Group members:maisam nasser ,Baraa shaikh ahmad 1201714

Submitted to: Dr. AbualseoudHanani

Department of Electrical and Computer Engineering

Birzeit University

# Introduction

This project aims to design, implement, and test an English alphabet character voice-frequency encoder. Each English character is represented by a combination of three voice-band frequency components (low, middle, and high). Additionally, the project includes the implementation and testing of a decoder that can recover the text string from the encoded multi-frequency signals. Two decoding approaches are employed: one utilizing frequency analysis and the other using bandpass filters.

It is noteworthy that the programming language used for this project is MATLAB.

# Problem Specification

The primary technical challenge is to design a reliable system for encoding and decoding audio frequencies. The input will consist of text provided by the user, which will be encoded, converted into sound, and then decoded from the audio file. The output will be in the form of the originally encoded text.

# Data

Data for encoding

Text data is requested from the user.

For each character: 3 frequencies (100HZ-3500 Hz) , duration = 0.04; % 40 ms , fs = 8000 and samples : 0 to 320.

Data for decodingfrequency analysis

Inputs to system are (.wav) files.

For each character: 3 frequencies (100HZ-3500 Hz) , duration = 0.04; % 40 ms , fs = 8000 and samples : 0 to 320.

Data for decoding

Inputs to system are (.wav) files

For each character: 3 frequencies (100HZ-3500 Hz) , duration = 0.04; % 40 ms , fs = 8000 and samples : 0 to 320.

size of frame = 320. narrow bandpass filter (iir)

# Evaluation Criteria

To measure the performance of our project, accuracy in the encoding and decoding processes will be a crucial metric. It determines how reliably our system translates between audio frequency signals and English letters.

The accuracy can be calculated using the following formula:

Accuracy=(Total Number of Letters/Number of Correct Letters)×100 %

Accuracy = 25/27 \* 100% = 92.59

# Approach

The project was structured into three main stages. The first stage focused on encoding each character based on the values of three frequencies, utilizing the cosine function. The resulting signal was transformed into a spectral representation using the Fast Fourier Transform (FFT). The magnitude of the FFT output was then stored in a matrix using the **abs** function. The encoded signal was further processed using the **cell2mat** function to convert it to an audio format, which was played using the **audioplayer** package. Additionally, the encoding result was saved as a .wav file using the **audiowrite** function.

In the second stage, the decoding process using frequency analysis was initiated by loading an audio file. The Fourier transform was applied to the signal, and the magnitude of the FFT output was obtained. The **findpeaks** function assisted in identifying the three highest peaks in the frequency domain. Sorting these frequencies facilitated the subsequent selection of the character associated with the three closest frequencies.

The final stage involved decoding using bandpass filters. Filters were applied to improve the signal, and Create narrow bandpass filter (iir)for each frequency (low,mid,high) the numel function helped determine the number of filter elements. The application of filters to the signal was critical, and a challenge arose due to character repetition during decoding. This issue was successfully addressed by using the original frame size of 320.

MATLAB Packages Utilized:

audioplayer: Used for playing the encoded audio.

uiputfile: Enabled user interaction to select the name and path for saving the encoded audio as a .wav file.

fft:Applied for Fourier Transform to obtain frequency components.

abs: Utilized to calculate the magnitude of FFT output.

cell2mat: Converted the encoded signal to an audio format.

audiowrite: Employed for writing the audio signal to a .wav file.

uigetfile: Allowed the user to choose an audio file for decoding.

findpeaks: Assisted in identifying peaks in the frequency domain.

sort: Used for sorting frequencies.

Bandpass filters: Enhanced signal quality during decoding.

# Results and Analysis

I encoded 10 strings and then decoded them twice, once using frequency analysis and once using band-pass filters. I noticed that there is a problem with the space character and the y character; Sometimes they are specified correctly, while other times spe is replaced by "y" and "y" by a space. The rest of the letters were deciphered correctly.

# Conclusions

In the end, we learned to design, implement and test the English alphabet audio frequency encoder which represents each English letter by combining three audio band frequency components (low, medium and high) and implement and test the system decoder in the first part, which can recover the text string from the encrypted signal. . Multi-frequency using two methods: using frequency analysis and using band-pass filters, all of which was done using MatLab software. We learned about many ready-made functions that helped us complete the project and made the project easier for us.

Input signal baraa mohanmad saved in file \*.wav

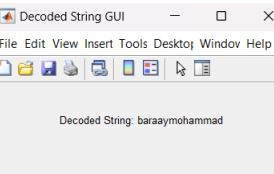
Results using frequency analysis

figure 1

# 

Results using bandpass Filters

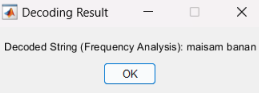
figure 2



Input signal maisam banan saved in file \*.wav

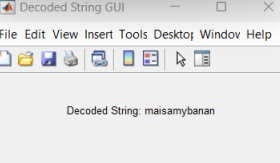
Results using frequency analysis

figure 3



Results using bandpass Filters

figure 4



# Development

We can focus on further refining the interfaces and explore the possibility of adjusting the frequencies assigned to 'y' and space characters. By modifying these frequencies intelligently, we aim to eliminate any conflicts and facilitate accurate identification by the program. This adjustment should enhance the system's ability to determine the values for 'y' and space reliably and present them correctly.

# References

1. <https://www.mathworks.com/help/matlab/ref/uiputfile.html.>(

)Uiputfile(.

1. <https://www.mathworks.com/help/matlab/ref/guidata.html.>

([guidata(obj,data)](https://www.mathworks.com/help/matlab/ref/guidata.html" \l "d126e642932).

1. <https://www.mathworks.com/help/matlab/ref/audioplayer.html> ( uiputfile(.
2. <https://www.mathworks.com/help/matlab/ref/fft.html> (FFT)
3. <https://www.mathworks.com/help/matlab/ref/max.html> (MAX)
4. <https://www.mathworks.com/help/matlab/ref/sum.html> (SUM)
5. <https://www.mathworks.com/help/matlab/ref/abs.html> (ABS)
6. <https://www.mathworks.com/help/matlab/ref/cell2mat.html> (cell2mat)
7. <https://www.mathworks.com/help/matlab/ref/audiowrite.html> (audiowrite)
8. <https://www.mathworks.com/help/matlab/ref/uigetfile.html> (uigetfile)
9. <https://www.mathworks.com/help/signal/ref/findpeaks.htm> l (Findpeaks)
10. <https://www.mathworks.com/help/matlab/ref/sort.html> (Sort)
11. <https://www.mathworks.com/help/matlab/data_analysis/filtering-data.html> (Filters)