

To explore the mechanism that allows music recognition apps
to work.

Cephas Yeung

28/4/2025

Contents

1	Introduction	4
2	Methodology / Design Choice	5
3	Testing method	7
4	Discussion	8
5	Conclusion	9
	Bibliography	10
<hr/>		
A	Figures	12
B	Testing results	15
C	Screenshots	16
D	Meeting logs	18

Abstract

For the purpose of this project, the replica of a music recognition app is created.

The algorithm first creates an almost unique fingerprint for each audio file from the spectrogram. Which the fingerprints are added to a database as a hashed value.

When record songs through the interface, the algorithm repeats the process of fingerprinting, but instead compare hash value with matched songs.

CHAPTER 1

Introduction

In the modern world, billions and billions of songs are created, and you might want to recognize song that you don't know. This article answer how does modern music recognition app work.

My aim for this research is to understand and replicate how modern day music recognition algorithms work.

From using a very concrete source of a patent created by the original founder and Chief Scientist of Shazam[®] (Now part of Apple Inc[®].) [8].

This source allows a great understanding of the background information of the program.

During the creation process, it was created using different resources to make the resource (e.g: patent, research paper) and also methods to test the success criteria set out by the algorithm.

In conclusion, the program is successful.

CHAPTER 2

Methodology / Design Choice

The artefact is separated in different pieces. (Structure: [Figure A.1](#)).

The software is run inside a container as the patent suggested that although not limited to any particular system, it is preferred for it to be run in a “Distributed system”. But by containerize the program, it allows it to happen.

Next, the program is separated into two parts. The web app part allows user to interact graphically. (Screenshots: [C](#)). And most importantly the main algorithm.

The program first records the audio, this is done through the web app.

And the main algorithm collects the audio file and sent to an algorithm to ”fingerprint” the piece of music. The algorithm consist of spectrogram produced from the SciPy signal library [9] where it uses STFT (Short-time Fourier Transform is a variation of the Fast Fourier transform built for making spectrogram). In this case, Fast Fourier transform are used to convert from time and amplitudes domain to a representation in the frequency domain. It’s illustrated in [Figure A.3](#). [2] And the data are processed using the library of NumPy[5].

An array of data from the spectrogram are being filtered through maximum filter in the SciPy library [9]. The two arrays are compared to use to pick peak points. The peaks are used to construct the pairs. Which is illustrated in [Figure A.4](#) and [Figure A.5](#).

The pairs of points hashed using the default hashing function (Hashing function is a set of things to do to make an output that is always the same length from some other input data. Hash functions and their associated hash tables are used in data storage to access data in a fast time. [6]) from python which generate an almost unique hash values in the program. [Figure A.6](#)

The hash point is than placed into a database alongside with its song name and where the point A’s real time. The song data (metadata) such as author and copyright detail are placed in a different database.

When the user record the sample through the user interface created, the recording sample is sent to the server. It is processed where it applies the same algorithm from above. First it constructs a spectrogram, create and compare it with a filtered spectrogram, peak point are picked and hash values are created, the hash value searched inside the ‘points’ table.

The list of successfully matched song are than going through the process of "scoring". The score calculated by constructing a histogram with the point A's real time value. The score is assigned to each song with the highest value in each histogram. This is because the peak point of some songs are the same for a completely different song. And this includes background noise where the algorithm could mistake it to be a different song.

[10, 7, 11]

CHAPTER 3

Testing method

The program is tested using the method described in this paper [11] and also a meeting from professional. An evidence from all the song from (most) of the fingerprinted music collection is used to test the program, this is so that it reduce the likelihood of a flawed test. Classical music is used because there is lower chance for creating copyright issues.

The test have increasing level of difficulty:

- Test 1: original song file (unmodified digital file)
- Test 2: original song file but shifted
- Test 3: original song file but with controlled level of noise. (E.g.: a sine wave)
- Test 4: original song file but played over the air and recorded.
- Test 5: original song but perform by different people.
- Test 6: transposed song

The test results is inside [Figure B.1](#). The program overall worked as expected but test 6 doesn't work because this algorithm is not designed for this.

CHAPTER 4

Discussion

As discussed in [the test conducted](#), this artefact is not capable to detect a transposed song. Because this song only matches the exact frequency that the song produces.

One way that the problem can solve is to change the mechanism entirely to not rely purely on the raw frequency data, instead by using the MIDI file of the song which contains a series of message like note. [1] Then from it, create a database of melody. Which can recognise songs from it by recording the song and passing into the algorithm to process for results. This allows a wider error range for recognizing song such as made from "humming". [4, 11]. However, this would make the algorithm costly and slow to run.

In addition, extra research might be needed to improve the speed, making it more commercially viable which is the goal set out by the Shazam[®] team. [10]

The research can also be further extended not limited to searching for songs, such as searching for patten inside picture or video.

CHAPTER 5

Conclusion

Further research might be needed into improving the accuracy, making it able to recognise variation of the same song. [\[11\]](#)

In conclusion, It met the expectation of the project end goal which are tested using a vigorous testing method set out by a professional opinion and an article.

Bibliography

- [1] amandaghassaei. *What Is MIDI?* Instructables. URL: <https://www.instructables.com/What-is-MIDI/> (visited on 01/31/2025).
- [2] *Fast Fourier transform*. In: *Wikipedia*. Page Version ID: 1273240384. Feb. 1, 2025. URL: https://en.wikipedia.org/w/index.php?title=Fast_Fourier_transform&oldid=1273240384 (visited on 02/06/2025).
- [3] FFmpeg developers. *ffmpeg: A complete, cross-platform solution to record, convert and stream audio and video*. Version n7.1. URL: <https://www.ffmpeg.org/>.
- [4] Asif Ghias et al. “Query by humming: musical information retrieval in an audio database”. In: *Proceedings of the third ACM international conference on Multimedia - MULTIMEDIA '95*. the third ACM international conference. San Francisco, California, United States: ACM Press, 1995, pp. 231–236. ISBN: 978-0-89791-751-3. DOI: [10.1145/217279.215273](https://doi.org/10.1145/217279.215273). URL: <http://portal.acm.org/citation.cfm?doid=217279.215273> (visited on 01/31/2025).
- [5] Charles R. Harris et al. “Array programming with NumPy”. In: *Nature* 585.7825 (Sept. 17, 2020), pp. 357–362. ISSN: 0028-0836, 1476-4687. DOI: [10.1038/s41586-020-2649-2](https://doi.org/10.1038/s41586-020-2649-2). URL: <https://www.nature.com/articles/s41586-020-2649-2> (visited on 01/12/2025).
- [6] *Hash function*. In: *Simple English Wikipedia, the free encyclopedia*. Page Version ID: 9718140. Aug. 22, 2024. URL: https://simple.wikipedia.org/w/index.php?title=Hash_function&oldid=9718140 (visited on 01/30/2025).
- [7] Cameron MacLeod. *abracadabra: How does Shazam work?* - Cameron MacLeod. URL: <https://www.cameronmacleod.com/blog/how-does-shazam-work> (visited on 10/28/2024).
- [8] Danielle Newnham. *INTERVIEW: Dr Avery Wang, Principal Research Scientist at Apple and Co-Founder and Chief Scientist...* Medium. Feb. 2, 2023. URL: <https://daniellenewnham.medium.com/fostering-innovation-and-choosing-the-right-founding-team-6287f7eed9f> (visited on 02/02/2025).

- [9] Pauli Virtanen et al. “SciPy 1.0: fundamental algorithms for scientific computing in Python”. In: *Nature Methods* 17.3 (Mar. 2020). Publisher: Nature Publishing Group, pp. 261–272. ISSN: 1548-7105. DOI: [10.1038/s41592-019-0686-2](https://doi.org/10.1038/s41592-019-0686-2). URL: <https://www.nature.com/articles/s41592-019-0686-2> (visited on 10/28/2024).
- [10] Avery Li-Chun Wang and Julius O. Smith III. “Systems and methods for recognizing sound and music signals in high noise and distortion”. U.S. pat. 8386258B2. Shazam Investments Ltd. Feb. 26, 2013. URL: <https://patents.google.com/patent/US8386258B2/en?assignee=Shazam+Investments+Ltd&num=25> (visited on 12/25/2024).
- [11] Cheng Yang. *Music Database Retrieval Based on Spectral Similarity*. Technical Report 2001-14. Backup Publisher: Stanford InfoLab. Stanford, 2001. URL: <http://ilpubs.stanford.edu:8090/489/>.

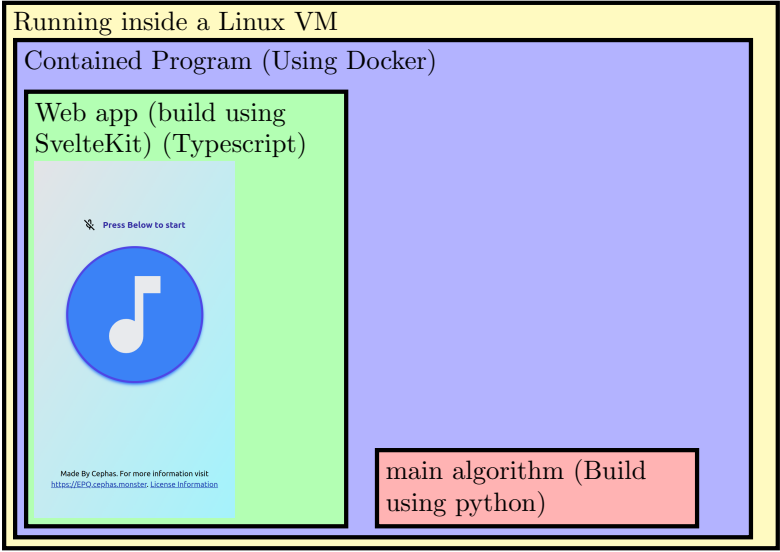


Figure A.1: Overall Architecture

0. User pressed start record button (🔊)

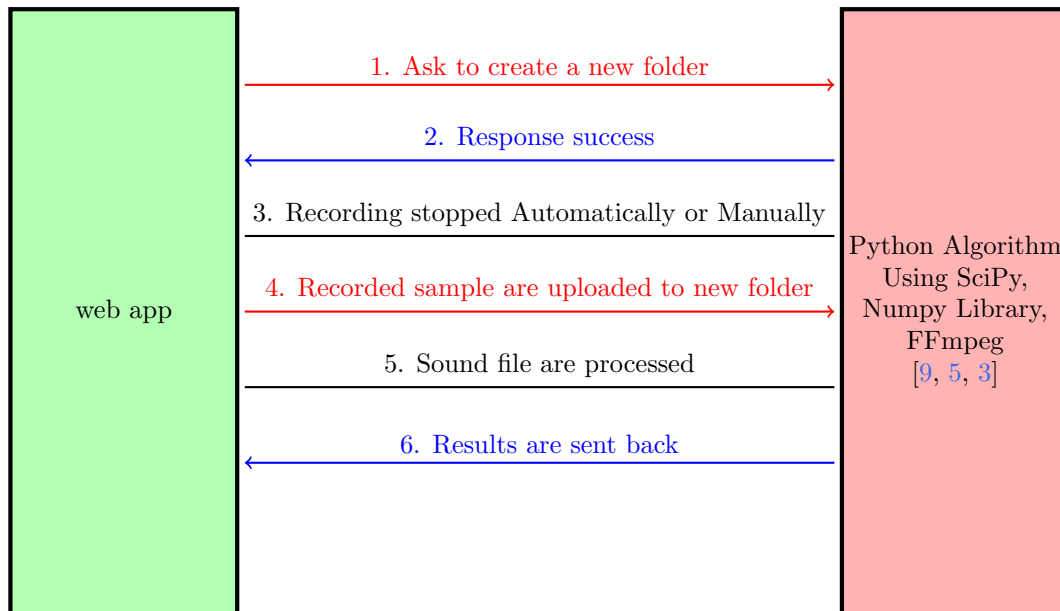


Figure A.2: Overall Architecture (app)

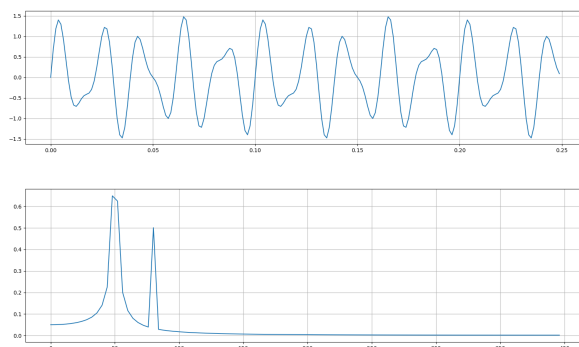


Figure A.3: Fast Fourier Transform as an example through transforming the top signal in to representation in the frequency domain at the bottom.

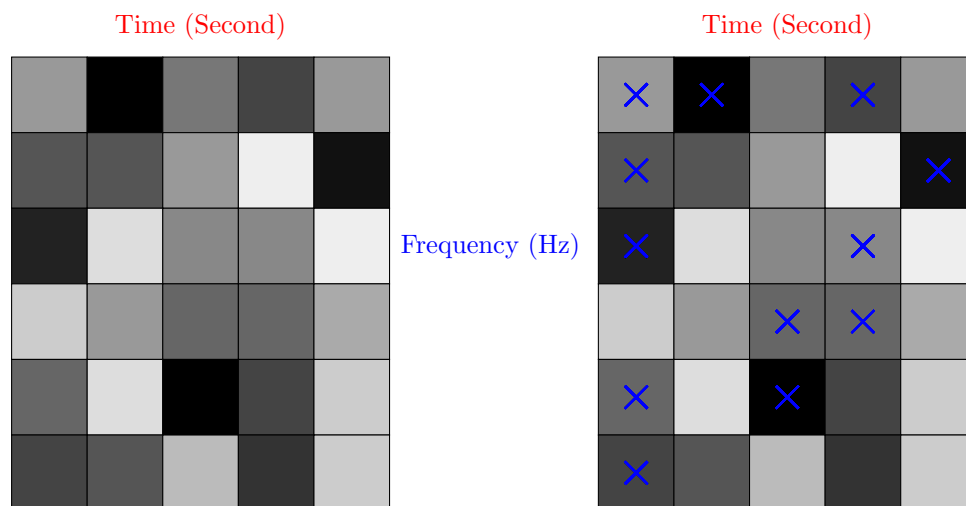


Figure A.4: Illustrating the picking of points, the darker the box, the darker the amplitude (value) represents

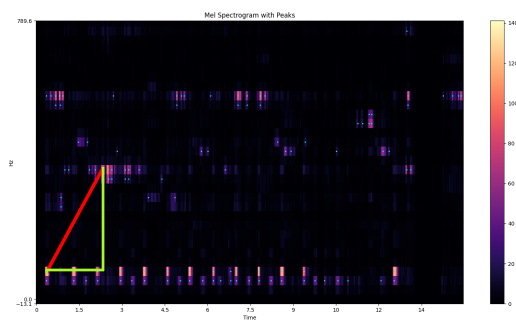


Figure A.5:

Point A (Hz)	Point B (Hz)	Time delta (s)	Point A time (s)	Track UUID
20.1	302.1	1.8	0.3	<i>ea47a3c5</i>
2388532207457024332			2.1	<i>ea47a3c5</i>

Figure A.6: Demonstrate / example of how item are hashed. [7]

APPENDIX B

Testing results

filepath	passed	SUBTOTAL	filepath	passed	SUBTOTAL
test/test_1.py	46	46	test/test_2.py	46	46
TOTAL	46	46	TOTAL	46	46

Figure B.1: test result table

APPENDIX C

Screenshots

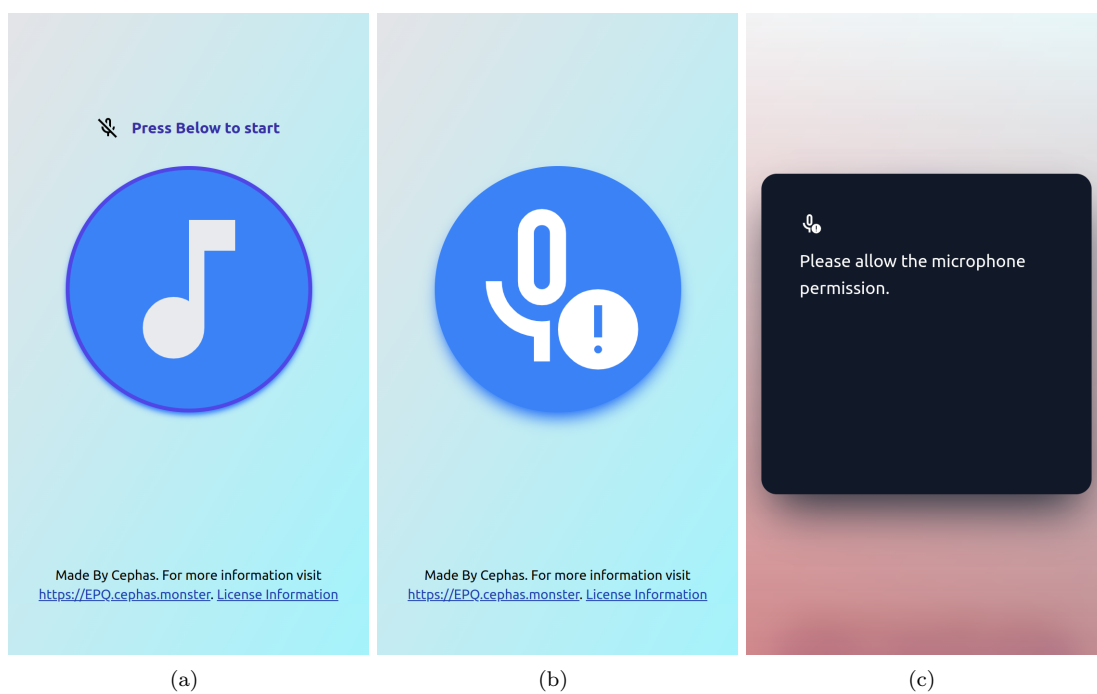


Figure C.1: (a) "Home", (b)(c) "Microphone permission are disabled"

All the figures are simulated to the size of an iPhone[®] SE display.

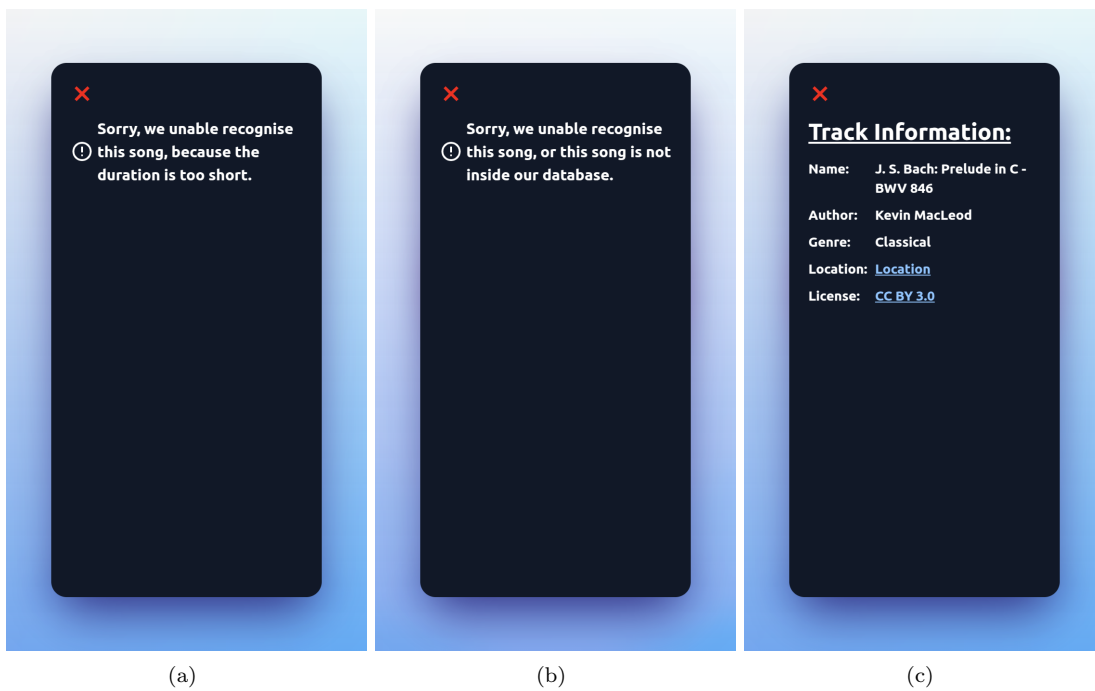


Figure C.2: (a) "Too Short", (b) "No song", (c) "Recognized song (e.g. Prelude by Bach)"

APPENDIX D

Meeting logs

Meeting with the supervisor at 25 November 2024

Summary:

Improve on the part A of the production logs, with more details on the titles described and what resources I will be using (such as the URLs should be provided), the timescale and what is involve.

As things I should be including to create a great project is the benefit that this might bring to the society and the impact.

And a plan to see what I should be sending or showing to the examiner in the future such as:

- Send product log
- Presentation of the artefact
- Evidence of the artefact
- Critical bibliography
- Place that in the Appendix and write a reference back to the appendix through the production log.

Meeting with a professional at 13 January 2025

Summary:

I explained the background details during the meeting.

I have explained that the program can only detect sounds from following test that I conducted such as:

- A test where there are no modifications to the original sound,
- A test where the song has cropped and reduced to a smaller size.
- A test where there is a constant noise generated by a program with a constant sine wave in the background.
- A test where the songs are mixed with a normal acceptable background noise such as coughing.

All of those above works as expected.

However the test that I did by recording when the sound is played on a music player on a different device doesn't work.

As an additional note, he has mentioned that the program is slow because of a reason which might be caused by the hash point not being an integer, as the hash value using integer is faster than using string as hash. (It was diagnosed with a slow database reading speed prior to the meeting).

The project has concluded as partially successful from the results produced from the searching algorithm.

Summary of Improvement I need to make from the project.

I should use a different hashing algorithm for the fingerprinting part of the program.

Round up the frequency, and the changes in time that are fed into the hashing algorithm.

Additional note:

The meetings are conducted with the consent of the professional. (With regards to the ethical issues)

The professional is my computer science teacher, as he has a decent qualification of a university degree.