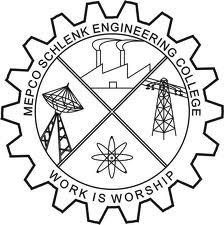
****

**AUDIO**

**ENHANCER**

## DEPARTMENT OF ELECTRONICS AND

## COMMUNICATION

## ENGINEERING

**MEPCO SCHLENK ENGINEERING COLLEGE, SIVAKASI**

## TABLE OF CONTENTS

|  |  |  |
| --- | --- | --- |
| **CHAPTER**  **NO** | **TITLE** | **PAGE**  **NO** |
| 1 | INTRODUCTION | 1 |
| 2 | MATLAB PROGRAMMING TOOL | 2 |
| 3 | TYPES OF NOISE | 3 |
| 4 | TYPES OF FILTERS | 4 |
| 5 | MATLAB PROGRAM | 5 |
|  | AUDIO MIXING |  |
|  | FREQUENCY SPECTRUM |  |
|  | PLOT AND RESPONSE |  |
|  | AUDIO AMPLIFIER |  |
|  | NOISE |  |
|  | LOW FREQUENCY |  |
|  | HIGH FREQUENCY |  |
|  | GAUSSIAN |  |
|  | FILTER |  |
|  | HIGH PASS |  |
|  | LOW PASS |  |
|  | WIENER |  |
| 6 | OUTPUT RESPONSE | 12 |
| 7 | CONCLUSION | 13 |

**INTRODUCTION**

**Audio enhancement** is the scientific analysis and improvementof audioclarity, typically to improve intelligibility.

**Noise reduction** is the process of removing [noise](https://en.wikipedia.org/wiki/Noise_(signal_processing)) from a [signal](https://en.wikipedia.org/wiki/Signal_(information_theory)). Noise reduction techniques exist for audio and images.

Noise reduction algorithms may distort the signal to some degree.

Then using analog tape-recording technology, they may exhibit a type of noise known as tape hiss.

This is related to the particle size and texture used in the magnetic emulsion that is sprayed on the recording media, and also to the relative tape velocity across the [tape heads](https://en.wikipedia.org/wiki/Tape_head).

**MATLAB PROGRAMMING TOOL**

* **MATLAB** (an abbreviation of "MATrix LABoratory") is a proprietary multi-paradigm programming language and numeric computing environment developed by MathWorks. MATLAB allows matrix manipulations, plotting of functions and data, implementation of algorithms, creation of user interfaces, and interfacing with programs written in other languages.
* The MATLAB application is built around the MATLAB programming language.
* Common usage of the MATLAB application involves using the "Command Window" as an interactive mathematical shell or executing text files containing MATLAB code.
* The software was popularized largely thanks to toolboxes created by experts in various fields for performing specialized mathematical tasks.
* MATLAB users come from various backgrounds of engineering, science, and economics.
* As of 2020, MATLAB has more than 4 million users worldwide.

**TYPES OF NOISE**

* **Additive noise** gets added to the intended signal.
* **White noise** is a random signal having equal intensity at different frequencies, giving it a constant power spectral density.
* **Black noise** is anoise with a spectrum corresponding to the [blackbody](https://en.wikipedia.org/wiki/Blackbody) radiation (thermal noise).
* **Gaussian noise** has a probability density function (PDF) equal to that of the normal distribution, which is also known as the Gaussian distribution.
* **Pink noise** or **flicker noise**, is a signal or process with a frequency spectrum such that the power spectral density (power per frequency interval) is inversely proportional to the frequency of the signal.
* **Multiplicative noise** refers to an unwanted random signal that gets multiplied into some relevant signal during capture, transmission, or other processing.
* **Quantization error** is the difference between an input value and its quantized value (such as round-off error.
* **Shot noise** or **Poisson noise** is a type of noise which can be modeled by a Poisson process.
* **Transient noise** pulses consist of a relatively short pulse followed by decaying low frequency oscillations.
* **Burst noise**, powerful but only during short intervals.
* **Phase noise**, random time shifts in a signal.

**TYPES OF FILTERS**

* **High-Pass** and **Low-Pass filters** are the simplest forms of digital filters, and they are relatively easy to design to specifications.
* **Band-pass Filters** are like a combination of a high-pass and a low-pass filter. Only specific bands are allowed to pass through the filter.
* **Band-stop Filters** are like a combination of a high-pass and a low-pass filter. Specific bands are not allowed to pass through the filter.
* **Wiener filter** is a filter used to produce an estimate of a desired or target random process by linear time-invariant (LTI) filtering of an observed noisy process, assuming known stationary signal and noise spectra, and additive noise. The Wiener filter minimizes the mean square error between the estimated random process and the desired process **.**
* **Notch filters** are the complement of Band-pass filters in that they only stop a certain narrow band of frequency information, and allow all other data to pass without problem.
* **Comb Filters** as their name implies, look like a hair comb. They have many "teeth", which in essence are notches in the transfer function where information is removed. These notches are spaced evenly across the spectrum, so they are only useful for removing noise that appear at regular frequency intervals.
* **All-pass filter** is a filter that has a unity magnitude response for all frequencies. It does not affect the frequency response of a filter. The All-pass filter does affect the phase response of the system.

For example, if an IIR filter is designed to meet a prescribed magnitude response often the phase response of the system will become non-linear. To correct for this non-linearity an All-pass filter is cascaded with the IIR filter so that the overall response has a constant group delay.

**MATLAB PROGRAM**

**Audio Mixing**

close all

clc

[x,fs]=audioread('Clean.mp3');

[z,fs]=audioread('Noise.wav');

y=x+z;

audiowrite('Test.wav',y,fs)

sound(y,fs)

**Frequency Spectrum FFT**

clear all

close all

clc

[z,fs]=audioread('Noise.wav');

fftx=fft(z,1024);

fft\_rearr=[fftx(513:1024) fftx(1:512)];

x\_axis=linspace(-fs/2,fs/2,512);

plot(x\_axis,abs(fft\_rearr))

**Frequencu Spectrum**

close all

clc

[z,fs]=audioread('Noise.wav');

y=fft(z,n);

fftx=(-n/2:n/2-1).\*(fs/n);

ffty=abs(fftshift(y));

plot(fftx,ffty,'LineWidth',1.5)

**Plot And Response**

subplot(3,2,1)

plotWave\_YW(0,x,fs,'time',1);

title('Clean speech')

subplot(3,2,2)

plotWave\_YW(0,x,fs,'freq');

subplot(3,2,3)

plotWave\_YW(0,z,fs,'time',1);

title('Noisy speech')

subplot(3,2,4)

plotWave\_YW(0,z,fs,'freq');

subplot(3,2,5)

plotWave\_YW(0,y,fs,'time',1);

title('Enhanced speech')

subplot(3,2,6)

plotWave\_YW(0,y,fs,'freq');

**Noise**

**Noise Generation Low Freq**

clear all

close all

clc

fl=600;

[x,fs]=audioread('Clean.mp3');

n=length(x);

t=(1:n)/fs;

z=cos(2\*pi\*fl\*t);

sound(z,fs)

audiowrite('Noise.wav',z,fs);

**Noise Generation high Freq**

clear all

close all

clc

fh=15000;

[x,fs]=audioread('Clean.mp3');

n=length(x);

t=(1:n)/fs;

y=cos(2\*pi\*fh\*t);

sound(y,fs)

audiowrite('Noise.wav',y,fs);

**Gaussian Noise**

clear all

close all

clc

fh=10000;

[x,fs]=audioread('Clean.mp3');

xn=awgn(x,15,'measured');

sound(xn,fs)

audiowrite('Noise.wav',xn,fs);

**Noise Generation Random**

close all

clc

y=y+2.5\*randn(size(t));

audiowrite('Noise.wav',y,fs);

**Filter Design High Pass**

close all

clc

[z,fs]=audioread('Test.wav');

y = highpass(z,fl+150,fs)

[x,fs]=audioread('Clean.mp3');

sound(x,fs)

sound(y,fs)

audiowrite('Output.wav',y,fs);

**Filter Design Low Pass**

close all

clc

[z,fs]=audioread('Test.wav');

y = highpass(z,fh-200,fs)

[x,fs]=audioread('Clean.mp3');

sound(x,fs)

sound(y,fs)

audiowrite('Output.wav',xn,fs);

**Wiener Filter**

close all

clc

[x, fs] = audioread('Clean.wav');

[z] = audioread('Noise.wav');

y = noiseReduction\_YW(z, fs);

audiowrite('Noise.wav',y,fs);

**Audio Amplifier**

[x, fs] = audioread('Clean.wav');

[y,Fs] = audioread('Output.wav');

[z,fs]=audioread('Test.wav');

y=y\*10

sound(x,Fs)

sound(y,Fs)

sound(y,Fs)

audiowrite('Noise.wav',y,fs);

**Actual Input**:

**Diagram

Description automatically generated**

**Input Signal with Noise**:

**Diagram

Description automatically generated**

**Diagram

Description automatically generatedOutput Signal After Processing**:**CONCLUSION**

* From our observations, we could say that the most difficult process in the audio processing system is the post processing process which is the audio enhancement.
* As per the results obtained from our audio tests, we could say that removing noise form a signal is a difficult process which couldn’t be done easily with general functions.
* It requires more advanced techniques like machine learning and artificial intelligence which could improve the outcome of the project.
* Since basic functions are used for the filtering and enhancing process the obtained results are sufficient to understand the basics of the enhancement process.
* Further improvement could be done by analyzing the signal using other advanced operations other than normal filters. Thus, this project gives better understanding of the basics in the enhancement of the audio.