ECE 573 Spring 2016
Project Design Document

Teamname:BiniNewBee
Team Shortname:bnb321
Date:4/10/2016

BY

Ce Wang and Shengxiang Zhu

cewang@email.arizona.edu,
szhu@email.arizona.edu.

The University of Arizona

ECE 473/573 Project Design Document

Shengxiang Zhu
Electrical and Computer Engineering Department
The University of Arizona
Tucson, USA
Email: szhu@email.arizona.edu

Ce Wang
Electrical and Computer Engineering Department
The University of Arizona
Tucson, USA
Email: cewang@email.arizona.edu

CONTENTS

***Abstract*—This document shows the basic design information of the project of team bnb321. In this paper, we will demonstrate the goal, ideas, methods, requirements, etc. of our project.**

***Keywords*—mechanical keyboard; audio classification;**

I. EXECUTIVE SUMMARY

A. Goals

Our project is to build an application that can classify different kinds of switches of mechanical keyboard by analyzing the sound when the key is pressed.

B. Methods

The input signal will be the sound from the mechanical keyboard when a key is pressed and the signal will be received by the microphone on the phone. After the input signal is accessed, the application will analyze the sound, including but not limited to, noise reduction, Fourier Transformation, frequency analysis, etc. The output information will be displayed in the user interface which shows which kind of key is pressed. In the B requirement, the application is able to classify two kinds of switches. In the A requirement, the application may be able to classify more kinds of switches.

C. Creativity

There are many applications that are able to demonstrate FT diagram but few can classify a certain sound. In particular, there is no application, according to our research, which is able to classify different switches of mechanical keyboard by sound. There might be some libraries for voice recognition but no library is found which supports the classification of mechanical keyboard.

II. PROJECT OVERVIEW

The team members in this project are fans of mechanical keyboard. We raised our idea to make such an app because we want to learn the potential of the techniques of audio recognition and apply it into some interesting fields. In this project, we will build an application that is able to detect the sound of pressing a key and recognize which kind of switch it is. Currently, there are 8 kinds of switches in the market, which are manufactured by a company called Cherry. Almost all of the switches used in mechanical keyboard are made by this company.

Our project is like an interesting experiment. It seems like there is no use of it. However, it is a start of many more application since what we do is not merely classify switches, actually we are digging the potential of audio recognition. If we successfully made this application, it means that we can do more on audio recognition. Imagine that we might even be able to recognize different keys within a keyboard, even with the same kind of switch! We might also detect more kinds of sounds, from the abnormal sound coming from a vehicle engine to the diagnose of diseases (e.g. by the sound coming from a patient' lung).

In this project we assume that the audio process can be implement by frequency analysis, a.k.a. Fourier analysis based on our experiment. In this experiment, we tested different switches from a test keyboard(a test keyboard which has 8 different switches) and researched into the FT diagram. It shows that different switches have their characteristic frequency which is like an ID of them. However, some switches share almost the same frequency (e.g. the red switch and the black switch, whose only difference is the activation force), as is a tricky part for us.

III. REQUIREMENTS

The requirements are as follows (Table 1):

Label	Requirements	
	Contents	***Level***
1.1	This application is able to classify different mechanical keyboards by analyzing the sound come from the keyboard, assuming that it works in a quiet environment.	B
1.2	This application would display the result of classification given a certain sound input.	B
1.3	This application would be able to classify at least 2 kinds of typical mechanical keyboard switch.	B
2.1	This application could work in a normal environment where there might be some noise.	A
2.2	This application would be able to display the FFT graph of the input signal.	A
2.3	This application would be able to classify at least 3 kinds of switches.	A

TABLE 1. REQUIREMENTS.

IV. UI DESIGN

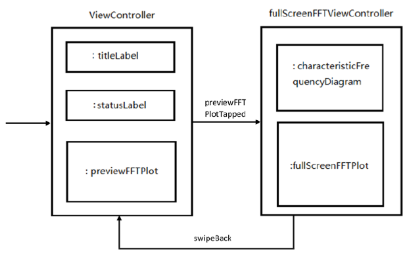


Fig. 1. UI DESIGN DIAGRAM.

- When user launches the app, View Controller class is loaded, which contains title Label (basic information of this app, class name and team name, etc.), status Label (prompts of input, result, error message, etc.), preview FFT Plot (a brief preview of real-time FFT diagram.
- In the View Controller, the app is detecting sound from microphone and analyzing and get the results in the background, while the status Label shows the status of the calculation.
- Once the preview FFT Plot is tapped, the app goes into full Screen FFT View Controller, where there is characteristic Frequency Diagram (shows the characteristic frequency of different switches and how the app classifies them) and full Screen FFT Plot (shows the real-time FFT diagram with characteristic frequency denoted on that, this diagram is different from the preview FFT Plot in that it shows detailed information of how the classification diagram works and how different characteristic frequency matches that of certain switches, which is shown in the above diagram). In this view, user can learn how this app works straight forwardly.

V. DOMAIN ANALYSIS

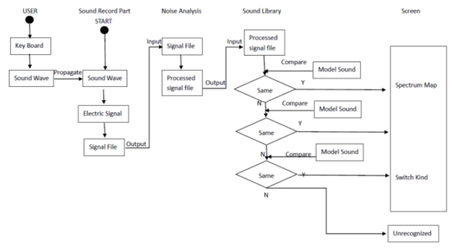


Fig. 2. INTERACTION DIAGRAM.

First, record part accepts sound wave and record, which means transfer wave signal to electrical signal in spectrum map. Then, Sound Record saves the file and transmit it to next level. Noise Analysis according to the algorithm, removes the noise signal in the file and get processed file without noise effect. After Sound Library received the processed file, compares the spectrum map of the file with Model sound in library which recoded in a really quit environment. If the file is almost same with Model Sound, Screen will show the Spectrum map of Key taped and show the kind of the key. If the file has many differences with the model sound in an area of frequency, especially in high frequency, compares it with next Model Sound until the keys frequency has a good comparison with model sound. However, if all of the model sound in library could not compare with the taped sound, screen will display unrecognized and the user may tap once more or our application do not support this kind of switch.

VI. IMPORTANT ANALYSIS

The domain algorithm is analyzing specific frequency. And remove the noise frequency in spectrum map. After recorded the sound signal, using FFT algorithm analyze the spectrum map. According to our experiment result, the frequency of key board is higher than normal noise which includes human voice, engine sound etc. And what we need to do is analyzing high frequency part of spectrum map. Cause the frequency of different key is different. Like for blue switch and black switch, the frequency of blue switch is much higher than black. So, by analyzing the average of certain frequency area and compare it with model spectrum map, we can get the result we need.

VII. CLASS DESIGN

Sound Record is the first part to manage the sound signal. It records and saves the sound as electric signal file and outputs it to next part. Noise Analysis part is the second part to correct the noise effect in origin sound file and output processed file. Sound library receives the processed file and compares it with model sound files until find the same or almost same one. If it do compares, the application shows the recognized result and spectrum map in Screen. If there is no model sound compared, the application also shows result that this switch could not be recognized and may need the user taps more times to get the final result.

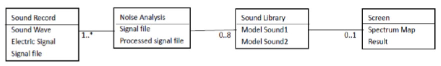


Fig. 3. INTERACTION DIAGRAM.

VIII. TESTING STRATEGY

Since our project is mainly about the classification of sound of switches, the test of UI will be rather simple. However, the process of detection, analysis and classification would be very complex.
Algorithm test:
The algorithm can be broken down into 3 parts: detection, analysis(Fourier transformation) and classification.

A. UI Test

Since this application has a very simple UI, we can make an exhaust test, which includes all the use cases. If the UI is able to correctly recognize user's action and display the result, the test would be passed.

B. Algorithm Test

The algorithm can be broken down into 3 parts: detection, analysis (Fourier transformation) and classification.

- In order to test this part of algorithm, we will launch the application in different environments (quiet, normal, noisy) and strike a key and see if the app detects the stroke correctly. Since it is a black box testing and there are many uncertain situations, we will test it for many times. The ideal result is that the app is able to detect most of the strokes under different environments. The specific parameters (e.g. the dB value of background noise) is to be determined.
- We will use some pre-recorded sound files to verify the correctness of Fourier transformation. The FT results from the app should be the same as the results from a script in MATLAB.
- Normally, before we start this test, the former tests should be passed so we can assume that the detection and FT process are correct. Then using the same prerecorded file and the "correct" label (since we know which key is pressed when we record our sample files) to verify the correctness of classification. According to the sample files, this application should be able to classify the sound correctly (has the same result as the label).

IX. INTERACTION WITH PLATFORM

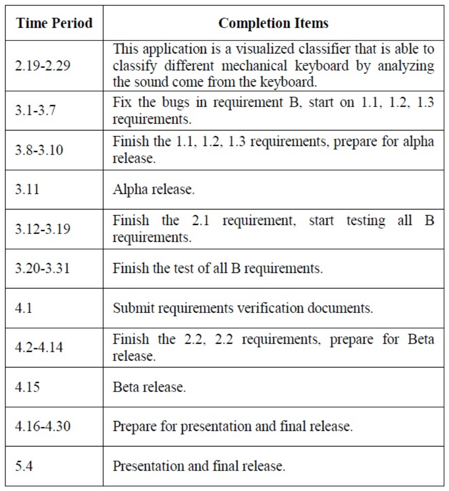
Since the application is developed in iOS 9 platform, we will use libraries from iOS 9 as interface. The only sensor we will use is the microphone. We will use libraries including but not limited to AV Foundation, AVKit, UIKit, etc.

X. TASK ALLOCATION & BREAKDOWN

Shengxiang Zhu will mainly take care of the UI development, and Ce Wang will mainly take care of the audio process algorithm.

XI. TIME FOR COMPLETION

The timeline of our project in as follows:



Time Period	Completion Items
2.19-2.29	This application is a visualized classifier that is able to classify different mechanical keyboard by analyzing the sound come from the keyboard.
3.1-3.7	Fix the bugs in requirement B, start on 1.1, 1.2, 1.3 requirements.
3.8-3.10	Finish the 1.1, 1.2, 1.3 requirements, prepare for alpha release.
3.11	Alpha release.
3.12-3.19	Finish the 2.1 requirement, start testing all B requirements.
3.20-3.31	Finish the test of all B requirements.
4.1	Submit requirements verification documents.
4.2-4.14	Finish the 2.2, 2.2 requirements, prepare for Beta release.
4.15	Beta release.
4.16-4.30	Prepare for presentation and final release.
5.4	Presentation and final release.

TABLE 2. TIME FOR COMPLETION.

XII. GLOBAL/SHARED TASKS AND EXPERIENCES

We are a 2-member team, so we will work together closely. Shengxiang Zhu will develop the part of UI while helping Ce Wang with algorithm and testing. Meanwhile, Ce Wang will use my UI to implement the algorithm and help Shengxiang Zhu to test UI.