BECE202L

SIGNALS AND SYSTEMS

A project report titled

MUSICAL NOTES IDENTIFICATION APPLICATION USING SIGNAL PROCESSING

By

23BEC1035	SUJITHRA S
23BEC1197	THERSHNA TK
23BEC1233	T LASYA
23BEC1329	BOJJA DIVYA

BACHELOR OF TECHNOLOGY

IN

ELECTRONICS AND COMMUNICATION ENGINEERING

Submitted to

Dr. S. Sivakumar

November 2024

School of Electronics Engineering

DECLARATION BY THE CANDIDATE

I hereby declare that the Report entitled "**Musical Notes Identification Application using Signal Processing"** submitted by me to VIT Chennai is a record of bonafide work undertaken by me under the supervision of **Dr. S. Sivakumar, Associate Professor, SENSE, VIT Chennai.**

ACKNOWLEDGEMENT

We wish to express our sincere thanks and deep sense of gratitude to our project guide, **Dr. S. Sivakumar,** School of Electronics Engineering for his consistent encouragement and valuable guidance offered to us in a pleasant manner throughout the course of the project work.

We are extremely grateful to **Dr. Ravi Sankar A,** Dean of the School of Electronics Engineering (SENSE), VIT University Chennai, for extending the facilities of the School towards our project and for his unstinting support.

We express our thanks to our **Head of The Department Dr. Mohanaprasad.K** for his support throughout the course of this project.

We also take this opportunity to thank all the faculty of the School for their support and their wisdom imparted to us throughout the courses till date.

We thank our parents, family, and friends for bearing with us throughout the course of our project and for the opportunity they provided us in undergoing this course in such a prestigious institution.

BONAFIDE CERTIFICATE

Certified that this project report entitled "**Musical Notes Identification Application using Signal Processing"** is a bonafide work of Sujithra S (23BEC1035), Thershna TK(23BEC1197),T Lasya (23BEC1233) and Bojja Divya(23BEC1329) carried out the "J"-Project work under my supervision and guidance for BECE202L Signals and Systems.

Dr. S. SIVAKUMAR

School of Electronics Engineering

VIT University, Chennai

Chennai – 600 127.

TABLE OF CONTENTS

ABSTRACT

This project focuses on developing a MATLAB-based musical note recognition software designed to identify musical notes by analysing input frequencies. The software captures real-time or pre-recorded audio, applies noise reduction, and normalizes the signal for improved accuracy. By utilizing the Fast Fourier Transform (FFT), the audio signal is converted into its frequency components, allowing the software to detect the dominant frequency. This dominant frequency is then mapped to the corresponding musical note, enabling reliable note identification. The system specifically determines the musical note based on the dominant frequency, providing a practical tool for music analysis and educational applications.

INTRODUCTION

Musical note identification through signal processing involves analysing sound signals to detect the pitch or frequency of musical notes. This process relies on breaking down complex audio signals into their frequency components and determining which frequency corresponds to a particular musical note. Signal processing techniques offer a powerful way to perform this task, enabling the development of software tools that can automatically recognize musical notes from both live and recorded audio.

Musical notes are defined by their pitch, which is essentially the frequency of the sound wave produced when the note is played. In Western music, each note corresponds to a specific frequency, and these notes are organized in a standard scale. For example, the note A4 is tuned to a frequency of 440 Hz. By analysing the frequency of sound signals, it becomes possible to map these frequencies to their corresponding musical notes.

In real-world scenarios, however, identifying musical notes is not a straightforward task. Audio signals are often complex, containing noise and various harmonic components that can interfere with accurate note detection. Additionally, musical performances can vary in volume, tempo, and sound quality, making it essential to use sophisticated signal processing techniques to ensure reliable identification.

The signal processing methods used in musical note identification typically include:

1. **Frequency Analysis**: Using mathematical transformations, such as the Fast Fourier Transform (FFT), the time-domain audio signal is converted into the frequency domain. This reveals the various frequency components of the signal, making it easier to identify the dominant frequency that corresponds to the musical note.
2. **Noise Reduction**: Real-world audio signals often contain noise that can distort frequency detection. By applying noise reduction techniques, such as digital filtering, extraneous sounds are filtered out, improving the clarity of the signal.
3. **Normalization**: Variations in the amplitude (volume) of the signal can affect the accuracy of frequency detection. Normalizing the signal ensures that the audio input is at a consistent level, which helps improve the reliability of the note identification process.

Musical note recognition software built on these signal processing principles has a wide range of applications, from educational tools that help students learn music theory to practical applications like tuning instruments or analysing musical performances. Through the use of advanced signal processing techniques, these systems can accurately and efficiently identify musical notes, providing a powerful tool for musicians, researchers, and educators alike.

METHODOLOGY

1. Project Overview

- Develop a systematic process for identifying musical notes based on frequency analysis.

2. Recording of Musical Notes

- Capture high-quality audio recordings of musical notes using a microphone or pre-recorded files.

3. Time-Domain Signal Analysis

- Analyse and visualize the recorded audio signal in the time domain to observe waveform characteristics.

4. Application of Windowing Technique

- Implement a windowing technique to isolate individual notes by detecting energy drops between them.

5. Conversion to Frequency Domain

- Use the Fast Fourier Transform (FFT) to convert isolated notes from the time domain to the frequency domain.

6. Fundamental Frequency Extraction

- Identify the fundamental frequency for each note by locating the maximum frequency component in the FFT output.

7. Frequency Matching to Musical Notes

- Compare the detected frequencies against reference pitch values to match them to corresponding musical notes.

8. Performance Evaluation

- Assess the effectiveness of the note identification process by measuring accuracy and reliability.

9. Error Analysis and Mitigation

- Identify sources of errors in the process and propose strategies to reduce them.

10. Integration into Practical Applications

- Explore practical applications of the note identification system in music education and performance feedback.

11. Documentation and Results Analysis

- Document methodologies, MATLAB code, and results for thorough analysis and review.

12. Future Work and Enhancements

- Identify opportunities for advanced research and development in musical note identification techniques.

13. Presentation of Findings

- Prepare a presentation to showcase the methodology and practical implications of the project.

DESIGN CONCEPT

The audio signal is processed using MATLAB's Fast Fourier Transform (FFT) to convert it from the time domain to the frequency domain, allowing the identification of each note's fundamental frequency. The fundamental frequency is matched with a predefined table of musical note frequencies. The system compares the detected frequency to identify the corresponding musical note.

FAST FOURIER TRANSFORM

The Fast Fourier Transform (FFT) is an efficient algorithm used to compute the Discrete Fourier Transform (DFT) and its inverse. It reduces the complexity of calculating the DFT from to , making it much faster for large datasets. In signal processing, FFT converts a time-domain signal into its frequency-domain representation, allowing the identification of the signal's frequency components. This is essential in applications like audio analysis, where the dominant frequencies (such as musical notes) need to be extracted from complex waveforms.

PITCH DETERMINATION

Octave Context:
C3: One octave below Middle C (130.81 Hz)
C4: Middle C
C5: One octave above Middle C (523.25 Hz)
Below Middle C Octave: Multiply frequency by 2 and repeat.
Above Middle C Octave: Divide frequency by 2 and repeat.
Middle C Octave: Identify pitch based on frequency ranges and return octave relative to Middle C.

MATLAB CODE

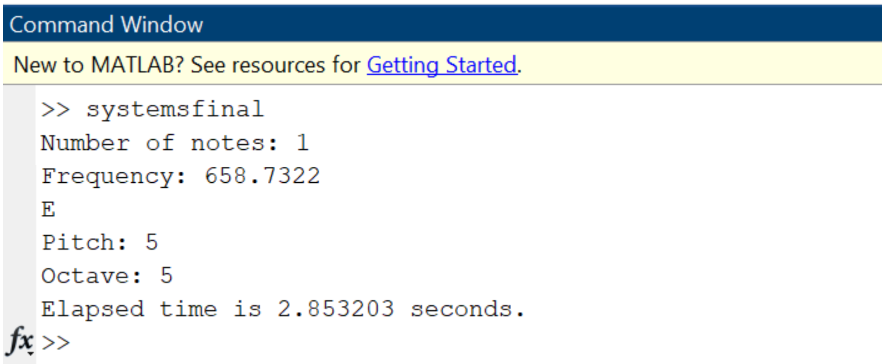
```
% Read audio file
[data, Fs] = audioread('piano_A2.wav');
tic;

% Get note windows
w = notewindows(data);

% How many notes in this sample
num_notes = length(w) - 1;
disp(['Number of notes: ', num2str(num_notes)]);

% Initialize frequency array
freq = zeros(1, num_notes);

% Create a new figure for plotting
figure;

% Plot the original audio signal
subplot(3, 1, 1); % 3 rows, 1 column, 1st subplot
t = (0:length(data)-1) / Fs; % Time vector for the original audio
plot(t, data);
title('Original Audio Signal');
xlabel('Time (seconds)');
ylabel('Amplitude');
grid on;

for i = 1:num_notes
   % Take the window for the current note
   cur_note = data(w(i):w(i+1));
   len = length(cur_note);

   % Compute FFT of the current note
   cur_fft = abs(fft(cur_note));
   cur_fft = cur_fft(1:len/2+1);  % Retain positive frequencies

   % Create frequency axis
   freq_axis = (0:len/2) * Fs / len;

   % Find maximum magnitude and its index
   [Y, I] = max(cur_fft);
   max_frequency = freq_axis(I);

   % Store frequency
```

```
    freq(i) = max_frequency;

    % Display frequency and note info
    disp(['Frequency: ', num2str(max_frequency)]);
    [p, o] = findpitch(max_frequency);
    disp(['Pitch: ', num2str(p)]);
    disp(['Octave: ', num2str(o + 4)]);

    % Plot the FFT for the current note
    subplot(3, 1, 2); % 3 rows, 1 column, 2nd subplot
    hold on; % Hold on to plot multiple FFTs
    plot(freq_axis, cur_fft);
    title('FFT of Notes');
    xlabel('Frequency (Hz)');
    ylabel('Magnitude');
    xlim([0, Fs/2]);  % Set x-axis limits
    grid on;  % Add a grid for better visualization

    % Highlight the peak frequency on the plot
    plot(max_frequency, Y, 'ro', 'MarkerSize', 10); % Highlight the peak
end

% Finalize the FFT plot
legend('FFT Magnitude', 'Peak Frequency');

% Plot frequencies detected for each note
subplot(3, 1, 3); % 3 rows, 1 column, 3rd subplot
plot(1:num_notes, freq, 'o-');
title('Detected Frequencies of Notes');
xlabel('Note Index');
ylabel('Frequency (Hz)');
grid on;

toc;

% Additional Functions (include these below your main code)

function divs = noteparse(data)
    len = length(data);
    threshup = 0.2 * max(data);  % 20% of the maximum value
    threshdown = 0.04 * max(data);

    quiet = 1;  % flag for quiet/noisy state
    j = 1;
```

```
for i = 51:len-50
    if quiet == 1  % looking for beginning of a note
        if (max(abs(data(i-50:i+50))) > threshup)
            quiet = 0;  % found the start
            divs(j) = i;  % record division point
            j = j + 1;
        end
    else
        if (max(abs(data(i-50:i+50))) < threshdown)
            quiet = 1;  % note's over
            divs(j) = i;
            j = j + 1;
        end
    end
  end
end

function w = notewindows(data)
  divs = noteparse(data);
  d2(1) = 0;

  for i = 1:length(divs)
    d2(i + 1) = divs(i);
  end

  d2(length(divs) + 2) = length(data);

  for i = 1:length(d2)/2
    w(i) = (d2(2*i - 1) + d2(2*i)) / 2;
  end
end

function [pitch, octave] = findpitch(freq)
  octave = helpfindoctave(freq, 0);
  pitch = choosepitch(freq / 2^octave);
end

function oct = helpfindoctave(f, o)
  if f >= 254.284 && f <= 508.5675
    oct = o;
  elseif f < 254.284
    oct = helpfindoctave(2*f, o - 1);
  elseif f > 508.5675
    oct = helpfindoctave(f / 2, o + 1);
  end
```

```
end

function p = choosepitch(f)
    if f >= 254.284 && f < 269.4045
        p = 1; disp('C');
    elseif f >= 269.4045 && f < 285.424
        p = 2; disp('C#');
    elseif f >= 285.424 && f < 302.396
        p = 3; disp('D');
    elseif f >= 302.396 && f < 320.3775
        p = 4; disp('D#');
    elseif f >= 320.3775 && f < 339.428
        p = 5; disp('E');
    elseif f >= 339.428 && f < 359.611
        p = 6; disp("F");
    elseif f >= 359.611 && f < 380.9945
        p = 7; disp("F#");
    elseif f >= 380.9945 && f < 403.65
        p = 8; disp("G");
    elseif f >= 403.65 && f < 427.6525
        p = 9; disp("G#");
    elseif f >= 427.6525 && f < 453.082
        p = 10; disp("A");
    elseif f >= 453.082 && f < 480.0235
        p = 11; disp("A#");
    elseif f >= 480.0235 && f < 508.567
        p = 12; disp("B");
    else
        error('Frequency outside of acceptable range');
    end
end
```

RESULTS


Command Window
New to MATLAB? See resources for Getting Started.
>> systemsfinal
Number of notes: 1
Frequency: 658.7322
E
Pitch: 5
Octave: 5
Elapsed time is 2.853203 seconds.
>>

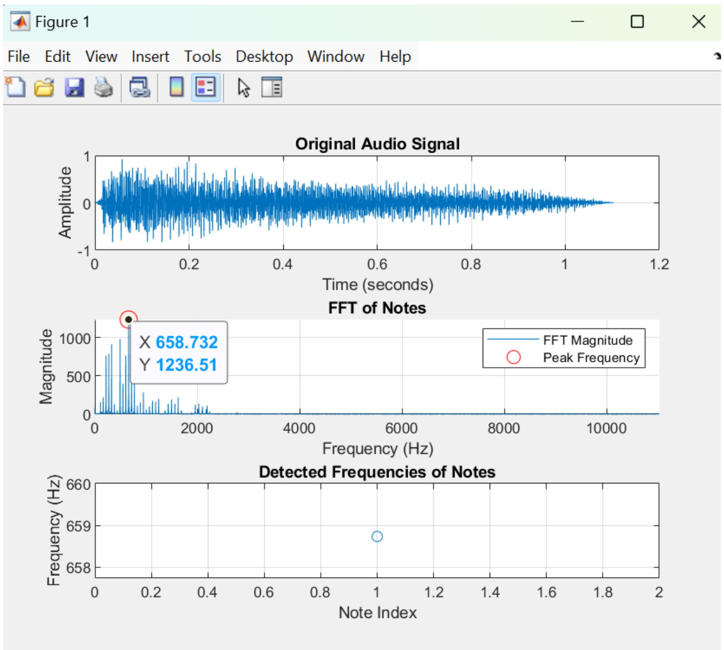


Figure 1
File Edit View Insert Tools Desktop Window Help
Original Audio Signal
Amplitude
Time (seconds)
FFT of Notes
Magnitude
X 658.732
Y 1236.51
FFT Magnitude
Peak Frequency
Frequency (Hz)
Detected Frequencies of Notes
Frequency (Hz)
Note Index

CONCLUSION

In conclusion, our thorough investigation into musical note identification based on dominant frequency has provided valuable insights into the application of signal processing techniques in audio analysis. By leveraging Fourier Analysis and the Fast Fourier Transform (FFT), we developed a system capable of accurately detecting the dominant frequency of audio signals and mapping it to corresponding musical notes. This project has demonstrated the effectiveness of our approach in transforming audio input into meaningful frequency data, which is essential for applications in music analysis and sound processing. The ability to identify musical notes efficiently opens up various possibilities for practical implementations, including educational tools, audio processing applications, and real-time analysis systems. Our work has also highlighted the importance of key parameters, such as noise reduction and signal normalization, in enhancing the accuracy of frequency estimation. Through careful experimentation and analysis, we have identified areas for future exploration, including the potential for real-time processing and the integration of advanced algorithms to improve the system's robustness in diverse acoustic environments. Overall, this project lays a solid foundation for further research and development in the field of musical note identification using signal processing. The insights gained here not only contribute to the understanding of audio signal analysis but also pave the way for innovations that can impact various applications in the realm of sound and music technology.

FUTURE SCOPE

There is still much room for future development that would enhance the system and increase its usage value. The following items are some suggestions:

• **Advanced Note Detection:** Improve note detection by focusing on frequency changes rather than intensity variations. Address challenges in tracking gradual frequency changes.

• Non-Periodic Signal Analysis: Enhance analysis for real-life, time-varying signals. Develop algorithms to improve frequency estimation beyond traditional Fourier Analysis.

• Multiple Notes Detection : Adapt system to handle multiple simultaneous notes. Use advanced Fourier Analysis techniques to isolate and analyse individual notes.

PRESENTATION LINK

https://1drv.ms/p/c/c65a3ab2a75613f1/EdToHLdf29FAjqF-W9MVpoMBkJGTXGeHsa8UH33aOzHKbA?e=eQ9PM1

REFERENCES

- **https://www.google.com**
- **www.wikipedia.com**
- **https://in.mathworks.com/matlabcentral/fileexchange**