BIRZEIT UNIVERSITY

Faculty of Engineering and Technology

Electrical and Computer Engineering

DSP Fall 2021

Course project

Project submission deadline: Sat. 8/1/2021 23:55 on ITC
Project discussion: To be defind later

About the project:
This project must be done by teams of three students. The best arrangement is to choose a division of the project so that each of you can work on separate but interlocking parts. Teams of two or individual work will not be accepted.

Learning teamwork is also one of the more general goals of this course, so team projects will pick up points for demonstrating a successful ability to work with others.

The projects will be graded based on a project report (of around 3-4 pages) as well as in-class short presentations or discussion in my office.

Project submission must be via Moodle only, but please use PDF format and **not** Word .DOC files if at all possible, since I often have formatting problems with Word files.

Your report must have the following structure, using these section headings and using IEEE paper format [template can be found on Moodle]:

Introduction: A general description of the area of your project and why you're doing it.

Problem Specification: A clear technical description of the problem you're addressing. Formulating a general problem (e.g., transcribing music) into a well-defined technical goal (e.g., reporting a list of estimated fundamental periods at each time frame) is often the most important part of a project.

Data: What are the real-world and/or synthetic signals you are going to use to develop and evaluate your work?

Evaluation Criteria: How are you going to measure how well your project performs? The best criteria are objective, quantitative, and discriminatory. You want to be able to demonstrate and measure improvements in your system.

Approach: A description of how you went about trying to solve the problem. Sometimes you can make a nice project by contrasting two or more different approaches.

Results and Analysis: What happened when you evaluated your system using the data and criteria introduced above? What were the principal shortfalls? (This may require you to choose or synthesize data that will reveal these shortcomings.) Your analysis of what happened is one of the most important opportunities to display your command of signal processing concepts.

Development: If possible, you will come up with ideas about how to improve the shortcomings identified in the previous section, and then implement and evaluate them. Did they, in fact, help? Were there unexpected side-effects?

Conclusions: What did you learn from doing the project? What did you demonstrate about how to solve your problem?

References: Complete list of sources you used in completing your project, with explanations of what you got from each.

The reason for this somewhat arbitrary structure is simply to help you avoid some of the more problematic weaknesses I've seen in past years. If you're having trouble fitting your work into these sections, you should probably think more carefully about your project.

Project description:

Part one: String Encoder:

Design, Implement and test an English alphabet character voice-frequency encoder which represents every English character by a combination of three voice-band frequency components in the band 100-4000 Hz(low, middle and high). The case (upper /lower) of the letter is also represented by a forth frequency component which is 100 HZ for lowercase (i.e. small letters) and 200 HZ for the uppercase (capital letters).
For example, to encode small letter 'a', we use a signal that contains frequencies (100Hz, 400HZ, 800HZ, 1600Hz). Therefore, given the frequency combination for each character, you should be able to encode any character to the corresponding signal contacting the corresponding frequencies (see table below). To be consistent with each other, use the sampling frequency Fs = 8KHz.

Table 1.1 show you the frequencies (200HZ-4000Hz) for each character. You may assume duration of each character signal is no more than 40ms.

Table 1.1: Encoding frequencies for each English character

Arabic Character	Capital/Small	Low frequency component	Middle frequency component	High frequency component
A/a	200/100	400	800	1600
B/b	200/100	400	800	2400
C/c	200/100	400	800	4000
D/d	200/100	400	1200	1600
E/e	200/100	400	1200	2400
F/f	200/100	400	1200	4000
G/g	200/100	400	2000	1600
H/h	200/100	400	2000	2400
I/i	200/100	400	2000	4000
J/j	200/100	600	800	1600
K/k	200/100	600	800	2400
L/l	200/100	600	800	4000
M/m	200/100	600	1200	1600
N/n	200/100	600	1200	2400
O/o	200/100	600	1200	4000
P/p	200/100	600	2000	1600
Q/q	200/100	600	2000	2400
R/r	200/100	600	2000	4000
S/s	200/100	1000	800	1600
T/t	200/100	1000	800	2400
U/u	200/100	1000	800	4000
V/v	200/100	1000	1200	1600
W/w	200/100	1000	1200	2400
X/x	200/100	1000	1200	4000
Y/y	200/100	1000	2000	1600
Z/z	200/100	1000	2000	2400
space	X (don't care)	1000	2000	4000

To sum up, in this part, you have to design and implement the following:

1. Ask the user to enter a string (words separated by space).
2. Generate 40ms waveform segment for each character in the string that consists of the corresponding four frequencies, as described in the table above.
3. Concatenate all segments together to generate a waveform representing the whole string.
4. Play the waveform and store it in a .wav audio file in the working directory.

Part Two: String Decoder

In this part, you have to design, implement and test a decoder for the system in part one, which can recover the text string from the encoded multi-frequency signal stored in .wav files. Simply, your system should take an audio file (.wav) as an input and recognizes the encoded characters and display the string on the screen.
You must use the following two approaches to build your decoder:

- Use **frequency** analysis (e.g. Fourier transform) of the input signal to determine which frequencies that have the highest amplitudes in each **40ms** segment and decode the character and its case from them.

- Use bank of very narrow bandpass **Filters,** so that you design filters represent each given frequencies (9 frequency components for the letter and two frequencies for the case) and pass each 40ms segment of the input signal through them to pick the filters that gives the highest output, then determine the frequencies in each **40ms**.

To sum up, in this part you have to design the following:

1. Ask the user to enter the .wav file which contains the encoded string. Read the waveform and divide it into 40ms short segments.
2. Use the two methods described above to decode each segment into the corresponding character and then print the decoded string on the screen.

3. **Sufficiently test your two methods** with different number of encoded strings with different length (number of characters with small/capital letters), and report the accuracy of your systems. Accuracy is the number of correctly recognized letters divided by the string length.

Competition:

For the second part (decoder), we will provide you with a set of encoded strings in the same way described in part one. Every group has to use their two methods to recognize the encoded string and send them back to us. This is a major part for the project evaluation.
To encourage you, the first five teams who recognized all provided strings correctly will have a ***bonus***!

Project deliverables by each group:

1- Mini-report as described above in IEEE template.

2- System demonstration of each part as described above.

You can use any programming language you prefer for implementing your project. However, I highly recommend MATLAB, Octave, or Python because they have many useful functions.

Abualsoud Hanani & Alhareth Zyoud
Dec. 18th 2021