**AUDIO SIGNAL PROCESSING USING MATLAB**

**A PROJECT REPORT**

**Submitted**

**by**

**Shruthi Rajagopal (2017505066)**

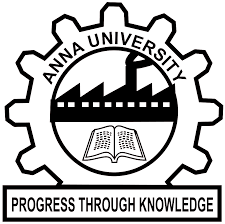
*In partial fulfilment for the award of the degree*

*Of*

**BACHELOR OF ENGINEERING**

*In*

**ELECTRONICS AND INSTRUMENTATION ENGINEERING**



**MADRAS INSTITUTE OF TECHNOLOGY**

**ANNA UNIVERSITY::CHENNAI 600 044**

**ABSTRACT**

**Audio signal processing** is at the heart of recording, enhancing, storing and transmitting audio content. Audio signal processing is used to convert between analog and digital formats, to cut or boost selected frequency ranges, to remove unwanted noise, to add effects and to obtain many other desired results.

The main objective of Audio Signal Processing is the enhancement of audio material or the optimisation of acoustic circumstances for users and receivers. We create a simple sound with noise and we filter this noise using a bandpass filter and produce digital representation of the audio sample.

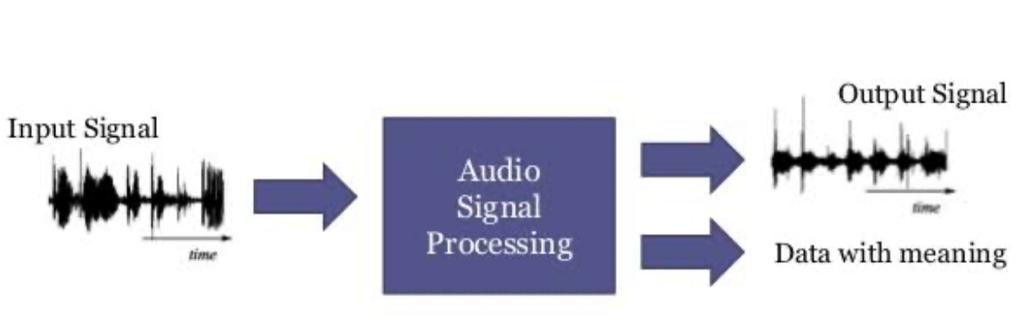
**CONTENTS PAGE**

|  |  |  |
| --- | --- | --- |
| **Chapter** | **Title** | **Page No** |
| 1 | Introduction | 4 |
| 2 | Literature Survey | 5 |
| 3 | Methodology | 7 |
| 4 | Simulation And Results | 9 |
| 5 | Conclusion And Future Work | 11 |
| 6 | References | 11 |

**CHAPTER-1**

**INTRODUCTION**

Audio Signal Processing is an engineering field which focuses on computational methods of intentionally altering auditory signals or sounds, in order to achieve a particular goal. It  is a subfield of [signal processing](https://en.wikipedia.org/wiki/Signal_processing) that is concerned with the electronic manipulation of [audio signals](https://en.wikipedia.org/wiki/Audio_signal).



Audio signals are electronic representations of [sound waves](https://en.wikipedia.org/wiki/Sound_wave)—[longitudinal waves](https://en.wikipedia.org/wiki/Longitudinal_wave) which travel through air, consisting of compressions and rarefactions. The energy contained in audio signals is typically measured in [decibels](https://en.wikipedia.org/wiki/Decibel). As audio signals may be represented in either [digital](https://en.wikipedia.org/wiki/Digital_signal_(signal_processing)) or [analog](https://en.wikipedia.org/wiki/Analog_signal) format, processing may occur in either domain. Analog processors operate directly on the electrical signal, while digital processors operate mathematically on its digital representation.

Processing methods and application areas include [storage](https://en.wikipedia.org/wiki/Audio_storage), [data compression](https://en.wikipedia.org/wiki/Audio_data_compression), [music information retrieval](https://en.wikipedia.org/wiki/Music_information_retrieval), [speech processing](https://en.wikipedia.org/wiki/Speech_processing), [localization](https://en.wikipedia.org/wiki/Acoustic_location), [acoustic detection](https://en.wikipedia.org/wiki/Detection_theory), [transmission](https://en.wikipedia.org/wiki/Transmission_(telecom)), [noise cancellation](https://en.wikipedia.org/wiki/Noise_cancellation), [acoustic fingerprinting](https://en.wikipedia.org/wiki/Acoustic_fingerprint), [sound recognition](https://en.wikipedia.org/wiki/Sound_recognition), [synthesis](https://en.wikipedia.org/wiki/Synthesizer), and enhancement.

**CHAPTER-2**

**LITERATURE SURVEY**

## [Audio-motor integration for robot audition](https://www.sciencedirect.com/science/article/pii/B9780128146019000122)

In the context of robotics, [audio signal processing](https://www.sciencedirect.com/topics/computer-science/audio-signal-processing) in the wild amounts to dealing with sounds recorded by a system that moves and whose actuators produce noise. This creates additional challenges in sound-source localization, signal enhancement and recognition. But the specificity of such platforms also brings interesting opportunities: can information about the robot actuators' states be meaningfully integrated in the audio processing pipeline to improve performance and efficiency? While robot audition have grown to become an established field, methods that explicitly use motor-state information as a complementary modality to audio are scarcer. This chapter proposes a unified view of this endeavour, referred to as audio-motor integration. A literature review and two learning-based methods for audio-motor integration in robot audition are presented, with application to single-microphone sound-source localization and ego-noise reduction on real data.

## [A Deep Learning Framework for Classifying Sounds of Mysticete Whales](https://www.sciencedirect.com/science/article/pii/B9780128113189000223)

This paper addresses a problem belonging to the domain of whale audio processing, more specifically the automatic classification of sounds produced by the Mysticete species. The specific task is quite challenging given the vast repertoire of the involved species, the adverse acoustic conditions, and the nearly non-existent prior scientific work. Two feature sets coming from different domains (frequency and wavelet) were designed to tackle the problem. These are modelled by an Echo State Network classifier.

1. **Audio Processing using Haskell**

The software for most today’s applications including signal processing applications is written in imperative languages. Imperative programs are fast because they are designed close to the architecture of the widespread computers, but they don’t match the structure of signal processing very well. In contrast to that, functional programming and especially lazy evaluation perfectly models many common operations on signals. Haskell is a statically typed, lazy functional programming language which allow for a very elegant and concise programming style. We want to sketch how to process signals, how to improve safety by the use of physical units, and how to compose music using this language.

**4.** [**Real-time, low latency audio processing in Java**](https://www.researchgate.net/profile/Stefan_Mueller_Arisona/publication/267233226_Real-time_low_latency_audio_processing_in_Java/links/54d3bbf80cf2970e4e60158f.pdf)  
This paper discusses the implementation of real-time and low latency audio processing in Java. Despite the fact that Java SE is widespread and has a large programmer base, it is clearly neither targeted at real-time, nor at low-latency applications. As such, doing good audio processing with this language is a challenging task and various issues have to be taken into account: these include limitations or properties of the audio drivers, the kernel and operating system, the audio API, the Java virtual machine and the garbage collector.

**CHAPTER-3**

**METHODOLOGY**

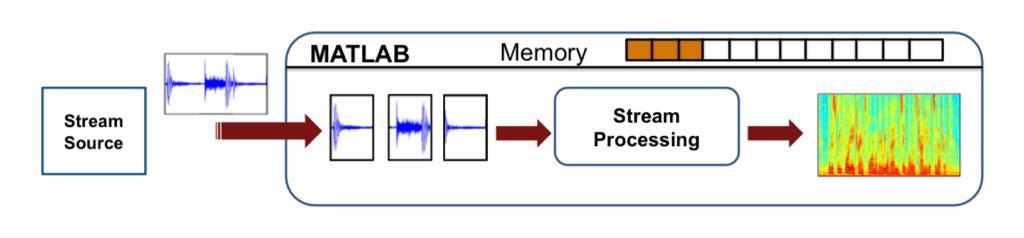
Matlab is widely used environment for signal processing and analysis.

DSP System Toolbox provides algorithms, apps, and scopes for designing, simulating, and analyzing signal processing systems in MATLAB and Simulink.

With DSP System Toolbox you can design and analyze FIR, IIR, multirate, multistage, and adaptive filters. You can stream signals from variables, data files, and network devices for system development and verification.

1. **Create and initialize an Audio Sample**

Create an audio sample and save it in a folder. Then using the Matlab function [[y](https://in.mathworks.com/help/matlab/ref/audioread.html" \l "btiabil-1-y),[Fs](https://in.mathworks.com/help/matlab/ref/audioread.html#btiabil-1-Fs)] = audioread([filename](https://in.mathworks.com/help/matlab/ref/audioread.html#btiabil-1-filename)) reads data from the file named filename as saved in the folder, and returns sampled data, y, and a sample rate for that data, Fs.



**Sound**( **y** ) sends **audio** signal **y** to the speaker at the default sample rate of 8192 hertz. **sound**( **y** , **Fs** ) sends **audio** signal **y** to the speaker at sample rate **Fs** . **sound**( **y** , **Fs** , nBits ) uses nBits bits per sample for **audio** signal **y** .

1. **Processing the Audio signal**

fir1 implements the classical method of windowed linear-phase FIR digital filter design. a=fir1(n,f,'high')is the fir high pass filter.

b=fir1(n,f,'low')is the fir low pass filter used to remove White Gaussian noise or to eliminate high frequency noise from the audio sample.

Then the audio sample is passed to the high pass filter first then passed to low pass filter. fvtool is a 'filter visualisation tool' when we pass an audio signal to it we need to cascade the two FIR filters then supply this as the argument to see its response. fvtool(p,1) opens FVTool and displays the magnitude response of the digital filter defined with numerator p and denominator 1.

Then we plot the original signal and filtered signal. y is the digital plot of the original signal. p is the plot of the filtered signal. When we run the program we get the desired digital representation of the filtered signal as well as the plot of the original signal with noise.

**CHAPTER-4**

**SIMULATION AND RESULTS**

**Code to initialize the audio sample**

clc;

close all;

clear all;

[y,fs] = audioread('Audio signal.ogg')

Sound(y,fs)

**Output**

*\*Audio plays\**

**Code for processing the audio and plotting its digital output**

clc;

close all;

clear all;

f=0.8;

n=6;

a=fir1(n,f,'high'); %fir high pass filter

b=fir1(n,f,'low'); %fir low pass filter

[y,fs] = audioread('Audio signal.ogg');%load audio file

o=filter(a,1,y);

p=filter(b,1,o);

fvtool(p,1);

subplot(2,1,1);

plot(y);

subplot(2,1,2);

plot(p);

**PLOT:**

Filter Response:

******

******

**CHAPTER-5**

**CONCLUSION AND FUTURE WORK**

Audio signal processing is used to convert between analog and digital formats, to cut or boost selected frequency ranges, to remove unwanted noise, to add effects and to obtain many other desired results.

So in this, we create a simple sound with noise and we filter this noise using a bandpass filter. This is a subfield of signal processing that is concerned with the electronic manipulation of audio signals. Processing methods and application areas include [storage](https://en.wikipedia.org/wiki/Audio_storage), [data compression](https://en.wikipedia.org/wiki/Audio_data_compression), [music information retrieval](https://en.wikipedia.org/wiki/Music_information_retrieval), [speech processing](https://en.wikipedia.org/wiki/Speech_processing), [localization](https://en.wikipedia.org/wiki/Acoustic_location), [acoustic detection](https://en.wikipedia.org/wiki/Detection_theory), [transmission](https://en.wikipedia.org/wiki/Transmission_(telecom)), [noise cancellation](https://en.wikipedia.org/wiki/Noise_cancellation), [acoustic fingerprinting](https://en.wikipedia.org/wiki/Acoustic_fingerprint), [sound recognition](https://en.wikipedia.org/wiki/Sound_recognition), [synthesis](https://en.wikipedia.org/wiki/Synthesizer), and enhancement.

**CHAPTER-6**

**REFERENCES**

1. <https://en.wikipedia.org/wiki/Audio_signal_processing>
2. <https://www.mathworks.com/content/dam/mathworks/mathworks-dot-com/campaigns/portals/files/apple/AudioSignalProcessing.pdf>
3. <https://www.rit.edu/press/introduction-audio-signal-processing>
4. <https://circuits-diy.com/audio-signal-processing-in-matlab/>