA & Time Reversal Signal Processing | Template Matching Acoustic source localization proved to be more effective |
| Alain Crevoisier,  PietroPolotti | TDOA, Time Reversal, Acoustic Holography | Template matching proved more effective |
| Theodore B. Ziemkowski, HP | TDOA, Computer Vision | Both methods give promising results once the hardware provided is sufficient |
| Erez Posner, Nick Starzicki | Computer Vision (Training) | Promising results |
| Vishal D. Gade, Parth R. Gajmal | Computer Vision (Image Processing) | Promising Results |
| MayankGoel , Jacob O. Wobbrock , | Template matching using Gyroscope | Effective up to 84% |

Table : Literature Review

# Chapter 3: Requirements and design

This chapter gives lists down and describes the functional and non-functional requirements of the project and gives a detailed description of the architecture of our user interfacing system.

We will present a quick overview of the design and then we will give a brief description of each module and its functions one by one.

## 3.1 Requirements

This section details the function and non-functional requirements of the system.

### 3.1.1 Functional Requirements

1. Developers should be able to easily insert the system into their apps using our API
2. The system will sense waves and disturbances on the surface using the accelerometer.
3. The system will use the readings from the accelerometer to extract wave features such as direction and intensity.
4. The system will process the wave feature data utilizing A.I models and analytical methods.
5. The system will use the processed wave data to rule out false positives. i.e. disturbances that are not taps.
6. The system will detect the sound created by the waves using the phone’s microphones.
7. The system will use the sound data to triangulate the source of the vibrations.
8. The system will relay the information about the direction of the source of tap to the application using the API.

### 3.1.2 Non Functional Requirements

#### Usability/Reusability

We will create a simple API that developers will be able to use to integrate our system into their apps.

#### Performance

There should not be any noticeable lag b/w tapping and feedback on the screen of the phone. The system should not take more than 50ms to process the tap.

#### Compatibility

The system should be compatible with all smartphones running android OS.

#### Reliability

The system should reliably detect the direction of the source of vibrations accurate up to 45 degrees.

## 3.2 Design

This section provides a brief overview of the functions of each module of the system, details of the implementation and algorithms, APIs and models used are given in chapter 4.

The system has been divided into 2 major phases:

* Training
* Evaluation

### 3.2.1 Training

Training has been divided into following modules:

1. Accelerometer and Gyroscope Processing
2. Tap detector
3. Wave Recording
4. Feature Extraction
5. Training
6. Model

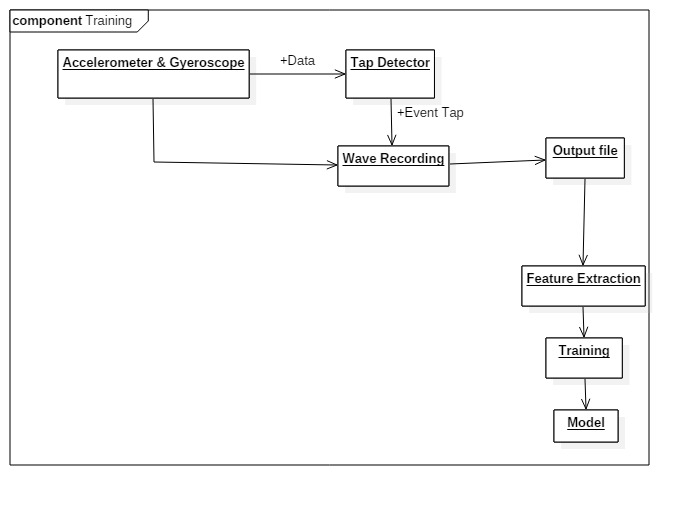


Figure : Training

#### Accelerometer and Gyroscope processing

This system will access the real time data from the accelerometer directly at a sampling rate of 100Hz by using the android API.

#### Tap Detector

The system needs to be able to detect which disturbances or taps need to be recorded as data which will be used to train our A.I model (details given in chapter 4). It looks for disturbances in the accelerometer’s readings compared to the normal values i.e. 9.8 m/s/s for the z axis and gets triggered when this value changes within a specific window of amplitude (disturbances that are too big or small will be ignored) in order to activate the wave recording module.

#### Wave Recording

Once the system has determined that a wave needs to be recorded, it takes the values from the readings of 32 points or samples on all three axes x, y and z and saves them in a CSV file.

#### Feature Extraction

After the wave data has been collected and saved, it will be used to extract wave features such as amplitude, centroid std. deviation etc. (details in chapter 4). These features will be used to train the A.I model.

#### Training

The features extracted from the waves will now be used to train models using a number of A.I models.

#### Model

Once the model has been trained it can be used in the evaluation phase to determine the direction of the sources of taps.

### 3.2.2 Evaluation

This is the part of the system that will run in real time on the user’s phone. The Evaluation phase consists of the following modules:

1. Accelerometer and Gyroscope Processing
2. Tap detector
3. Model Comparator
4. Microphone
5. Channel Separator
6. Time delay of Arrival Component
7. Result

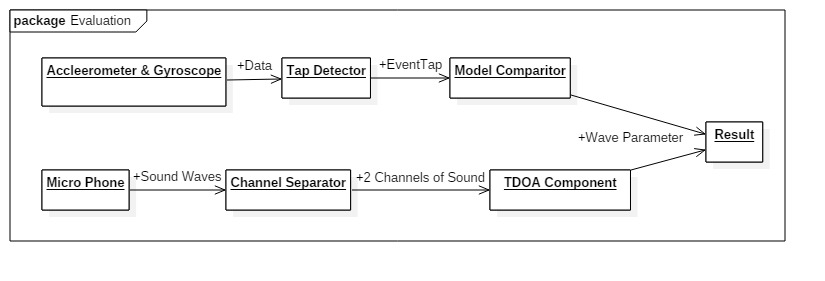


Figure : Evaluation

#### Accelerometer Processing

The system will access real time data from the phone’s accelerometer using to detect any taps or disturbances nearby.

#### Tap Detector

The system will detect taps made by the user. It will look for disturbances in the accelerometer’s readings compared to the normal values i.e. 9.8 m/s/s for the z axis and gets triggered when this value changes within a specific window of amplitude (disturbances that are too big or small will be ignored) in order to activate the comparison module.

#### Model Comparator

Once a disturbance has been identified as a tap, it’s features are extracted in real time and are evaluated against our previously trained A.I model in order to identify the direction of the source of the tap.

#### Microphone

The system will use the microphone to aide in the detection of taps, the microphones of the phone will listen for sounds within a certain window of loudness and frequency, depending on the material of the surface on which the phone is placed.

#### Channel Separator

The android API does not always support accessing both microphones separately. The sound data we receive will be multi-channel data which will have the audio from both the main mic and the secondary mic used for noise reduction so we will have to separate the audio of the microphones ourselves by isolating the channels.

#### Time Delay of Arrival

The two microphones are located on different parts of the phone’s body; the system will use this to its advantage. The system will calculate the time delay between the detection of the sound of the tap by the first and second mic. The system will then use this information to triangulate the source of the tap and reinforce the result gained from the accelerometer.

#### Result

After comparison with the model, the system will return the result which will give us the direction of the source of the tap, this information will be passed on to the application using the system through our API.

This chapter listed down all the functional and non –functional requirement of the system and gave a brief description of each module of the system and its functions. Further details of the implementation of the system and the A.I algorithms used are given in the following chapter.

# Chapter 4: Implementation & Test Cases

This chapter elaborates details of the final prototype that has been developed including the workflow, libraries, features and A.I models used in development and the prototype. In the sections we also first briefly discuss the ideal behavior that we expect from our final application and then move towards what we have achieved so far.

## Prototype Workflow

The prototype developed is a now a complete working demonstration of the system in real time. In the initial development stages, the application was ran in a passive state, by collecting the data on a smart phone and moving that to a computer for further processing, it was necessary as we had do discover all our options e.g. choosing which sensors, what features and which models give the best results without having to go through the formalities of implementing all that in real time, and when we have the results, finally incorporating them in our end application.

The training process includes several stages. First stage is Data Collection which happens on a smart phone. A sensor is registered in the application and its stream is constantly listened for taps. We followed a trigger based approach. First it should be explained what data we want to record. The goal is to record the sensor values whenever a tap is made around the phones surface. But for that we need to know when a tap occurs. After observing tap waves we created a rudimentary algorithm that successfully does the job of detecting a tap (note that it only detects a tap and does not identify its parameters). After a tap is detected we first record a fixed number of data points after this event and save it in a file for accelerometer and in the case of microphone we calculate certain features of the audio data and save that in a CSV file. Label is added for the supervised training, in this case it is the direction of the tap (recorded manually).

The trigger based tap detector works in a way that it maintains an average from last few time instances of the sensor values. When the phone is placed idle, the sensor values will not change drastically. A threshold for peak amplitude, wavelength, range and frequency is defined (different for all sensors) and when the wave reaches this specific window, we consider it to be a Tap. This works regardless of the phones orientation and placement as long as it is not being constantly disturbed.

Data Collection stage results in a CSV file which contains instances of taps as recorded by the sensor. For accelerometer each instance has 3 waves from all the axes and a label indicating the direction. And in case of Microphones, the features calculated on the phone for both Left and Right channels are written to the file. Detail of features is given in Section 4.3.

The last stage is the supervised training. We are using WEKA for this purpose because of its wide application in data statistics and our personal preference. The WEKA package consists of a comprehensive set of A.I models. It contains ANNs, Bayesian models, Random Forests, SVMs etc. We can configure the size and ratio of training and test data, select different models, display confusion matrices of results, view deciding features and plot models and data.

To deploy the model on a smartphone, we generate a java file containing the implementation code of the random forest classifier. The code is generated by running a python script which takes a model printed by WEKA in a .txt file as input. This java file is then added to our app in android studio. Since the code is purely java with no external dependencies involved, it is very fast and the latency of classification is very low.

Once the model has been added to the app, the app simply gives the data of each successive tap (received from the microphones or accelerometer) made on the surface to the model, the model quickly classifies it and the app displays the result on the GUI.

## Sensors Used

We are aiming to create an android system or API that will continuously listen for vibrations using any combination of the three sensors; accelerometer, gyroscope and microphone. The less number of sensors used the better. Sensors drain a smartphones battery very quickly. Power used depends mainly on the sampling rate that the sensor is set to give output on. To detect and classify such minute vibrations (generated by tapping) we need the sampling rate to be high to get a more high resolution wave. But to keep the systems impact on the battery minimal we need less number of sensors being used in the end system along with minimum sampling rate as well.

Currently the most important sensor in the system are accelerometer and microphone. As discussed ahead in Chapter 5, we have achieved relatively high accuracy using these sensor. Gyroscope does not provide valuable enough information of direction or distance of a tap. We were able to distinguish taps in four cardinal directions around the phone but at a high sampling rate of 100Hz in case of accelerometer and 2700Hz in case of Microphones.

## Detail of Features

The ideal system should be able to extract valuable and discriminating features from the waves in runtime and compare them with an already generated A.I. model to compute results about direction and distance of the taps. In our previous iteration of the prototype we were using a library jAudio[1] to extract all well-known features by brute force to help us identify the deciding features. At this point we had an idea of which type of features will help us, and after research on them we have implemented our own light weight and targeted feature extractor that works on Android in run time.

Although Spectrums are discussed in detail in the “Feature Extraction Process” section but for now audio features in the “Spectral Domain” deal with the shape and construction of the wave.

* **Amplitude Spectrum**

Frequency domain amplitude spectrum.

* **Rate of Change of Amplitude**

2nd order derivative of amplitude related features.

* **Power Spectrum**

Frequency domain power spectrum.

* **Spectral Centroid**

Indicates Mean / Centre of spectral center.

* **Spectral Flatness**

It is an indicator of the noisiness of a sound. How much the wave Fluctuates.

* **Spectral Slope**

Measure of how inclined is the shape of spectrum.

* **Spectral Roll off**

The frequency below which 99% of the energy spectrum lies

* **Spectral Skewness**

Indicator of what values the spectrum is inclined towards if any.

* **Spectral Kurtosis**

The pointedness of spectrum. Indicated pitch.

### Feature Extraction Process

We are presented with a one dimensional array of samples taken at different time intervals. We do not need to know the time of each sample, for that we have the sampling rate. To understand how the above features are extracted from these waves following concepts are discussed.

* Spectrum
* Cepstrum
* MFCCs

Spectrums are a representation of waves. [2] [3] Fourier theory suggests that a wave can be represented by a summation of its sinusoids. Each having different amplitude and phase. This representation is called a spectrum. Spectrum in simple words is a frequency-magnitude graph. Following wave is represented using a spectrum.

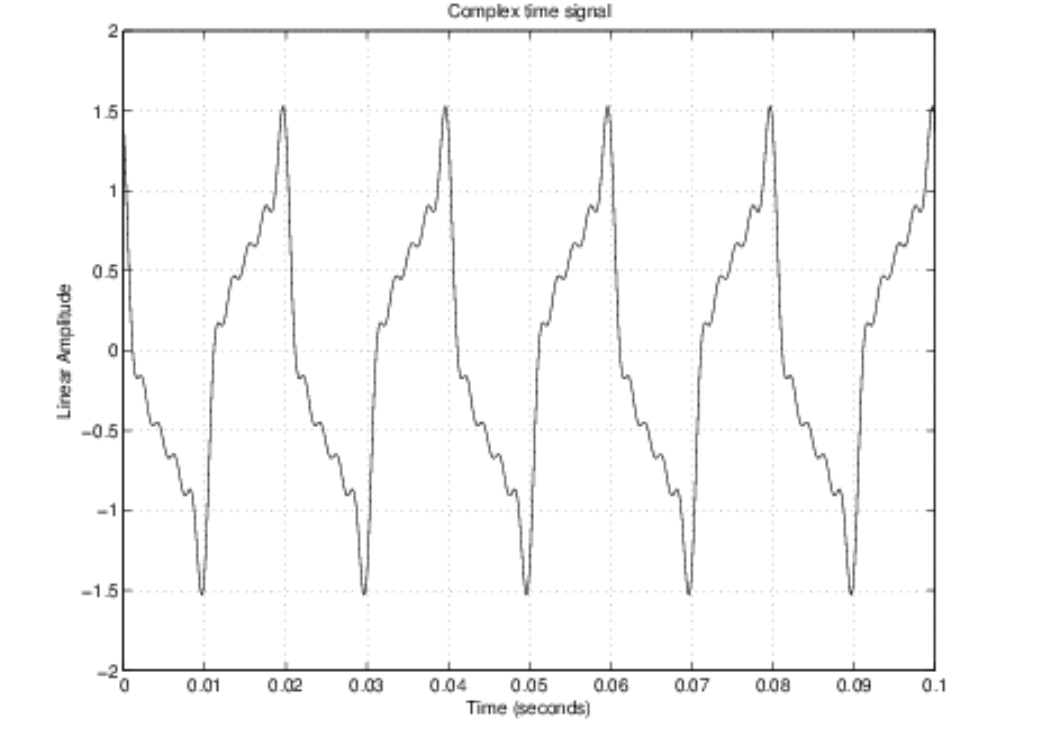


Figure : A complex time-domain signal Source [3]

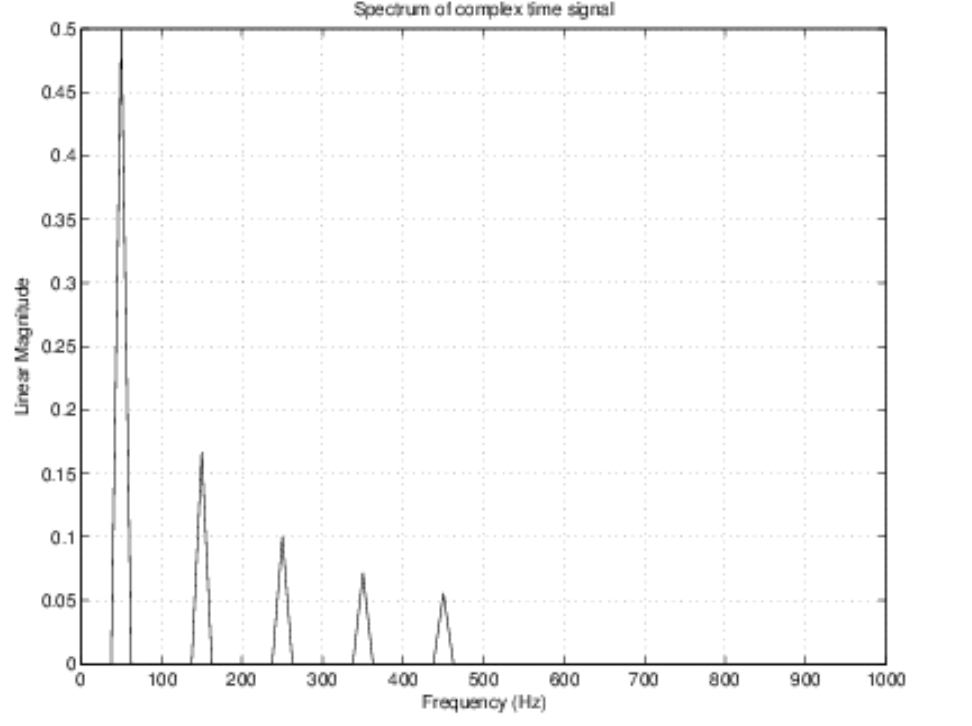


Figure 6: Magnitude spectrum of time-domain signal plotted above Source [3]

Once the spectrum is obtained, we can calculate the “Cepstrum” from it. Cepstrum [2] is calculated by taking the inverse Fourier transform of the logarithm of estimated spectrum of a signal. There can be many types of cepstrum, complex cepstrum, real cepstrum, power cepstrum and a phase cepstrum. Power cepstrum is widely used in speech recognition systems.

MFCC’s are derived from cepstrum of an audio signal. In MFC the frequency bands are equally spaced on the mel-scale as compared to the linearly spaced frequency bands in normal cepstrums. This allows for certain frequency bands to be at a higher weightage than others. Since our waves will be at a very low frequency even for the highest sampling rates on the sensors, we will focus more in the low frequency area.

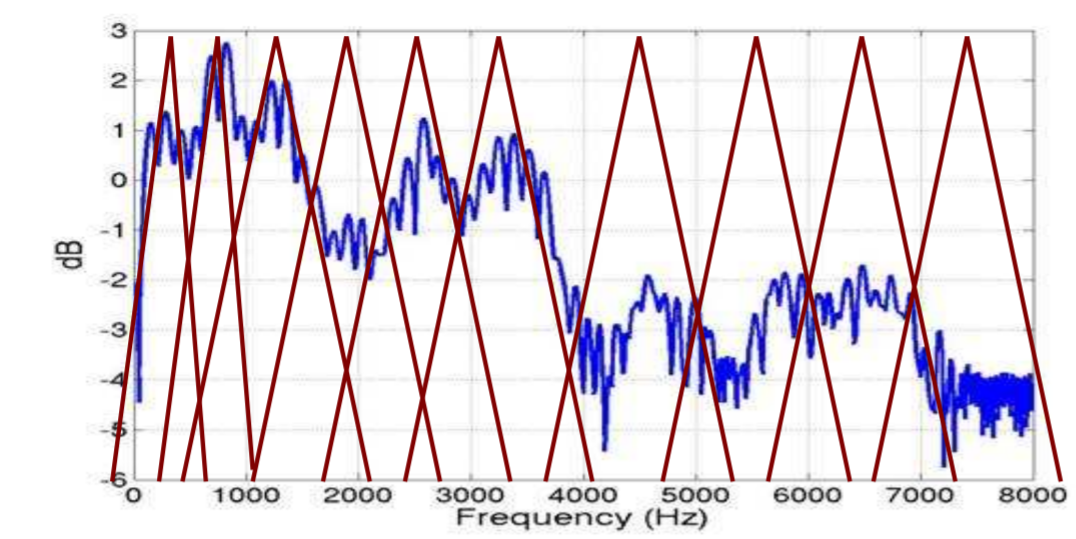


Figure 7: More filters in low frequency bands: A way to eliminate unnecessary information from signals Source [4]

Signals in the higher frequency domain will be considered the same when the spectrum is constructed. For example all samples in the range 7000 – 8000 Hz will be summed in a single 8000 Hz point on the spectrum, it makes it easier to ignore unwanted parts of the wave. Then spectrums are created on different intervals in the waves. We can use their average, standard deviation or any other statistical inference from them. This data is then used to train the A.I. model which compares the features to identify the direction of a wave.

## A.I. Models

We are using supervised learning in this project. Selecting the best model is necessary for the system. This will decide how much we can reduce the feature set, training instances, training time etc. As discussed in our functional specification documentation we will try different types of algorithms and decide between them based on the results and resources they use. Details of the most successful models in our project are discussed.

Data set at the time of initial testing consists of 2000 total observations. Data is divided into 4 directions each having 500 training instances i.e. 500 waves for “UP” direction, 500 for “RIGHT” etc. Each instance have 3 waves (X, Y and Z) and a direction label for accelerometer and variable features and a training label for microphones data.

### Decision Trees

Decision Trees work by recursively dividing the training data into sets of similar instances based on the most discriminatory feature. Different division criteria’s are tried to find the best one using greedy approach. Training will stop when division no longer results in different enough sets.

Decision Trees work for both categorical and numerically divided data, but one weakness is that once the model is generated it cannot be incrementally updated, there needs to a new model trained for new data.

From the different models tested up to this point, decision trees have been the most efficient. For accelerometer we have achieved an accuracy of 76% and for microphone 87%.

### Random Forest

Since Decision trees have given relatively high accuracy, random forests had to be tested. Random forests is a classification and regression technique that constructs multiple decision trees at the time of training. And outputs the class that is the mean of prediction of the individual trees. This corrects the “Over fitting” problem of trees on training data.

Random Forests have given us an accuracy of 79% for accelerometer and 92% for microphone data.

### Neural Network

It is a collection of perceptron layers. It can learn nonlinear relationships between the input and output easily. The technique used is back propagation of error and weight adjustment. These methods work on linear and categorical data both. A multilayered perceptron network achieved 65% accuracy on raw accelerometer data without features and 77% on raw microphone data, but to achieve a higher accuracy we need a much larger data set.

With traditional A.I methods, a large amount of time is used in feature extraction. Features are the deciding factor on which the model is built, however with deep learning neural networks, they identify features and make inferences from them themselves, and build the model around it. They only need data and its class with it. However this requires that the data is very large even millions of marked instances. Since we have extracted important features, it makes more sense to use a traditional A.I method rather than deep learning. The problem we face here is the amount of data should be tens of thousands of samples to avoid over fitting.

## Test Cases

The testing phase has been around doing Black Box Testing. For this we mainly test the accuracy of the system and its compatibility with different hardware and software configurations.

**4.5.1 Detect Taps on The Right**

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case ID:** | 1 | **QA Test Engineer** | Rahim Shahid |
| **Test case Version:** | 1 | **Reviewed By** | Waheed Amir |
| **Test Date:** | 10/3/2018 | **Use Case Reference(s)** | N/A |
| **Revision History** |  |  |  |
| **Objective** | Taps made by user on the right must be classified as right. |  |  |
| **Product/Ver/Module** | Accelerometer/Mic |  |  |
| **Environment:** | Android |  |  |
| **Assumptions:** | Taps are made on suitable surface. |  |  |
| **Pre-Requisite:** | Phone has two microphones. |  |  |
| **Step No.** | ***Execution description*** | **Procedure result** |  |
| 1 | User taps on the right side of the phone. | System will use the classifiers to determine direction of the tap. |  |
| 2 | The system displays that tap was made on the right. |  |  |
| **Comments** Our system is working according to our need. |  |  |  |
| *Status:* | Passed |  |  |

**4.5.2 Detect Taps on The Left**

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case ID:** | 2 | **QA Test Engineer** | Rahim Shahid |
| **Test case Version:** | 1 | **Reviewed By** | Waheed Amir |
| **Test Date:** | 10/3/2018 | **Use Case Reference(s)** | N/A |
| **Revision History** |  |  |  |
| **Objective** | Taps made by user on the Left must be classified as Left. |  |  |
| **Product/Ver/Module** | Accelerometer/Mic |  |  |
| **Environment:** | Android |  |  |
| **Assumptions:** | Taps are made on suitable surface. |  |  |
| **Pre-Requisite:** | Phone has two microphones. |  |  |
| **Step No.** | ***Execution description*** | **Procedure result** |  |
| 1 | User taps on the Left side of the phone. | System will use the classifiers to determine direction of the tap. |  |
| 2 | The system displays that tap was made on the Left. |  |  |
| **Comments** Our system’s accuracy is lower than expected while classifying the left side taps |  |  |  |
| *Status:* | Passed |  |  |

**4.5.3 Detect Taps made Above**

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case ID:** | 3 | **QA Test Engineer** | Omer Haroon |
| **Test case Version:** | 1 | **Reviewed By** | Waheed Amir |
| **Test Date:** | 10/3/2018 | **Use Case Reference(s)** | N/A |
| **Revision History** |  |  |  |
| **Objective** | Taps made by user above must be classified as above. |  |  |
| **Product/Ver/Module** | Accelerometer/Mic |  |  |
| **Environment:** | Android |  |  |
| **Assumptions:** | Taps are made on suitable surface. |  |  |
| **Pre-Requisite:** | Phone has two microphones. |  |  |
| **Step No.** | ***Execution description*** | **Procedure result** |  |
| 1 | User taps above the phone | System will use the classifiers to determine direction of the tap. |  |
| 2 | The system displays that tap was made above. |  |  |
| **Comments** Our system is working according to our need. |  |  |  |
| *Status:* | Passed |  |  |

**4.5.4 Detect Taps made Below**

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case ID:** | 4 | **QA Test Engineer** | Omer Haroon |
| **Test case Version:** | 1 | **Reviewed By** | Waheed Amir |
| **Test Date:** | 10/3/2018 | **Use Case Reference(s)** | N/A |
| **Revision History** |  |  |  |
| **Objective** | Taps made by user below must be classified as below. |  |  |
| **Product/Ver/Module** | Accelerometer/Mic |  |  |
| **Environment:** | Android |  |  |
| **Assumptions:** | Taps are made on suitable surface. |  |  |
| **Pre-Requisite:** | Phone has two microphones. |  |  |
| **Step No.** | ***Execution description*** | **Procedure result** |  |
| 1 | User taps below the phone | System will use the classifiers to determine direction of the tap. |  |
| 2 | The system displays that tap was made below. |  |  |
| **Comments** Our system is working according to our need. |  |  |  |
| *Status:* | Passed |  |  |

**4.5.5 Test Application on Accelerometer Sampling Rate Below Minimum**

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case ID:** | 5 | **QA Test Engineer** | Omer Haroon |
| **Test case Version:** | 1 | **Reviewed By** | Rahim Shahid |
| **Test Date:** | 10/3/2018 | **Use Case Reference(s)** | N/A |
| **Revision History** |  |  |  |
| **Objective** | Accelerometer Module should not participate in classification, but detection should be done |  |  |
| **Product/Ver/Module** | Accelerometer |  |  |
| **Environment:** | Android phone with Accelerometer sampling rate below 100Hz. |  |  |
| **Assumptions:** | Taps are made on suitable surface. |  |  |
| **Pre-Requisite:** | Phone has two microphones. |  |  |
| **Step No.** | ***Execution description*** | **Procedure result** |  |
| 1 | User starts the app using UITMS API. | System will perform sanity checks to ensure the phone has the required hardware. |  |
| 2 | App will start but results will lack the maximum achievable accuracy |  |  |
| **Comments** The system reduces functionality instead of crashing. |  |  |  |
| *Status:* | Passed |  |  |

**4.5.6 Test Application on Un-supported Microphone Configuration**

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case ID:** | 6 | **QA Test Engineer** | Saad Mazhar |
| **Test case Version:** | 1 | **Reviewed By** | Rahim Shahid |
| **Test Date:** | 10/3/2018 | **Use Case Reference(s)** | N/A |
| **Revision History** |  |  |  |
| **Objective** | Lack of availability of two physical microphones should result in an error message. |  |  |
| **Product/Ver/Module** | Microphone |  |  |
| **Environment:** | Android phone with Only one physical Microphone. |  |  |
| **Assumptions:** | Taps are made on suitable surface. |  |  |
| **Pre-Requisite:** | Phone has two microphones. |  |  |
| **Step No.** | ***Execution description*** | **Procedure result** |  |
| 1 | User starts the app using UITMS API. | System will perform sanity checks to ensure the phone has the required hardware. |  |
| 2 | App will not start, prompting an error message that required minimum hardware is not available. |  |  |
| **Comments** The system should halt execution, as triangulation from accelerometer only is not accurate enough for a product. |  |  |  |
| *Status:* | Passed |  |  |

## Bug Report

Following portion of the document mentions about the bugs that were found during the testing phase of the project:

### Single Channel Microphone Bug

Submitter  
Name: Omer Haroon  
Email: [l144119@lhr.nu.edu.pk](mailto:l144119@lhr.nu.edu.pk)

Date Seen15th March 2018

Bug DescriptionIf we use our microphone module on a phone with only 1 mic it outputs the same channel twice. The system perceives it as getting two distinct channels when in reality the channels are identical.

SeverityMajor

Steps to Reproduce1. Use the system normally on a phone with 1 microphone

Actual BehaviorThe system thinks it’s getting two distinct channels when in reality the channels are identical. This reduces the classification accuracy of the microphone module greatly.

Expected BehaviorIf the system is used on a phone with 1 mic then the microphone module should be ignored.

WorkaroundApply a check when the system is first started to determine the number of distinct physical audio channels being output by the phone. If it is equal to 2 then proceed as usual otherwise ignore the microphone module while classifying taps.

4.6.2 Un-Supported Sampling Rate Bug

SubmitterName: Rahim Shahid  
Email: [l144137@lhr.nu.edu.pk](mailto:l144137@lhr.nu.edu.pk)

Date Seen19th March 2018

Bug DescriptionIf the maximum sampling rate of the accelerometer on the phone being used is less than 100 Hz then it reduces the accuracy of the accelerometer model greatly.

SeverityMajor

Steps to Reproduce1. Use system normally on a phone with max sampling rate < 100 Hz

Actual BehaviorAccuracy greatly reduced since we collect data based on a 100 Hz sampling rate.

Expected BehaviorSystem should detect sampling rate less than 100 Hz and try to raise it to 100 Hz if possible otherwise display an error message.

WorkaroundMeasure the sampling rate of the accelerometer manually, if it’s less than 100 Hz then try to set to 100 Hz, if it isn’t supported then ignore the accelerometer module and give an error message.

4.6.3 Tap Overlap Bug

SubmitterName: Saad Mazhar   
Email: [l144151@lhr.nu.edu.pk](mailto:l144151@lhr.nu.edu.pk)

Date Seen12th April 2018

Bug DescriptionTwo taps made in quick succession are considered as 1 tap by the system.

SeverityCatastrophic

Steps to Reproduce1. Make two taps one after the other very quickly

**2**. The data from both taps will be recorded in the queue which is kept large to accommodate data from larger taps.

Actual BehaviorThe system considers the data from both taps while trying to classify a single one leading to inaccuracy in the classification.

Expected BehaviorThe system should consider only 1 tap at a time.

WorkaroundThe system should stop listening for more taps 100 milliseconds after each tap so as to avoid overlapping of tap data in the queue. We cannot reduce the queue size because some taps have more data points require more space to be stored.

## Conclusion

The prototype is a demonstration of the complete working project that detects and identify direction of taps in four cardinal directions in real time with reasonable accuracy. We collect sensor wave data from the phone, extract important features and train an A.I model on this data. After that we test our model on new data to check for accuracy. The sensors we are using are accelerometer and microphone. Ideally we should use the least amount of sensors and minimum sampling rate but we need to make a trade-off between the two. Feature set should also be minimum. Features in the Spectral domain play the most important role in predicting the direction of waves. Spectrums are calculated on multiple time intervals in a wave for relative comparisons. The ideal training model should be trained fast, use least number of features in making a decision and should train on least amount of data. The most efficient model on our training data were Random Forests, a combination of multiple decision trees. In the testing phase, most of the test cases test the accuracy of the system and its compatibility across odd hardware and software configurations.

# Chapter 5: Experimental results and analysis

The purpose of our experiments was to discover the best configuration of sensors on the best working surfaces for the project. We test all sensors separately and then combined to find the best combinations. Our decision will be based on accuracy, portability and availability. Following is the detail of our test bench.

**Surface:** Wood, Plastic, Metal, Glass.

**Phones:** Samsung Galaxy S8+, Samsung Galaxy Note 8.

**Pre-Conditions:** Tap made with reasonable intensity, and distance from the device.

## 5.1 Accelerometer Results

* **Detection**

Using the Accelerometer to detect the occurrence of a tap has been done with close to a 100% accuracy. Using this sensor, we are able to accurately filter out other disturbances from a surface. This works on all of the surfaces Wood, Plastic Metal and Glass.

* **Triangulation**

Using the accelerometer to identify the direction of a tap yields a maximum accuracy of about 75% accuracy. This works on Wood, Plastic and Glass surfaces.

* **Portability**

A model trained with accelerometer data loses most of its accuracy if the surface is switched i.e. if a model is trained on wood, its accuracy will fall to 20 – 30% if we test it on glass.

* **Availability**

Accelerometer is available in almost all modern smartphones.

## Microphone Results

* **Detection**

Detecting taps using the microphone showed weak results on all surfaces, mostly because of surrounding noise. Filtering out the inaudible noise of a tap from other background noise is a difficult task.

* **Triangulation**

If once we know that a tap has occurred, we can access the mic data from that time instance and accurately determine the direction of a tap. Our experiments give 90%- 95% accuracy in controlled conditions.

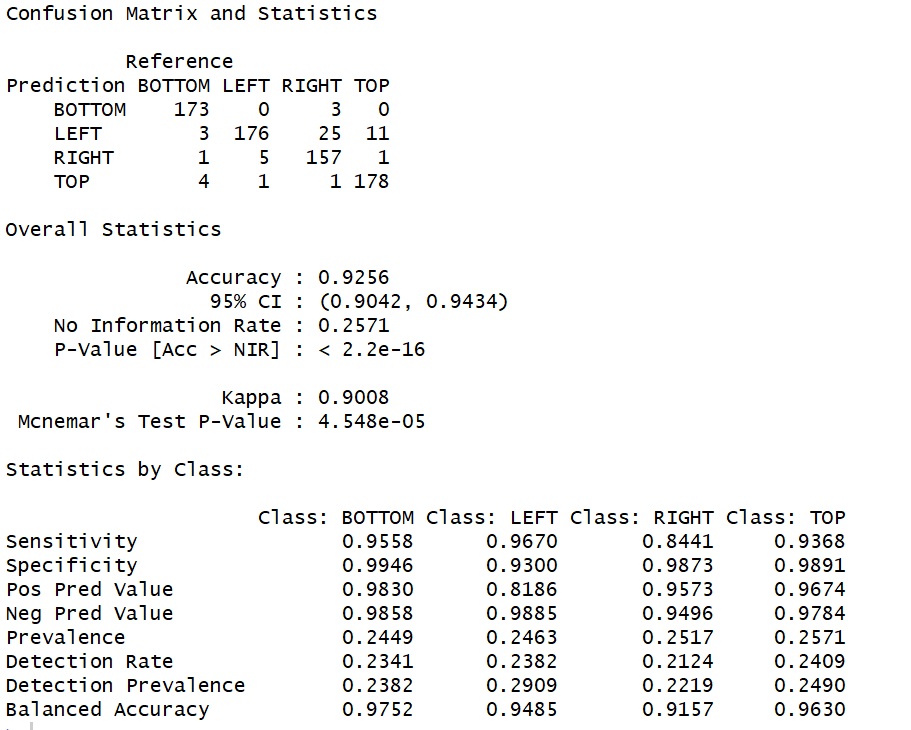
* **Portability**

Availability of 2 mic sensors allow comparative features to be extracted from the data. This will allows us to make the model more portable compared to the accelerometer. A model trained with mic data on one surface will reduce its accuracy from say 90% to mostly in the range of 65% - 75%.

* **Availability**

All Android smartphones are not guaranteed to have 2 physical microphone sensors available. However more recent or higher end phones do come equipped with them.

Confusion Matrix and some other useful statistics of our Microphone model are given below.



## Conclusion

From the above results we conclude that the best configuration would be to use accelerometer for the detection part, and as soon as a tap is detected record the microphone data from that time instance. Triangulate tap source location using the microphone. However, we will not completely ignore the accelerometer in the triangulation phase as it will help us reinforce the mic results and when a phone is not equipped with 2 mic sensors, we will base our results completely on the accelerometer.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Sensor** | **Detection** | **Triangulation** | **Portability** | **Availability** |
| Accelerometer | Very High Accuracy | Moderate Accuracy | Poor Portability | Easily Available |
| Microphone | Poor Accuracy | High Accuracy | Moderate Portability | Moderate Availability |

Table 2: Sensors Analysis

# Chapter 6: Conclusions

The purpose of this project has been to research on developing a relatively unused form of input mechanism that has been made possible with the wide availability of smart phones integrated with different types of sensors. The goal has been to detect and classify “taps” that a user makes around a phone placed on a flat surface. Being able to classify taps based on their source direction will enable many kind of use cases built around this input method.

The working principle is that when physical contact is made with a solid surface, with the intention to produce a tap in our case, the vibration pattern of the substance changes according to how the energy from the strike distributes on the surface. These vibrations propagate in all directions, our assumption is that the vibrations will be distinct based on where they originate from. We want to be able to utilize only the materials structure to convey this information to the phone. For this a sensitive accelerometer or a microphone required, which are available in a modern smartphone.

Previous research has been made in this area but on sound data. Locating source of a sound is called Acoustic Source Localization. The different techniques used for this are TDOA (time-delay of arrival), Template Matching and NAH (near field acoustic holography). The first two are the most successful ones. TDOA is basically using multiple sensors to triangulate the location an acoustic source. This is done by calculating delay of detection on each sensor. Template matching approach is a comparison based technique, where a sound wave is compared to already available data produced on different locations.

Although most of the process is generic, we have targeted Android smart phones. The device must have an accelerometer and microphone, which will be used to listen for waves generated in the surface the phone is placed on. Minimum error in angle is set to 45 degrees i.e. the system should be able to identify taps made in at least four directions around the phone. After achieving these results, an API should be developed to be used in apps and a demonstration app be developed.

The API developed up till now completely demonstrates the project. As soon as the accelerometer detects a tap, microphone and accelerometer data is collected and features extracted. A model is trained on these features and deployed in the API. Now for using and testing, accelerometer detects a tap and again those features are extracted and compared with the models, the final result is an aggregate of the different models deployed in the API.

We have observed that for triangulation microphones give a much better accuracy than accelerometer, hence we give a higher weightage to the result by the microphone model. Tap detection is done unquestionably by the accelerometer as microphones are much more prone to noise. The best classification model has been Random Forests so far with an accuracy of around 90%.

## 6.1 Challenges & Limitations

We have faced a number of problems during the research and implementation of the project. Most of the problems have been resolved, but a few still remain, some of them are hardware limitations, and some have a software solution but require extensive research themselves. Below are discussed problems related to accelerometer and microphone sensors, and an issue of portability for the entire project.

## 6.2 Accelerometer

Availability of only a single sensor is the major hurdle in the process of triangulation using accelerometer. We have to rely totally on AI based techniques, which too could not achieve an accuracy higher than about 75%.

## 6.3 Microphone

Microphones solve the problem of triangulation, but they are much more prone to noise. The availability of two microphones is actually for noise cancellation, but since we are using both the channels input to our process, that is not an option. Software noise cancellation is much harder to implement and still less accurate.

## 6.4 Physical Location of Sensors

Changing the relative location of the two microphones and the accelerometer results in some loss of accuracy, however this can be solved by tweaking a few parameters and thresholds in the system.

## 6.5 Portability

The biggest limitation of this project is surface portability. The goal was to be create a plug and play system that does not require any extra hardware. However as we started work on the project, we realized that the models we train will not be portable among different surfaces as waves/disturbances generated in a surface depend on its medium. The only solution we propose for this problem is to pre-train models on different types and thicknesses of surfaces and switch models on runtime.

## 6.7 Future Work

The core of the project is complete, however it is still a demonstration as it requires controlled conditions and pre requirements. To make it into a product, vast amount of data will have to be collected for different surfaces and models will be trained for each of them. This will ensure the system remains portable among different surfaces. By adding just one sensor, true triangulation can be done. If there are 3 microphones and one accelerometer, much more directions can be calculated. Even if an extra accelerometer is available and two microphones only, comparative study can be applied on both their data which will reveal much more information on the source of the tap. However this requires a manufacturer to add an extra sensor to support this input mechanism. The implementation has been done in a layered architecture and can be deployed on any underlying hardware, we recommend a iOS version to be created as well.

# References

**Software**

[1] McEnnis, Daniel, Ichiro Fujinaga, Cory McKay, Philippe DePalle. 2005. "JAudio: A feature extraction library". ISMIR. GitHub repository <https://github.com/dmcennis/jAudioGIT>.

**Book**

[2] Steven W. Smith, Ph.D, The Scientist and Engineer's Guide to Digital Signal Processing, January 1999.

**World Wide Web**

[3] Gary P. Scavone, McGill University, <https://www.music.mcgill.ca/~gary/307/week1/spectra.html>

“Audio Spectra”.

[4] Kishore Prahallad, Carnegie Mellon University, Speech Technology: A Practical Introduction, <http://www.speech.cs.cmu.edu/15-492/slides/03_mfcc.pdf>

**Technical Report**

[5] D T Phama, M Al-Kutubia, Z Jia, M Yanga, Z Wanga, and S Cathelineb, ― Tangible Acoustic Interface Approaches, Manufacturing Engineering Centre, Cardiff University, Cardiff CF24 3AA, UK pp. 25–30, 2007.

[6] Alain Crevoisier (University of Applied Science Western Switzerland (HES-SO / EIVD) Rue Galilée 15, CH-1400 Yverdon Switzerland), PietroPolotti (ISPG) ― Tangible Acoustic Interfaces and their Applications for the Design of New Musical Instruments

**Patent**

[7] theodore b. Ziemkowski, loveland, co (us) ― sound-based virtual keyboard, device and method, hewlett packard company p 0 box 272400

**Thesis**

[8] Mayank Goel1 , Jacob O. Wobbrock2 , Shwetak N. Patel1 ― gripsense: Using Built-In Sensors to Detect Hand Posture and Pressure on Commodity Mobile Phones

[9] "Nearfield Acoustic Holography (NAH) I - Theory of Generalised Holography and the Development of NAH", by JD Maynard, E.G. Williams and Y. Lee.

[10] Paradiso ja, leo ck, checka n and hsiao k. Passive acoustic knock tacking for interactive windows. Acm chi 2002 conference, minneapolis, minnesota, 20-25 april 2002.