SOFTWARE SYSTEM FOR AUDIO SAMPLE RECOGNITION

A THESIS SUBMITTED TO THE GRADUATE SCHOOL OF APPLIED SCIENCES OF NEAR EAST UNIVERSITY

By
TIJESUNIMI E. OGIDAN

In Partial Fulfillment of The Requirements for the Degree of Master of Science in Software Engineering

NICOSIA, 2017

SOFTWARE SYSTEM FOR AUDIO SAMPLE RECOGNITION

A THESIS SUBMITTED TO THE GRADUATE SCHOOL OF APPLIED SCIENCES OF NEAR EAST UNIVERSITY

By
TIJESUNIMI E. OGIDAN

In Partial Fulfillment of The Requirements for the Degree of Master of Science in Software Engineering

NICOSIA, 2017

Tijesunimi Ezekiel OGIDAN: DEVELOPING A SOFTWARE SYSTEM FOR AUDIO SAMPLE RECOGNITION

Approval of Director of Graduate School of Applied Sciences

Prof. Dr. Nadire ÇAVUŞ

We certify this thesis is satisfactory for the award of the degree of Master of Science in Software Engineering

Examining Committee in Charge:

Asst. Prof. Dr. Yoney KIRSAL EVER	Committee Chairman, Software Engineering Department, NEU
Assist. Prof. Dr. Boran ŞEKEROĞLU	Commitee Member, Software Engineering Department, NEU
Assist. Prof. Dr. Kamil DİMİLİLER	Supervisor, Automotive Engineering Department, NEU

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Last name: Tijesunimi E. OGIDAN

Signature:

Date:

To my brother, David...

ACKNOWLEDGEMENTS

I owe the success of this thesis to my principal supervisor, Assist. Prof. Dr. Kamil Dimililer, for his patience, support and guidance through the project. I appreciate his efforts in pointing me in the right direction and helping me through the challenges brought on by this project. This work would not have been possible without him.

I also want to thank Assist. Prof. Dr. Boran Şekeroğlu for his support and help throughout the course of my masters education at this university. I appreciate him for being there to help me when I had questions. My thanks also goes to Assist. Prof. Dr. Yoney Kirsal Ever for her support and unparalleld kindness throughout my masters education and during the course of the thesis.

Finally, I would like to thank my family, but near and far, as well as my friends, who altogether form the best support system anyone could wish for.

Thank you all, and God bless.

ABSTRACT

This thesis takes us through the design and development of an audio recognition application that allows a user to identify songs played in their immediate environment. It uses an algorithm that allows it to recognize audio even with surrounding noise and interference with high accuracy and efficiency.

The system uses the idea of audio fingerprinting to extract features from a song in the form of a spectrogram. The features are then processed to represent the signal as an array of hashes. These unique hashes are then stored in a database to form a pool of learned songs. When a user hears a song they would like to identify, they only need to activate the system and specify how long they would like the application to listen. To identify the played sample, features are also extracted from the played sample and are processed as well to represent the sample as an array of hashes. This hashes are then run through the database, and combinatorial hashing is used to find a match.

Combinational hashing is used to speed up the matching process because of the sparseness of the peaks in the constellation map.

Python programming language was used through the project because it is a simple language and because of the helpful open-source python tools available.

Keywords: Machine hearing; pattern recognition; object recognition; spectrogram; fingerprinting; hashing

ÖZET

Bu tez, bir kullanıcının kendi çevrelerinde oynatılan şarkıları tanımlamasına olanak tanıyan ses tanıma uygulamasının tasarımı ve geliştirilmesi sürecini ele alıyor. Çevredeki gürültüyü ve parazitliliği yüksek doğruluk ve verimlilikle bile tanımasını sağlayan bir algoritma kullanıyor.

Sistem, bir spektrogram formunda bir şarkının özelliklerini çıkarmak için ses parmak izi alma fikrini kullanıyor. Daha sonra özellikler, sinyali bir karma dizisi olarak göstermek için işlenir. Bu eşsiz karmalar, daha sonra öğrenilen bir şarkı havuzu oluşturmak için bir veritabanında saklanır. Bir kullanıcı tanımlamak istedikleri bir şarkıyı dinlediğinde, yalnızca sistemi etkinleştirmesi ve uygulamanın ne kadar süre dinlemesini istediğini belirtmesi gerekir. Oynatılan örneğin tanımlanması için, çalınan örnekten de özellikler çıkarılır ve örneği karma dizisi olarak göstermek için işlenir. Bu karma veriler veritabanı üzerinden çalıştırılır ve bir eşleşme bulmak için kombinasyonel karma kullanılır.

Takımyıldız haritasında tepe noktalarının seyrek olması nedeniyle, eşleme sürecini hızlandırmak için kombinasyonel karma kullanılır.

Python programlama dili, basit bir dil olduğundan ve yararlı açık kaynaklı python araçlarından dolayı proje boyunca kullanılmıştır.

Anahtar Kelimeler: Makine işitme; Örüntü tanıma; nesne tanıma; spektrogram; parmak izi oluşturma; ayrık fourier dönüşümü; karma

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF ABBREVIATIONS

IFPI:	International Federation of the Phonographic Industry
RIAA:	Recording Industry Association of America
CBID:	Content-Based audio Identification
PRS:	Pattern Recognition System
PR:	Pattern Recognition
OR:	Object Recognition
ADC:	Analog to Digital Converter
FPS:	Frames per second
FFT:	Fast Fourier Transform

LIST OF TABLES

CHAPTER 1

INTRODUCTION

Recognition applications are an interesting part of present day technology. They entail a phone or computer recognizing human speech, motion, visual cues, or audio, and performing certain operations based on or with the input. Some popular applications of this technology fingerprint scanners and image processors. Audio recognition technology is a fast growing aspect of recognition technology with applications including copyright compliance, licensing, content-based integrity verification and content-based audio identification.

The process of audio sample recognition involves, first of all processing a collection of audio files to generate audio fingerprints of each of the files. These fingerprints would then be used to populate a database against which an audio fingerprint of the audio file we need information on shall be compared to find a match.

Audio fingerprints are digital summaries deterministically generated from audio signals. Audio fingerprinting systems make use of audio fingerprints to represent audio objects for comparison. An audio fingerprint is a compact, perceptual digest from a raw audio signal that can be stored in a database so that pairs of tracks and audio samples can be matched. Considering the idea of pattern matching, the ability to extract the essence of a class of objects while keeping its characteristics is a major tool in any recognition system.

Audio fingerprinting technologies have gained popularity due to the fact that they allow audio recognition regardless of the audio samples format and without need of meta-data or watemark embedding. However, the ability to accurately identify audio samples depends on the matching algorithm. The different approaches to fingerprinting are usually described with different rationales and terminology depending on the background: pattern matching, multimedia (music) information retrieval or cryptography (robust hashing).

Some popular commercial applications of audio recognition technology include Shazam, SoundHound and Bird Song ID – an application used to identify bird species based on the sounds they make.

1.1 Thesis Problem

At times when a person hears a nice song they like as part of a movie's sound track, in an advertisement, at a public place, etc. and wants to know the name, or other information about the song, being able to simply ask someone about it can really come in handy. However, this is not always the case.

Audio recognition would work as a good solution to this problem. It provides the means to recognize and identify a wide range of songs with only a 10 second segment of the song to go by.

Furthermore, in applications where copyright compliance and licensing are necessary, having technology, such as this, that can easily recognize copyrighted audio material would be very helpful. The algorithm used in this technology can be adopted into such applications and used to process audio signals and compare them against a database of copyrighted material.

1.2 The Aim of the Thesis

The aim of this thesis is to develop an audio recognition system that is robust enough to make accurate matches in the presence of a reasonable amount of noise and interference. The system would also be able to recognize a song based on samples as short as one second recorded from any point in the actual song. It would also recognize audio samples of any format.

Finally, we aim to ensure that the computational cost at each stage of the learning and recognition process is reduced as much as possible to ensure that the system is as efficient as possible. The size of the fingerprints are also put into consideration to make sure that the memory used is as low as possible.

1.3 The Importance of the Thesis

From an academic point of view, this thesis would serve as a great source of exposure into techniques for audio recognition using open source tools with special focus on the idea and techniques for audio fingerprinting.

Furthermore, the techniques presented by this thesis, and the system itself can be initiated into systems used for copyright compliance and licensing.

Some other uses of an audio recognition systems like the one developed in this thesis include :

- Customers at record stores can get the songs that they have absolutely no information about by humming into an audio recognition system
- The system could be used to identify songs that contain samples of other songs, and help with understanding how artistes influenced each others arts.
- Suspicious sounds and background noises could be easily spotted in surveillance footage.
- Users can easily identify songs they hear in their immediate environment.

1.4 Limitations of the Study

The system would only be able to recognize audio with a reasonable amount of noise. Because the system cannot differentiate between noise and music, excess noise might overshadow the audio signal and be processed as part of the audio fingerprint and cause a mismatch (false positives) or cause the system not to find a match at all.

This study does not define exclusively how this system can be applied or included into other applications for licensing and copyright compliance, as there are factors, such as the architecture of the host applications, that can affect how it is included.

Finally, the ability of the system to work is based on the assumption that the audio sample to be identified is played at the same speed as the audio file whose fingerprint is stored in the database. If the sample audio to be identified is played at a faster or slower speed, the audio signals would have different frequencies and hence different fingerprints

1.5 Overview of the Thesis

This thesis has been structured to give exposition to the general idea of audio recognition and then show tools and steps I employed in developing my system for audio recognition.

The first chapter gives an introduction to the idea and concept of a recognition application then branches into audio sample recognition specifically. It also goes on to explain the idea of audio fingerprinting and how it works. Its applications are also discussed.

The chapter continues by explaining the aim and importance of this thesis particularly and ends with a brief discussion of the limitations of the design as well as this exposition.

The second chapter discusses ongoing research and development and their contribution to the subject of audio recognition with emphasis on the practice of audio fingerprinting. It also points out a range of views on the subject and suggests new points of views on the subject.

In the third chapter, we look at sound signals and its properties. We analyze these properties to see how they affect our perception of sound, and we discuss ways to manage or manipulate these properties to help with solving the problem.

The fourth chapter brings us back to this particular project and highlights the tools used in the system and the reasons for which each tools were picked. It also explains the design approach used in the development of the system and how the tools and design are expected to affect the results at the end.

The fifth chapter gives a step by step walk through of the development process and steps taken.

In the sixth chapter, we run tests to measure the efficiency of the system. The results are analyzed and discussed and used to draw conclusions which are elaborated in the seventh chapter.

Further work and development on this system are also discussed in the seventh chapter of this thesis.

Code for the major functionality of the system are made available in the appendix section.

CHAPTER 2

LITERATURE REVIEW

The music industry worldwide is clearly one of the biggest industries in entertainment. With thousands of songs being released yearly, and more artistes coming up, this is not a growth that would be slowing down anytime soon. In a paper (Grosche, Muller, &Serra,2012) it is pointed out that with the increase in amount of songs available to the public, methods for exploring music collections is needed.

2.1 The Need for Content-based Audio Retrieval

The way that music is stored and used has been greatly revolutionized through the years. The amount of music available to the general public has also multiplied greatly. This new developments have therefore created the need for newer methods of music retrieval.

Furthermore, it has become a common understanding within the field of audio engineering that depending on words that describe audio content for audio retrieval is not always reliable. In a paper (Grosche et al, 2012) It is explained that the possibility that there might be no textual description available for the sample audio points out the need for a content based retrieval strategy that would only utilize the raw audio material. In his PhD dissertation, Pedro Cano (2006) also pointed out the fact that such data provided by humans is error prone.

2.2 Facets of Content-based Music Retrieval

In the paper by Grosche et al (2012), Content-based music retrieval is split into four facets of applications with varying requirements of granularity and specificity . These facets are listed below with the applications or uses they entail.

2.2.1 Version Identification

This facet involves identifying a different version of the same song, thus it would require the entire song to identify the differences (lower granularity), which are expected to be slight. This facet However requires a reasonable level of specificity.

This facet entails:

- Remix or Remaster retrieval
- Cover songs detection
- Version Identification

2.2.2 Audio Identification

This facet involves identifying a specific song based on a short clip (5 to 10 seconds) of the song. This facet requires high specificity and just a fragment (low granularity) of the song.

This facet entails:

- Plagiarism detection
- Copyright monitoring
- Audio fingerprinting
- Audio identification

This project focuses mainly on this facet.

2.2.3 Audio Matching

This facet has to do with identifying music that share similarities. It works based on a query-by-example paradigm. This facet requires a moderate amount of specificity and just a fragment of the song.

This facet entails:

- Variation/motif discovery
- Musical quotations discovery
- Audio matching

2.2.4 Category-based Retrieval

This facet is a broader aspect that entails identifying songs that can be classified together under a specified category. This facet requires low specificity and any amount of granularity.

This facet entails:

- Music/speech segmentation
- Year/epoch discovery
- Key/mode discovery
- Loudness-based retrieval
- Tag/metadata inference
- Mood classification
- Genre/style similarity
- Recommendation
- Instrument-based retrieval

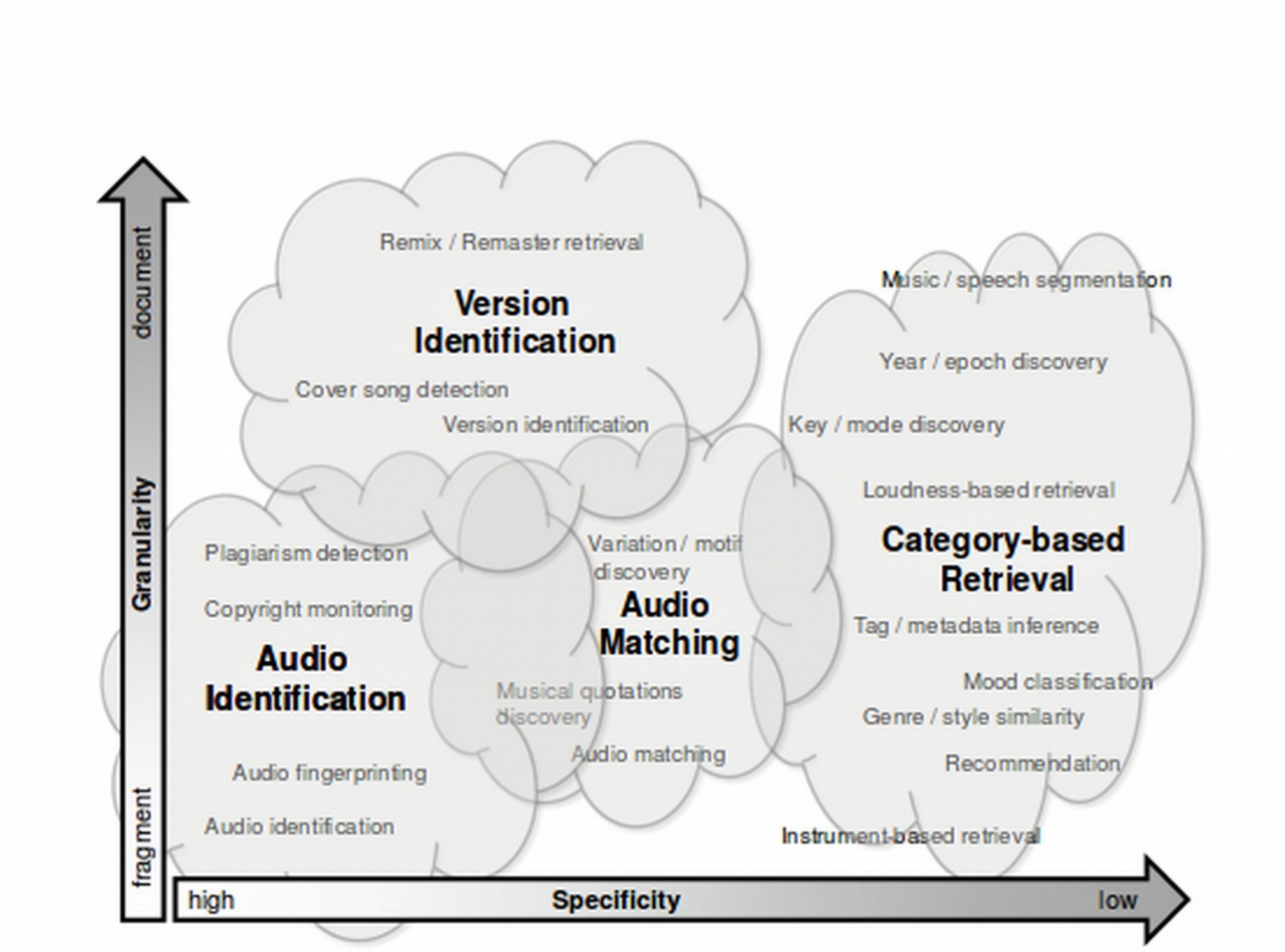

Fig 2.1: Specificity and granularity graph showing the various aspects of content-based Audio retrieval

2.3 Audio Fingerprinting Algorithms

Audio identification is the most popular content-based music retrieval application as well as the most commercialized. This is where the concept of audio fingerprinting falls as the audio fingerprints are what are compared to identify matches in the identification process.

The IFPI and the RIAA in their *Request for Information on Audio Fingerprinting Technologies*, (RIAA, 2001) tried to analyze and evaluate many identification systems, and enumerated the following requirements for fingerprinting algorithms.

- **Accuracy:** This is the number of correct identifications, missed identifications and wrong identifications (false positives)

- **Reliability:** In play list generation for copyright enforcement organizations, methods used to assess that a query is present or not in the repository are necessary. In these cases, a song should not be identified as a match if it has not yet been broadcast.

- **Robustness:** This refers to the ability of the system to accurately identify an item, regardless of the level of compression, distortion, or interference. Some common things that cause alterations in audio are pitching, equalization, background noise, and audio coders like GSM and MP3.

- **Granularity:** This refers to the ability to identify a whole title from samples that are only a few seconds long. The algorithm would have to deal with the lack of synchronization between the sample's fingerprint and the fingerprints stored in the database. This makes the search more complex, as the algorithm would need to compare audio in all possible alignments.

- **Security:** This refers to the level of vulnerability of the system to external and intentional tampering. Here, the manipulations that are dealt with are particularly made to deceive the algorithm.

- **Versatility:** This refers to the ability to identify audio of any format. A good level of versatility would mean the database can be used for different applications.

- **Scalabiltity:** This refers to the systems ability to work with very large databases or a large number of concurrent identifications. This could affect the accuracy of the system as well as its complexity.

- **Complexity:** This has to do with the computational cost of the fingerprint extraction, the size of the fingerprint, the complexity of the search, the complexity of the fingerprint comparison, the cost of adding new items to the database, and so on.

- **Fragility:** In some applications, such as content-integrity verification systems, the detection of changes in the content is required. This is in opposition to the robustness criteria, as in this case, the fingerprint is expected to be robust for only transformations that require the preserved content.

Often, there is a trade-off between the criteria.

Research and work in the field of audio sample recognition has its roots in a good understanding of signals and signal processing techniques and methods as well as pattern recognition. in relation to the newly emerged field of machine hearing. John Treichler (2009) mentioned recognition as one of the signal processing areas that are in the midst of a long trajectory of development

2.4 Machine Hearing

Most of the research efforts within the field of sound analysis has focused on speech and music,However these areas of study can be housed under the bigger research field of machine hearing. This field would lead to the creation of systems that can differentiate between speech, music and noise and even be able to tell the direction that certain sounds are coming from. The systems would also have the ability to organize what they hear. They would also be able to identify music genres and styles and much more.

In a column by Richard F. Lyon (2010) in the IEEE signal processing magazine, he split the machine hearing system into four consisting modules:

- **A Peripheral Analyzer:** This is a sound analysis front end that would do the same job that the cochlea of the human ear does, by producing a half wave rectified representation of the input sound waves while preserving the structure of the wave.

- **Auditory image generators:** This aspect would generate an image representation of the input waves.

- **A feature extraction module:** This aspect takes the images from the previous module and extracts multi scale features that will be used in the next modulele.

- **A classifier or decision module:** Depending on the application of the machine hearing system, machine learning methods and techniques are used by the system in learning to map from the extracted features to the expected decisions. This can work in a single step like single-layer perceptrons or use multiple layers of internal structure.

The modules Lyon (2010) enumerated are identical to the traditional methods used in other deep learning fields like machine vision.

CHAPTER 3

SOUND SIGNALS

Now that we understand the concept of searching for audio based on their content and we have a basic understanding of fingerprinting, we need to understand the composition and structure of sound to help us understand how to handle or process it.

3.1 Sound Waves

A sound wave is a representation of varying pressures in a medium such as water or air over a period of time. They are formed by the vibration of an object, the source of the sound, which causes the air around it to vibrate. These vibrations are then perceived as sound.

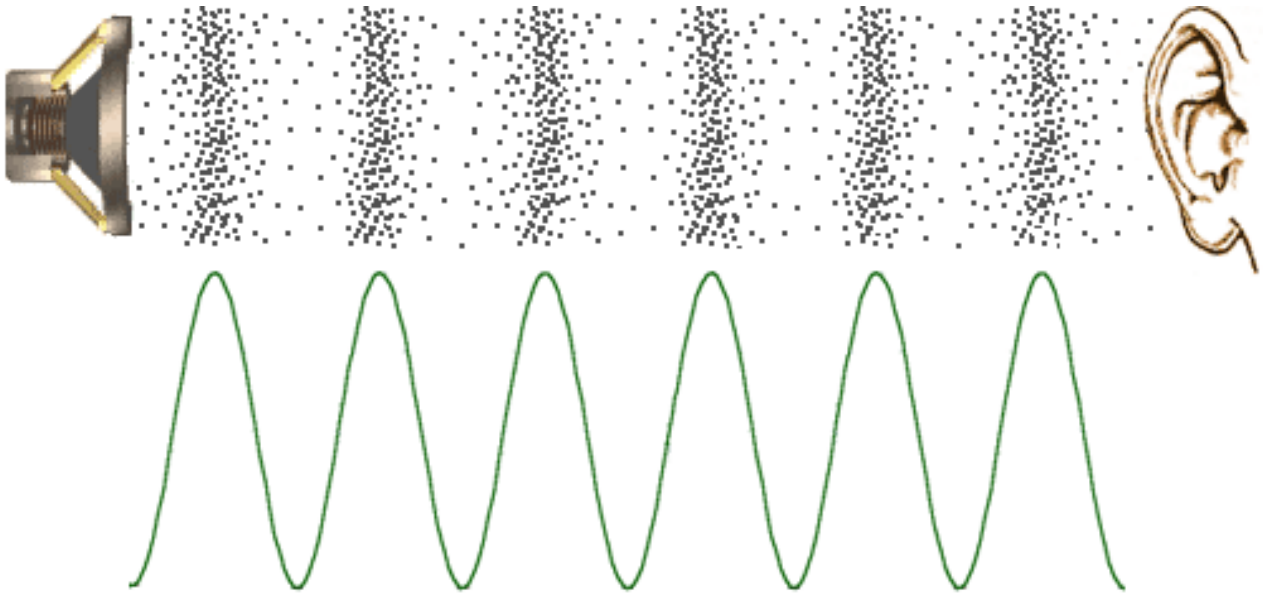

Fig 3.1: sound as changes in air pressure and sound as a wave

Wave forms of sound waves are often represented graphically to help us analyze and work with sound waves from a mathematical point of view. The image below represents a tone at a fixed frequency.

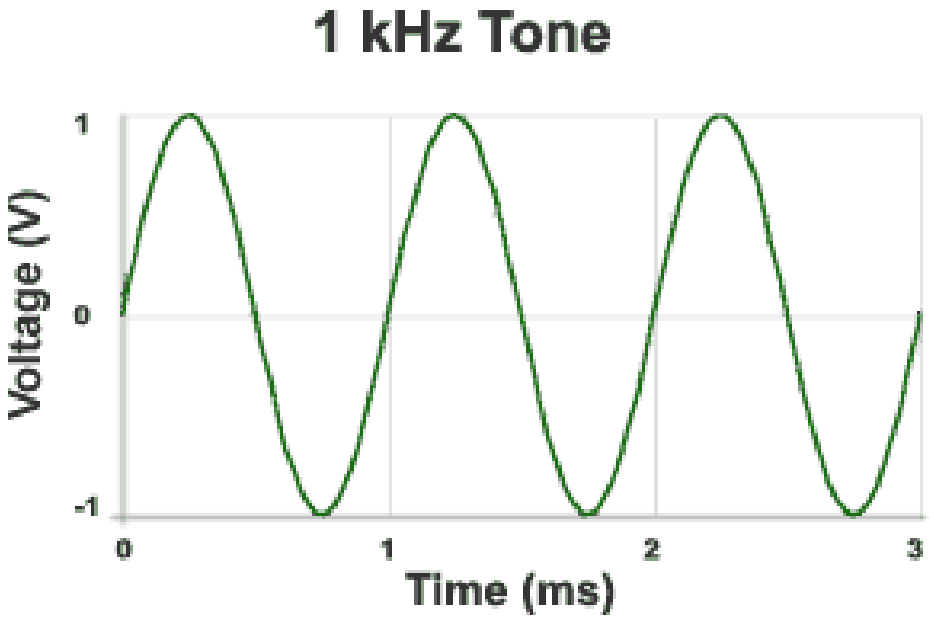

Fig 3.2: A wave form

3.2 The Human Ear

For the human ear to hear, sound waves are collected in the outer ear (auricle or pinna). Then they move down the ear canal and hit the eardrum, causing it to vibrate. These vibrations then get transmitted to the inner ear by the bones of the middle ear. The inner ear plays changes the vibrations created by the sound waves to electrical impulses, which can be recognized and decoded by the brain. The inner ear or cochlea is filled with fluid. The movement of bones in middle ear causes this fluid to move. This movement then causes the tiny hairs cells lining the cochlea to move back and forth. The movement of these hair cells causes an electrical signal to be sent to the brain. The brain receives these impulses in its hearing centers and interprets them as a type of sound.

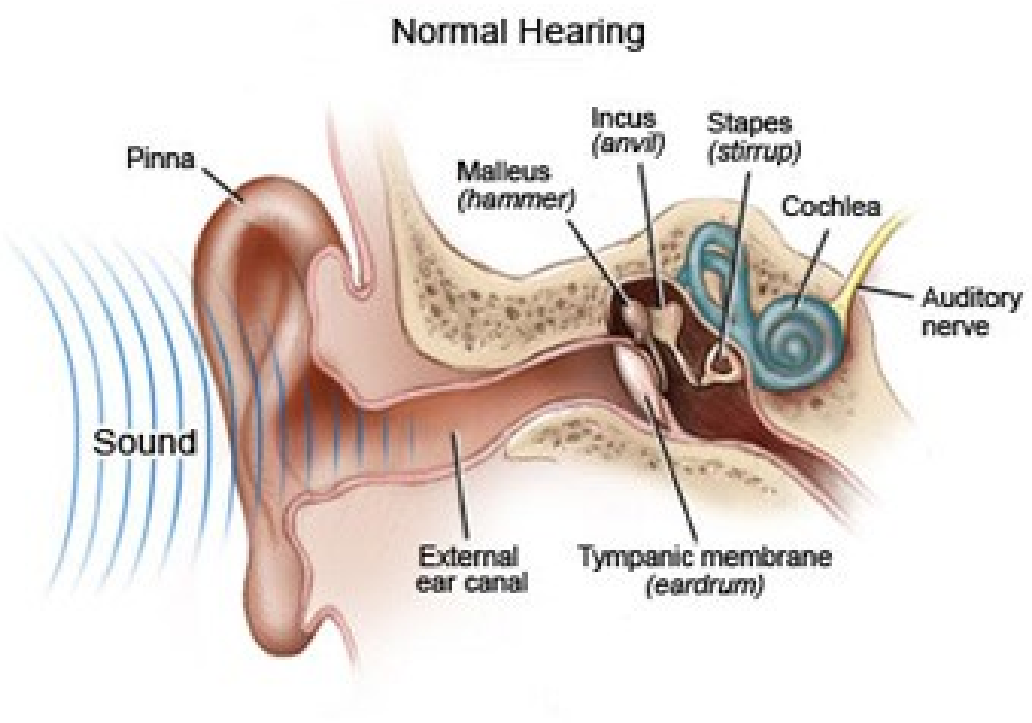

Fig 3.3: How the ear works

3.3 Properties of a Sound Wave

Now that we understand how what sound is and how it is represented as a wave, we need to understand the attributes that are used to distinguish between individual sounds. The attributes discussed below are what would be analyzed and processed within the system.

3.3.1 Amplitude

The amplitude of a sound signal is perceived as the power or loudness of the sound. Higher amplitudes cause more displacement of the particles of its medium. In a sound waveform, the amplitude is represented b the height of the sound wave.

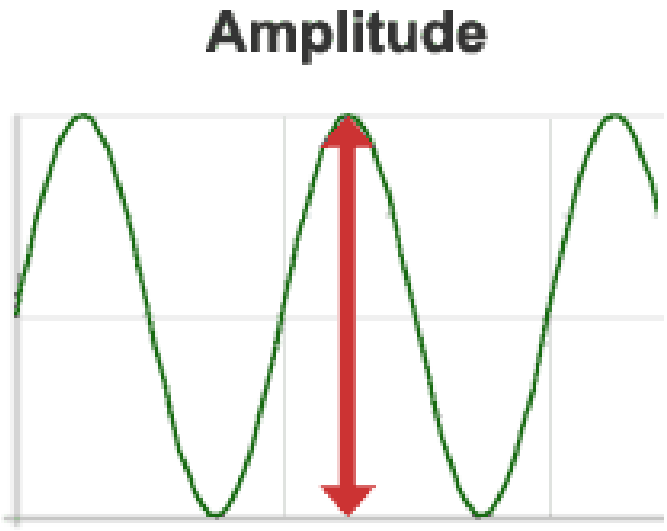

Fig 3.4: waveform showing amplitude

3.3.2 Frequency

The Frequency of a sound signal is perceived as the pitch of the sound. In a waveform, the frequency is measured as the number of cycles per second. It could more simply be defined as the speed of the vibration of the medium. Frequency is measured in Hertz (Hz).

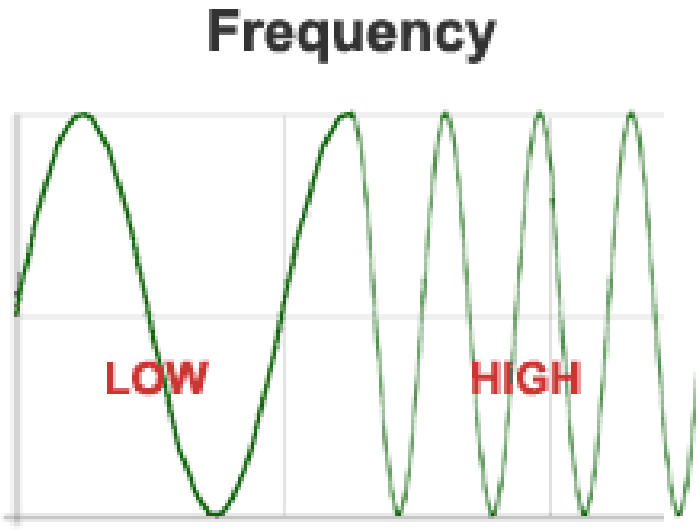

Fig 3.5: waveform showing low and high frequencies

3.3.3 Wavelength

This is perceived as the distance which the sound travels during one period or cycle. Graphically, this is represented as the length of the wave.

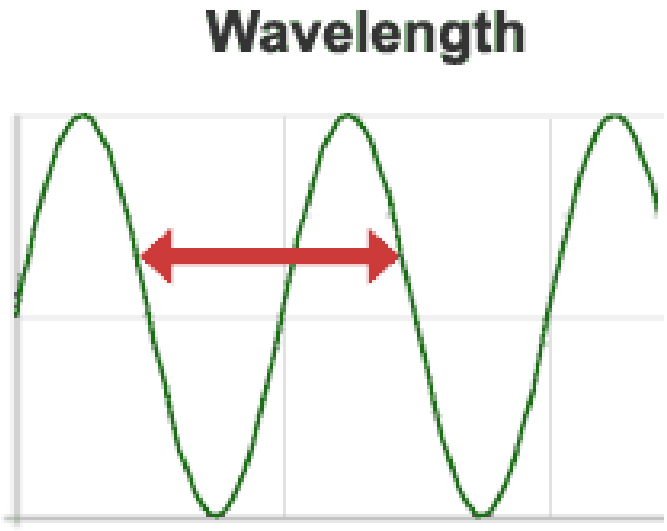

Fig 3.6: waveform showing wavelength

3.4 Sampling a Signal

This concept of encoding or converting sound waves to a form that is understandable by the brain is used in recording devices as well. Sound waves are actually continuous pressure signals, but in digital systems, sound is stored and processed as a stream of numbers (samples) that represent air pressure at particular times. The number of samples taken per second is the sample rate.

In recording devices, a microphone is connected to an analog to digital Converter (ADC) to convert a continuous analog signal to discrete digital samples. This is done by capturing digital values that represents amplitudes of the signal. These samples are then stored or processed. The discrete digital signals could also be passed through a digital to analog converter (DAC) connected to a speaker to

play the sound. Fig .1 shows the framework of a recording device and the representation of the sound wave as analog and digital (sampled) signals.

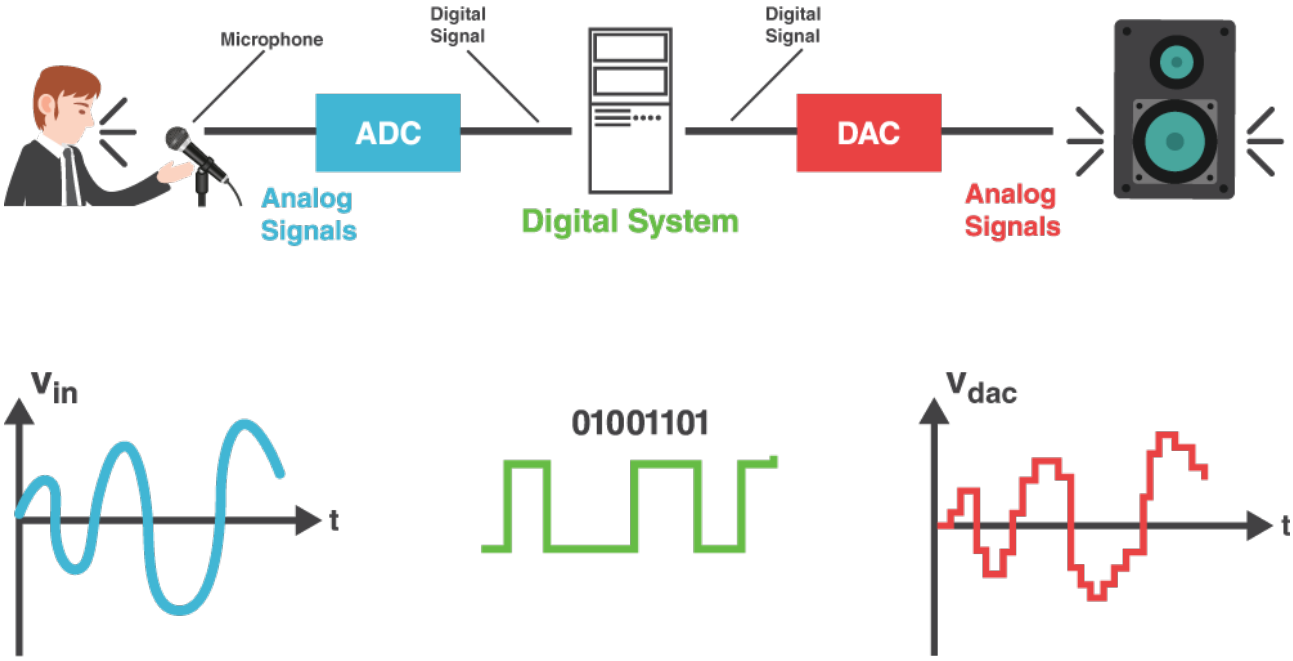

Fig 3.7: Signal sampling in a recording device

3.4.1 Sample Rate

Sample rate is the number of audio samples of an analog signal carried per second for conversion into a digital signal.

The Nyquist-Shannon theorem gives us a condition for sample rates that allows a digital signal to represent all information from an analog signal of fixed bandwidth. It states that the sampling rate for the analog signal has to be greater than twice the frequency of the analog signal. Keeping this in mind, and the fact that the human ear can detect frequencies between 20Hz and 20,000Hz, the sampling rate of 44,100Hz is used mostly in sampling for compact discs and MPEG-1 audio. 48,000Hz is mostly used for video and 88,200Hz or 96,000Hz is sometimes used for classical or complex music.

This sampling rate was first chosen by Sony, because it was easy to record on video equipment running at 25 or 30 FPS .

3.4.2 Bits Per Sample

The sampled audio file represents the sound signal as bits in each sample, so bits per sample is used to express how many bits are used to represent a sample. A higher number of bits means a higher quality for the sampled audio, but also more memory for storage, so there is a trade off. The generally used values for audio sampling are 8 (2556 values) and 16 (65536).

CHAPTER 4

SOFTWARE TOOLS AND DESIGN

In the development of this system, I have used mostly python open source libraries and software packages. The software packages used are listed below.

4.1 Tools

- **MySQLdb:** This was used to interface with the MySQL database. MySQLdb is an interface to the MySQL database server that provides the python database API. It serves as a wrapper round _mysql to make it compatible with the python database API.

- **PyAudio:** This was used to grab audio from the microphone. PyAudio is a library that provides python bindings for PortAudio (the cross-platform, open-source audio I/O library written in C). The figure below shows PyAudio being used to capture sounds from a baby and relaying it to a client browser after processing.

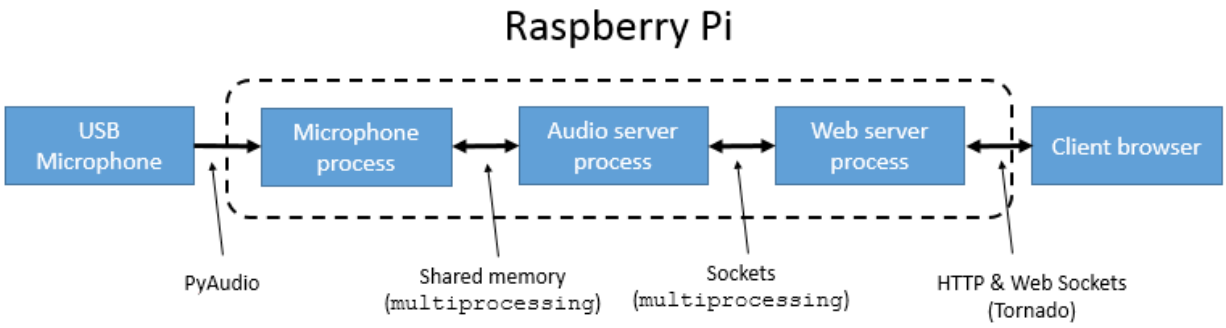

Fig 4.1: PyAudio being used in a raspberry pi baby sleep monitor

- **Pydub:** This was used to process audio files. Pydub is a python wrapper for FFmpeg (the cross-platform library for recording, converting and streaming audio and video). FFmpeg is a

free software framework used to transcode media files. It contains other libraries and codecs packed together as one software package.

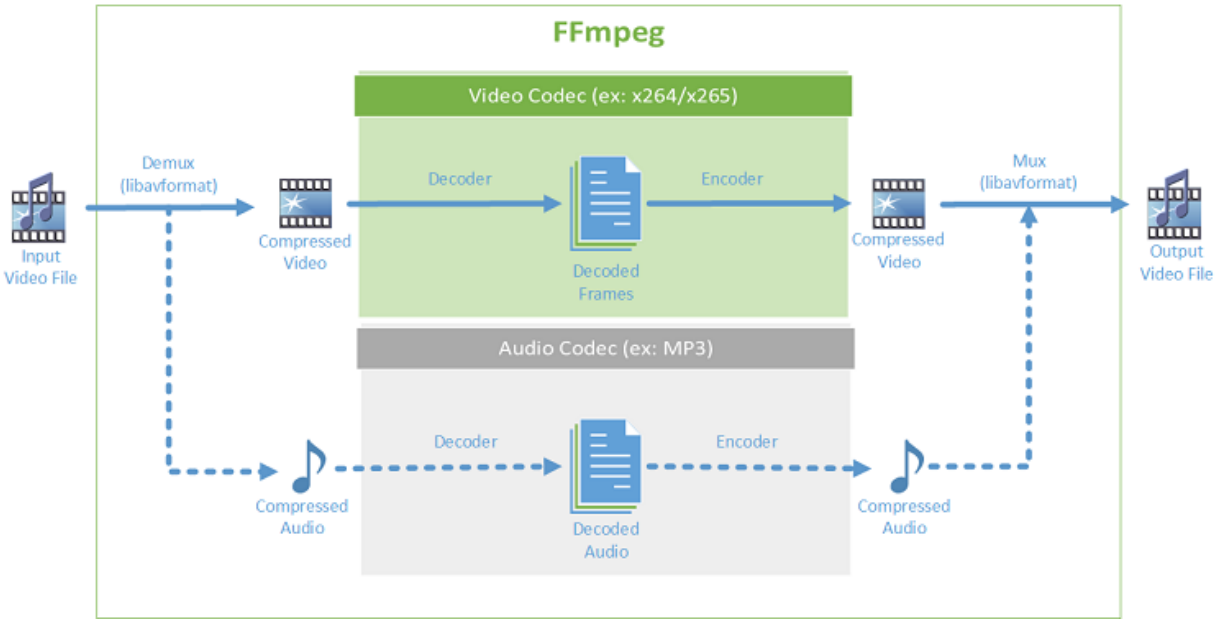

Fig4.2: FFmpeg transcoding highl-level flowchart

- **FFmpeg:** FFmpeg was also used by itself, to convert audio files to .wav format.

- **NumPy:** This was used to take the FFT of audio signals. NumPy is used for scientific computing in python. It contains;

    - An N-dimensional array object
    - Sophisticated functions
    - Tools that allow integration of C/C++ code
    - Linear algebra, fourier transform and random number capabilities.

- **SciPy:** This was used for peak finding on the spectrograms. SciPy is a python based collection of open-source software for mathematics, science and engineering. Some of the packages contained in sciPy include;

- NumPy
- SciPy library (basic library used for scientific computations)
- Matplotlib
- Ipython

- **Matplotlib:** This was used to plot the spectrograms and other graphs. Matplotlib is a Python library for 2D plotting that produces publication quality figures in a variety of formats and interactive environments across platforms. It can be used to generate plots, histograms, bar charts, scatterplots, error charts, power spectra, etc.

4.2 Design

Looking at the problem, it becomes clear that the functionality of the system has to be split into two. Firstly, there is the learning functionality of the system, where the problem of learning or memorizing the songs is dealt with, and then there is the recognition functionality of the system, where the problem of recognizing the song that is being played is dealt with.

However, within these two functions of the system, there are a couple of things to be considered. They are discussed below.

4.2.1 Learning Function

For the system to be able to learn the songs, we have to decide on a way to represent each song in the database uniquely. One of the best methods or approaches for this is Audio fingerprinting.

4.2.1.1 Audio Fingerprinting

Audio fingerprinting is the process of using an algorithm to extract certain features of an audio file as a unique identity of that particular audio file. The extracted features are known as the audio finger print.

To fingerprint this song, we use FFTs over small segments of a song's sample over time, we can create a spectrogram of the song.

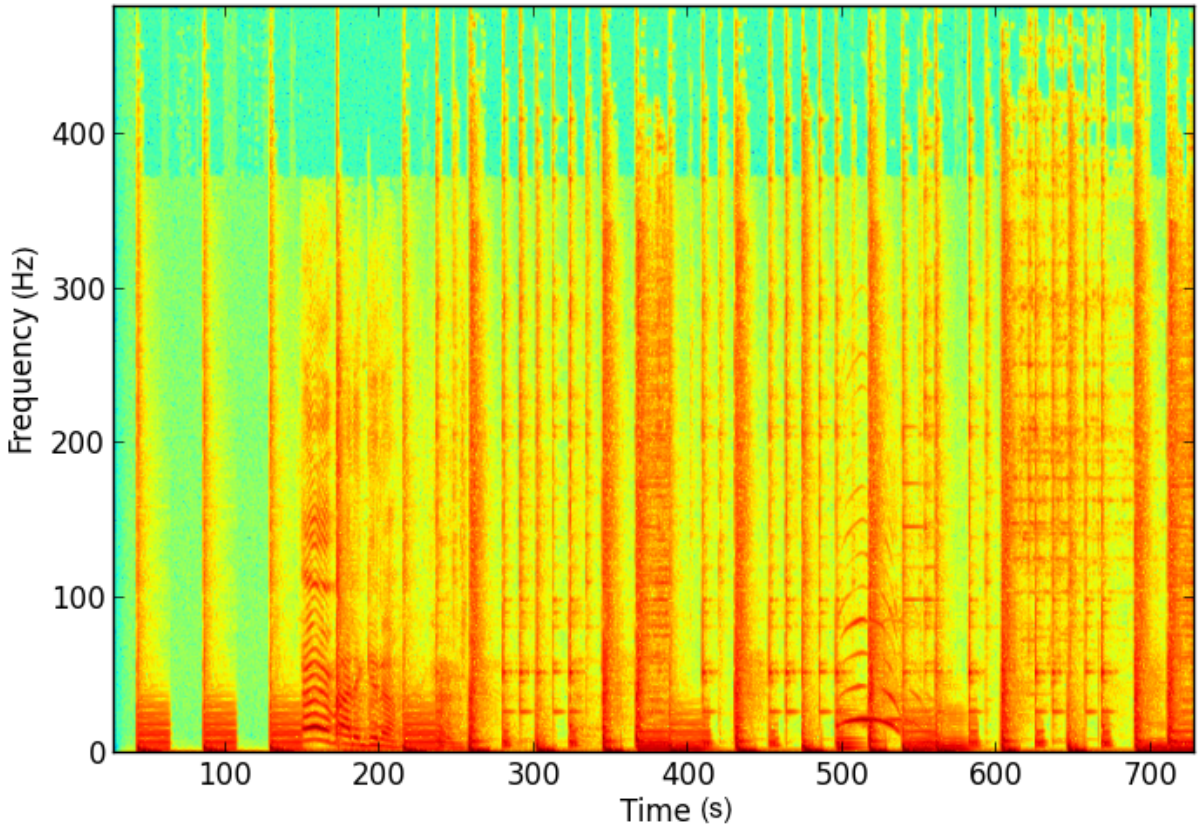

Fig 4.3: Spectrogram of "Blurred Lines" by Robin Thicke

A spectrogram is a visual representation of a signal in the form of a graph of the frequency and time of the signal, showing the amplitude of the signal at particular frequencies. The color shows the real value of these amplitudes, with red being a higher amplitude, and green being a lower amplitude.

As stated earlier, the spectrogram is a visual (graphical) representation of the audio signal, therefore, even background noises get fingerprinted, and their features get extracted too. To prevent this from affecting our results, we have to ensure that our algorithm is robust.

4.2.1.2 Peak Finding

Taking the robustness of the algorithm into consideration, we employ the idea of peak finding to identify peaks in the amplitude. The peak refers to an amplitude with a value higher than those around it. We do this because amplitudes of lower values are more likely to be over powered by noise and external interference.

To find the peaks, we treat the spectrogram as an image and employ the image processing capabilities of SciPy to identify the peaks after using a high pass filter to highlight high amplitudes.

After this processes, we end up with a spectrogram showing a unique set of peak amplitudes that more accurately represent the song. The spectrogram below shows the peak amplitudes in the song “Blurred Lines” by Robin Thicke.

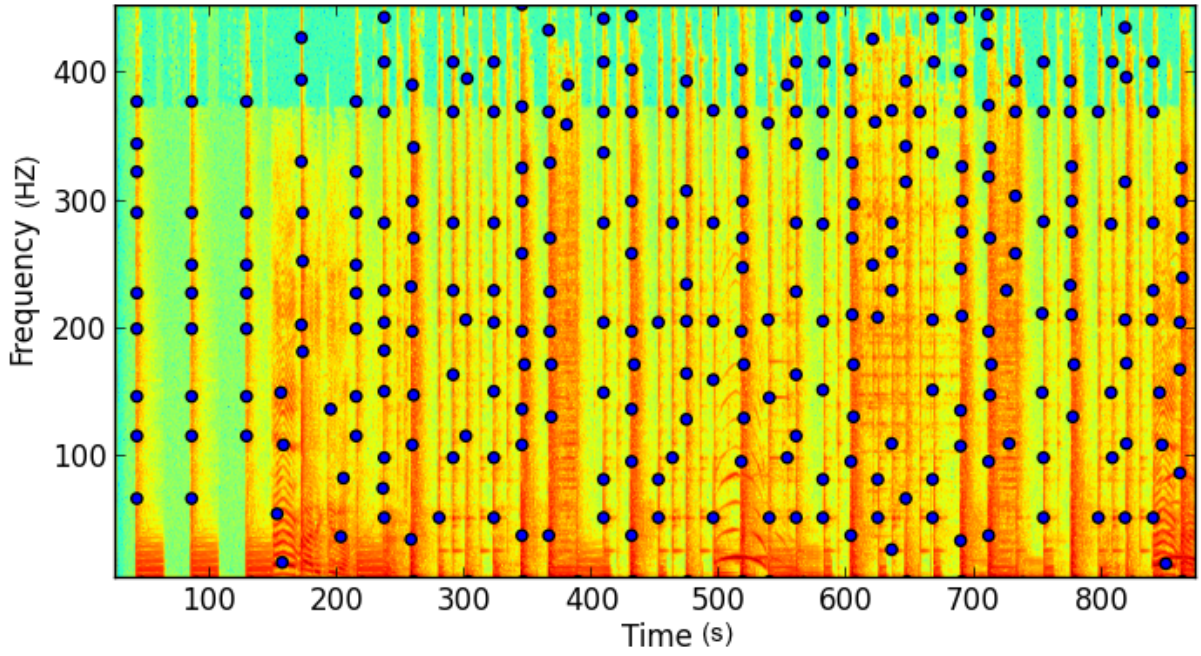

Fig 4.4: Spectrogram of "Blurred Lines" by Robin Thicke showing peak amplitudes

These amplitudes can number up to tens of thousands for a song of about 3 to 4 minutes. These time and frequency pairs (amplitude values) are then converted to discrete integer values by binning them.

4.2.1.3 Hashing

Although we have this representation of the audio signal we are left with another problem. Since the process of binning has reduced the information of the peaks from infinite to finite, we would most probably and quite often find the same peaks in two different songs. Therefore, to avoid mismatches and false positives, we combine the peaks to form fingerprints using a hash function.

A hash function by definition is a function that takes data of arbitrary size and maps it to data of fixed size, so this is a very logical fix. This would ensure that very few different inputs would return the same output.

Looking at our spectrogram, we take into consideration the frequencies of the peaks and analyze the time difference between the peaks to help us get a unique fingerprint for the song. Eq (1) describes the hash function.

$$\textit{hash}(\textit{frequencies of peaks, time difference between peaks}) = \textit{fingerprint hash value} \quad \textbf{(1)}$$

The figure below shows a zoomed version of the blurred lines spectrogram representing the frequencies of some peaks and the time difference between them.

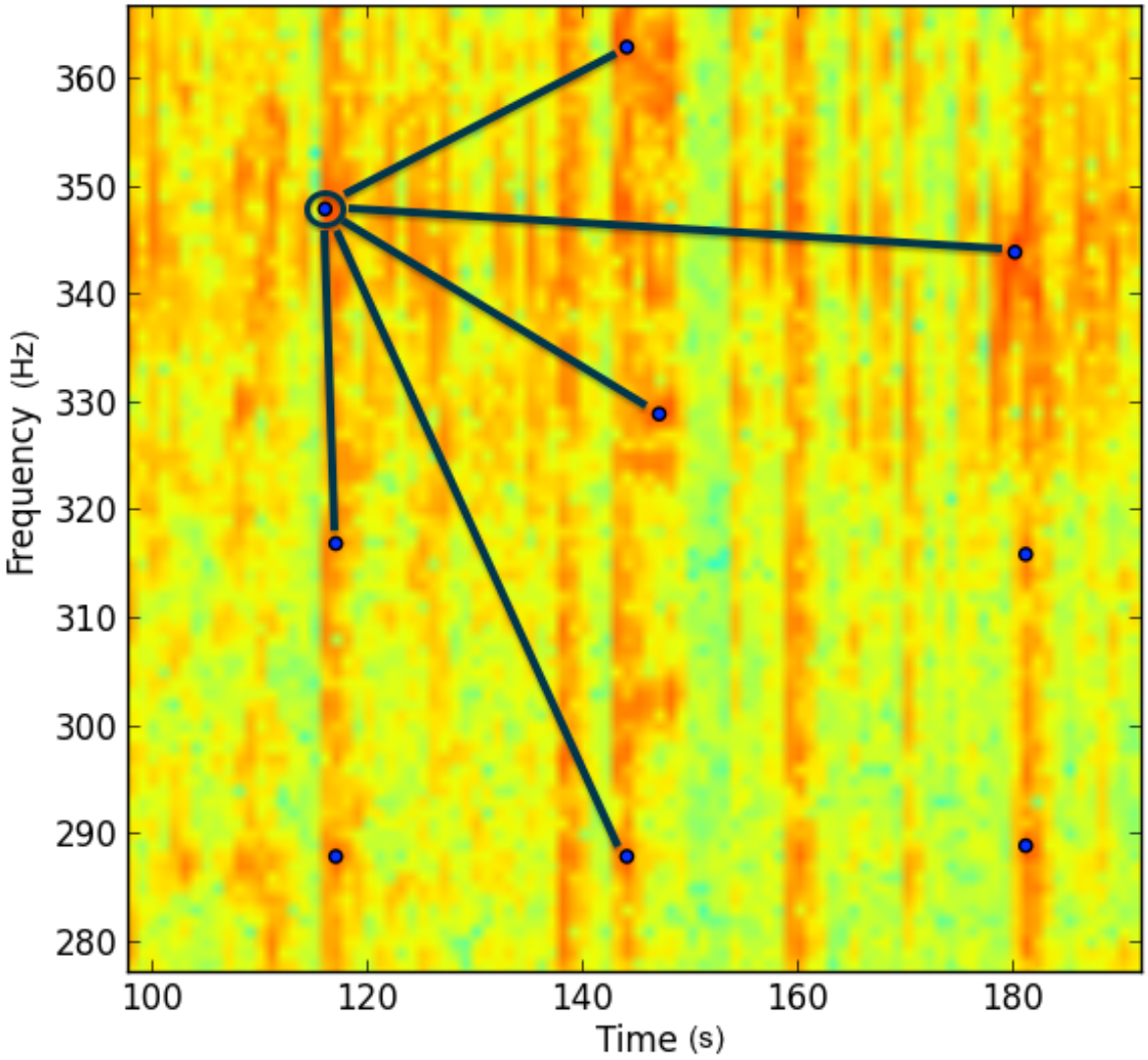

Fig 4.5: Zoomed in spectrogram of "Blurred Lines" by Robin Thicke showing a constellation of peaks

This fingerprint generated by the song is what would be used to populate the database. However, this is not the end.

4.2.1.4 Computing Offset

Considering that we would be playing only a sample of the song during the recognition phase, and this sample would not always be recorded from the start of the song, we have to revisit how the system would handle this input samples. We have to think of the sample (the few

seconds of the song played for recognition) as a subset of the entire original song. Therefore, the hashes we extract from the sample need to have an offset value that is relative to the beginning of the sample.

Even though we cannot determine the offset from the start of the original song to the start of the sample, we do have the relative offsets of the frequencies. Using this, we come up with the following computation for the difference.

Difference in start point = offset from original song in database – offset from recorded sample

All correct matches would always have the same difference.

4.2.2 Recognition function

The recognition phase of the system is a less hectic process. The recorded sample is put through the same processes to extract fingerprints, and the fingerprint hashes are compared at the starting point of the recorded sample on the original song.

CHAPTER 5

SOFTWARE DEVELOPMENT

In this chapter, I would be showing a more technical approach to the solution, and code for major software functionality, more detailed code would be in the appendix section.

5.1 Database

For the database, we have 3 main fields. The song id, the offset to help with getting the start point of the recording, and the hash, which constitutes the songs fingerprint.

5.2 Recording

For the function of capturing audio samples through a laptop microphone, your average modern sound card makes that easily doable. We make use of the PyAudio library in python to do this. We first set the frequency of the sample, the sample size and the chunk size (sample size). We then open the input stream of our sound card and write to a byte array.

5.3 Recognition

Here we first of all equate the upper hash of the recorded sample to the offset, then we search for the upper hash within the fingerprints of each song in the database. For songs that have the same upper hash, the subsequent hashes are compared to narrow down the possible matches.

5.4 Aligning Matches

Here, the matches are aligned using the difference between the start of the song to the start of the recording to match the extracted peaks from the recorded audio to the peaks on the original song.

Details of the source code for this project have not been outlined here for copyright purposes.

CHAPTER 6

RESULTS AND DISCUSSION

6.1 Tests

To test the accuracy of the recognition over time, I tried to recognize music recorded for *n* number of second (with n ranging from 1 second to 5 seconds) for 10 songs. I also took songs of similar genres, so as to allow for similar frequencies and hence similar amplitudes. The table below shows the results of the test.

Time (secs)	Correct matches	Percentage (%)
1s	6 / 10	60
2s	9 / 10	90
3s	10 / 10	100
4s	10 / 10	100
5s	10 / 10	100

Table 6.1: Granularity and accuracy test results

6.2 Discussion

Judging from the results, it is obvious that the system is able to make a more accurate recognition of the sample with more time to record. This is because, given more time, it can capture a longer sample and extract more hashes, thus reducing the possibility of an identical match especially in songs that sound identical.

6.2.1 Robustness

The system is designed to recognize songs even in the presence of noise, by focusing on the strongest amplitudes in the waveform. This is to help it focus on the parts of the input waves that are more certainly a part of the song rather than noise. However, for songs that have generally low amplitudes (like soul music or lofi), the peak amplitudes might not survive heavy noise or interference.

6.2.2 Granularity

The results show that the system has a reasonable amount of accuracy even for short recording times. Therefore, we can say the system has a good level of granularity. However, a longer recording time would ensure that the recognition is more accurate.

The ability of this system to recognize music at all depends on the (reasonable) assumption that the sample song to be recognized is played at the same speed as the original song on the database. If this is not the case, the frequency of the audio would be affected, causing the spectrograms to have different peaks.

6.3 Comparisons

The system developed in this project is able to recognize songs within a database of 10 songs within 2 to 3 seconds. However, as the database increases, the time of search increases. This is due to the fact that the sample has to be compared to each song in the database. In the worst case scenario, the time complexity of the search is O(n) , where *n* is the number of songs in the database.

In bigger systems like Shazam and SoundHound, they make use of server computers to maintain high speeds of recognition. This is especially necessary considering the amount of songs the services host. As of the year 2008, shazam had up to 8 million songs within its database. That figure has grown drastically now.

6.4 Challenges

Challenges were faced mostly in deciding on an approach to solve the problem. A major part of this problem stemmed from finding a way to represent the songs in a database that would allow each representation to be unique. After much deliberation, it was decided that using hashes was a feasible solution.

Furthermore, for this project, a HP 635 laptop was used. The limitations posed by the specifications of the laptop also served as a challenge to the fingerprinting and recognition speed of the system.

CHAPTER 7

CONCLUSIONS AND FUTURE WORK

7.1 Conclusion

With the results from the end of the tests, it is safe to say that the project was a success. Apart from being able to recognize music, this project has helped to highlight an efficient and robust audio fingerprinting algorithm. It has become clear that hash functions can be used to efficiently represent extracted audio features in a database, while ensuring that each song is distinguished even though they may have similar features.

7.2 Future Work

Machine hearing is a new and emerging area of the broad field of machine learning. In this era where we aim to make smarter and independent computers, machine hearing seems like the next big step.

The ability of a computer to hear and understand could be extended to allow it identify music genres when it hears a song, differentiate between music and speech, identify noise, and take commands via speech.

Work on this field from the approach of neural networks would help make more efficient machine hearing systems with a wider range of applications and uses.

Most research into this field has however involved making visual representations of sound and treating it as a machine vision problem, however advancements in pattern recognition could make machine hearing to stand as an individual field over the years.

REFERENCES

Balen, J. V. (2011). *Automatic Recognition of Samples in Musical Audio*. Retrieved from http://mtg.upf.edu/system/files/publications/Van-Balen-Jan-Master-thesis-2011_1.pdf

Cano, P., Batlle, E., Mayer, H., & Neuschmied, H. (Eds.). (2002). *Robust Sound Modeling for Song Detection in Broadcast Audio* (-). New York, USA: Audio Engineering Society.

Grosche, P., Muller, M., & Serra, J., (2012). *Audio Content-Based Music Retrieval.*, Retrieved from http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.403.628&rep=rep1&type=pdf

Cano, P., Batlle, E., Kalker, T., & Haitsma, J. (Eds.). (2002). *A Review of Algorithms for Audio Fingerprinting*. New Jersey, USA: IEEE.

Treichler, J. (2009). *Signal processing: A view of the future, Part I,* IEEE Signal Processing Mag., (pp. 116 -120). New Jersey, USA: IEEE.

Lyon, R.F. (2010). *Machine hearing: An emerging field,* IEEE signal Processing Mag., (pp. 131 – 139). New Jersey, USA: IEEE.

Downey, A. B. (2014). Sounds and Signals. In A. B. Downey (Ed.), *Think DSP: Digital signal processing in python* (pp. 1-12). Needham, Massachusetts, USA: Green Tea Press.

Kpalma, K., & Ronsin, J. (2007). An Overview of Advances of Pattern Recognition Systems in Computer Vision. In G. Obinata, & A. Dutta (Eds.), *Vision Systems: Segmentation and Pattern Recognition* (pp. 169-194). doi:10.5772/4960

Cano, P. (2006). *Content-based audio search: from fingerprinting to semantic audio retrieval*. Retrieved from http://www.tesisenred.net/handle/10803/7543

Diaz, J., & Muniz, R. (2007). *Voice Recognition System*. Retrieved from http://web.mit.edu/6.111/www/s2007/PROJECTS/14/Project14.pdf

Wang, A. L. C. (2007). *An industrial-strength audio search algorithm*. Retrieved from https://www.ee.columbia.edu/~dpwe/papers/Wang03-shazam.pdf

Stinson, D.R. (1994). *Combinatorial techniques for universal hashing.* Journal of Computer and System Sciences, (pp. 337-346). Amsterdam, Netherlands: Elsevier

Ellis, D. (2003). *Audio signal recognition for speech, music, and environmental sounds,* Retrieved from: https://www.ee.columbia.edu/~dpwe/talks/ASA-austin-2003-11.pdf

Ellis, D. (2011). *Environmental sound recognition and classification,* Retrieved from: https://www.ee.columbia.edu/~dpwe/talks/HSCMA-2011-06.pdf

Vacher, M., Fleury, A.,Portet, F., Serignat, J.F., Noury, N., Complete sound and speech recognition system system for health smart homes: Application to the recognition of activities of daily living, Recent Advances in Biomedical Engineering.

APPENDICES

APPENDIX 1

USABILITY AND EFFICIENCY ASSESSMENT OF SHAZAM

A study was done at the Universiti Utara Malaysia (UUM) to gather information of the usability of Shazam. A questionnaire consisting of 10 items was given to 15 students who used Shazam. Out of the students, 27% were Bachelor degree level, 53% were Masters degree level, and 20%% were PhD level.

Each of the 10 items were used to rate the following respectively:

- Comparison to other similar applications
- User interface
- Organization of the user interface
- Ease of use
- Compliance to user requirements
- Organization of buttons
- Ease of understanding the service
- That expected buttons were present
- Ease of navigation
- Overall impression

The results of the study are shown in Table A-1.1.

Items	No. of Participants	Mean	Std. Deviation	Status
Q1	15	4.67	0.82	Very high
Q2	15	4.80	0.41	Very high
Q3	15	4.67	0.62	Very high
Q4	15	4.60	0.51	Very high
Q5	15	4.47	0.64	Very high
Q6	15	4.40	0.91	Very high
Q7	15	4.40	0.91	Very high
Q8	15	4.13	0.74	High
Q9	15	4.53	0.64	Very high
Q10	15	4.73	0.59	Very high

Table A-1.1: Ratings for Shazam