BIRZEIT UNIVERSITY

Ibrahem Duhadi , Eman Maraita, Zakiya AbuMurra

Students in Birzeit University

1190283@Student.birzeit.edu, 1190768@Student.birzeit.edu, 1191636@Student.birzeit.edu

Abstract

The main idea in this project, is to design and implement two basic communication elements, encoder and decoder. To be able to transmit strings over the network, the encoder is needed to represent each character of the string in a specific way to recognize it. At the same time, the decoder is needed in the receiver side to extract the original string from the encoded one. The method used in encoding is to represent each character in English alphabet characters as a signal containing four different frequencies. For decoding operation, the operation is done using two different methods, the first one is using the frequency analysis and the second one is using a narrow band pass filter with distinct frequency values.

1. Introduction

The issue to be deal with in this project is the encoding and decoding operations for the strings.

Encoding and decoding are used in many forms of communications, including computing, data communications, programming, digital electronics and human communications. These two processes involve changing the format of content for optimal transmission or storage.

In computers, encoding is the process of putting a sequence of characters (letters, numbers, punctuation, and certain symbols) into a specialized format for efficient transmission or storage. Decoding is the opposite process, the conversion of an encoded format back into the original sequence of characters [1].

The encoder to be design and implement is the one, which deal with English alphabet characters, the transmitter has to enter a string or may be a statement (words separated by spaces) and then the encoder should be able to obtain each character of the string individually to encode it.

The user input string, the encoder take each character of the string and encode it as follow:

Each English character is represented by three band of frequencies (low, middle, high) frequencies from the range of 100-4000 Hz, but each character can be small or capital letter so that a fourth frequency should be included to determine if the character is small or capital. So that the two frequencies (100/200) Hz are used to show that when using the frequency 100 Hz with the three frequencies, then the character is small, when using the frequency 200 with the other three frequencies, this means that the character is capital one.

Each character is represented by a signal containing 4 different frequencies with specific time 40 ms for each character, then all encoded characters are concatenated as a signal to form the whole string, then the signal is stored in a .wav file to transmit it, according to the following frequency values table:

Table1. 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

According to above table, any English character can be encoded to a signal of four different frequencies to transmit it over the network.

In the second side (receiver side) which should be able to read and deal with the encoded characters (string), so that the decoder is needed to recover and reconstruct the string from the encoded multi-frequency signal stored in a .wav file. The decoder should recognize the encoded character correctly, and display the correct string on the screen.

There are two methods have to follow in decoding operation. The first one is using the frequency analysis (Fourier transform), which deals with impulses, and according to the value of w determine the corresponding character frequencies, this is done by taking the magnitude of the signal in frequency domain, and then take the maximum four values (peaks) to convert them to frequency values.

The second method of decoding is using narrow band pass filters, this method start with designing a number of band pass filters each one of them has a different and unique center frequency within the range of frequencies defined in the table1.1 above. After designing the filters, the each segment of encoded signal should pass through them to pick the filters that gives the highest output, then determine the frequencies in each segment of the signal to form the character.

2. Problem specification

The problem to be solved is sending a text message over the network, for this aim, the encoder and decoder described above was designed.

Each English alphabet character encoded to a signal containing 4 different frequencies, the implementation of the encoder may encounter many problems, such as when convert each character

to a signal with frequencies and send it over the network, an interference with another signal on the same network may appear, and this affect the string to be sent, it may be change or remove or reach the receiver with changed order.
On the other hand, a noise may occur, also affect on the transmitted string.
The same problems can appear at the receiver side, for the reasons of noise and interference the signal that received can be different form the transmitted one.

The practical problems encountered while writing the python code are how to find the suitable functions for coding, how to use them and how to determine which one of them is the best to obtain the highest performance. And another problem was to understand that the band pass filter can't pass a high frequencies, on other words, when the frequency is greater than 4000 then a high pass filter is needed.

3. Data

The data to be encoded in this project is a string (text message), the user has to enter a string or a statement (string containing spaces), this string has to encode in a waveform (signal), the waveform then stored in a .wav file to be sent over the network. In the corresponding side (receiver side), the .wav file is read, and the decoder is needed to be able to understand the string stored in the encoded waveform.
While the signal is travelling over the network, it might get some errors or noise, which has an effect on the accuracy of the transmitted signal, the frequencies may be change or interference with other signals on the network.
According to these reasons, the received signal may not match or lightly different from the original signal which encoded.

4. Evaluation criteria

The performance and accuracy of the encoder and decoder code was tested by comparing the encoded string with the decoded one, and by counting the number of correct decoded characters comparing with the encoded ones, to find the value of the performance of the system. This is evaluation way that used in this project to discover if it works correctly or not.
The criteria used in this project, is the simplest one, which is represented by trying to encode a string or statement containing small, capital letters and spaces.

So that the evaluation criteria used is to input different string, and then try to encode it in the transmitter, and decode it in the receiver, then calculate the accuracy value.

The system was tested with different strings, and at each time it gave very precise answers, and the performance and accuracy was 100% with no errors.

All the result and tested string will describe in the result and analysis section.

5. Approach

The operations of encoding and decoding are coded using python language, with different libraries to help designing the encoder and decoder.
First, all frequency values appear in table1.1 above are stored in excel file, in order to read it easily.

The function readFile() was created, to read the frequency values from the excel file as follow:
The value of the first column in each row was splitting depending on "/", this gave two values of frequency only 100 and 200 Hz, and the other frequencies was read normally from the file and stored in a dictionary.
In the encoder part, the system first ask user to enter a specific statement or string, it can have spaces, capital and small letters. Then defining the sampling frequency as required to equal 8000 Hz in 1ms, and knowing that the duration of each character should be 40 ms, and then the samples for each char will be 40ms * 8000 = 320 samples. After that, the list was defined to store the encoded string inside it, to calculate the required time for each string will be the number of character in the string multiplied by the duration of each character, which is 320 samples, this appear in the following code line.

```
t1 = np.arange(0, f * len(string), 1)
```

Figure5. 1 (codeLine-1)

And then using a for loop, to pass through each character in the string, and then represent each character as a cosine signal containing 4 different frequencies, and then all encoded characters are stored in a list defining before.

```
for i in range(0, len(string)):
    encode_string.append(
        np.cos(2 * np.pi * dec[string[i]][0] * t / Fs) + np.cos(2 * np.pi * dec[string[i]][1] * t / Fs) + np.cos(
            2 * np.pi * dec[string[i]][2] * t / Fs) + np.cos(2 * np.pi * dec[string[i]][3] * t / Fs))
```

Figure5. 2 (CodeLine-2)

Then the encoded string is plotted in time domain as a waveform. After that, the list which containing the encoded string is stored in an array, to help to store the encoded string in a .wav file, and then the value of sample_rate and samples are obtained using the function readaudio(), which read the .wav file and return the sample_rate and samples. After reading the file, then the sound is appear, this sound represent the encoded string, the sound is played using the function playaudio(), this function needs the value of sample_rate and samples in order to play the sound of signal.
Getting to this point, the input string encoded to a signal containing frequencies and stored in a .wav file.
After that, the decoder is needed to reconstruct the message (original inputted string) from the encoded file (.wav file).
The decoder is implemented using two different methods as described below.
The first method to decode the encoded string is using frequency analysis (Fourier transform), which is represented as decodefft() function, which starts with reading the stored .wav file to get the sample_rate and samples, then playing the sound using the values of sample_rate and samples.
Then a variable called period is defined to get the value of time duration of each string which is the integer value of (sample_rate * 40ms), which means the value 320.
Then an empty string is defined to store the decoded string after decode it.
Then a for loop from 0 to take different values of samples, then defining an empty list to store the values of samples, then the list changed to array in order to take the Fourier transform of it.
The Fourier transform token using the function fft(), this function takes the array of samples, and return the frequency response of it, which containing the phase and amplitude. But there is no need for phase value, so that the absolute value of the frequency response was taken.
Then the value of peaks was found by using the existing function find_peaks(), then each peak found was multiplied by

the value 25 in order to get the real value of the frequency corresponding to each peak.
The value 25 is obtained by dividing the sample_rate by the samples for each character, which means 8000 / 320.
Then the values of frequencies are stored in a list, and the comparison operation started, each value of peak is compared with values stored in the dictionary containing frequency values for characters, in order to recognize the encoded character.
Then the value of the encoded signal is plotted in frequency domain.

The second method used for decoding the signal is using band pass filters, this method is represent in the function decodefilter(), which starts with getting the unique frequency values without repeat, and this is done using the function getfrequencies() method which uses a set to find the distinct frequency values and store it in the set.
And in the same way as the first method, the values of sample-rate and samples were obtained.
And then a number of band pass filters was created, specifically 11 band pass filters, because as appear in table1.1 there are 11 different value of frequencies.
The band pass filters created by using the method bandpassfilter(), which works as follow:
First, the frequencies is obtain as described above, then a variable called order was defined with value 2, to show that the filters to be designed are second order filters.
Then a for loop used from the first value to the last value of different frequencies, and a band pass filter is created with center frequency value appear in the list of unique frequencies, with lower and upper band value equal to center frequency ±10.

After multiple attempts, it seems that the band pass filters don't cover all frequency values, so that when the frequency is greater than 4000 Hz a high pass filter is created, otherwise a band pass filter is created, as follow:

```
if hieght >= 4000:
    [b, a] = scipy.signal.butter(order, lowc, btype='hp')
else:
    [b, a] = scipy.signal.butter(order, [lowc, hieghtc], btype='bandpass')
```

Figure5. 3 (CodeLine-3)

Then the band pass filter that designed, it uses the Z transform with numerator and denominator
Returning to decodefilter() function, and after obtaining the frequencies and required values to design the band pass filters, the decode operation start.
A for loop is generated to get the samples of each character, knowing that each character has 320 samples, these 320 samples with there signals containing 4 different frequencies will pass through all band pass filters with all different center frequencies. Then when the filter recognize a value then it will generate a large power, which means that the center frequency of this filter is one of the 4 different frequencies, and when the power is very small, then this means that the center frequency for this filter is not a one of the 4 frequencies, the power is calculated as follow :

```
y1 = np.sum(y1 * y1)
```

Figure5. 4 (CodeLine-4)

Then after obtaining the frequencies, the comparison operation with the frequencies stored in the dictionary starts, to get the correct characters to form the correct string.

The last function used is the accuracy() function, which will return the performance and the accuracy of the system, this function depends on comparison between the input string (encoded string) and the decoded one, then the number of matching letters counted, the value of the accuracy is calculated by dividing the number of matching letters over the length of the original string and then multiplied by 100.

6. Result and analysis

The system was tested using a lot of inputted strings using the evaluation criteria described in evaluation criteria section above.

1. First test:

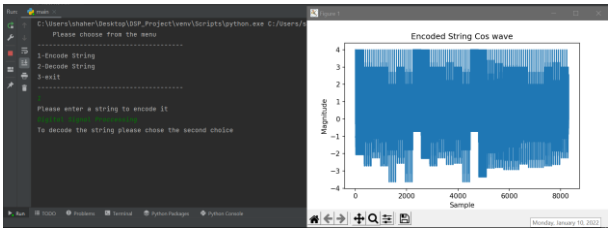

Figure4. 1 (Test-1-encode)

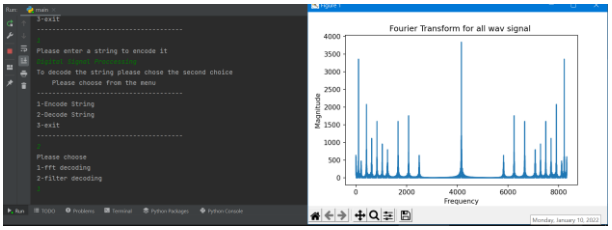

Figure4. 2 (Test-1-fftDecode)

```
The encoded string is =>  Digital Signal Proccessing

The accuracy =  100.0
```

Figure4. 3 (Test-1-Result and accuracy-fftDecode)

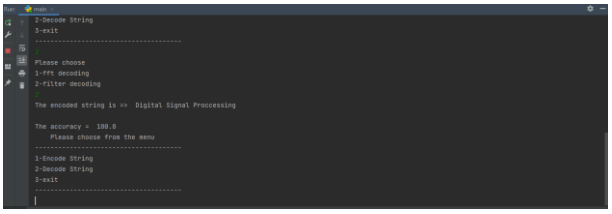

Figure4. 4 (Test-1-fiterDecode with result and accuracy)

From the first testing example, the input string was (Digital Signal Proccessing), each character on the statement was encoded to a signal, and decoded. As appear in the result of decoding the string (using the two methods of decoding) the system has an accuracy equal to 100%, which means that all encoded character (Digital Signal Proccessing) has been decoded correctly.

2. Second test

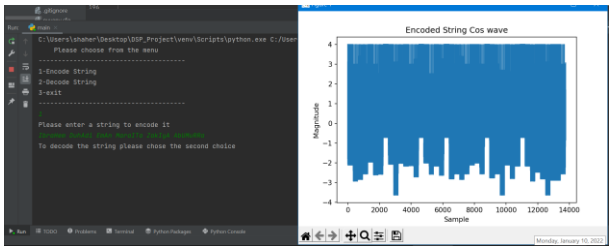

Figure4. 5 (Test-2-encode)

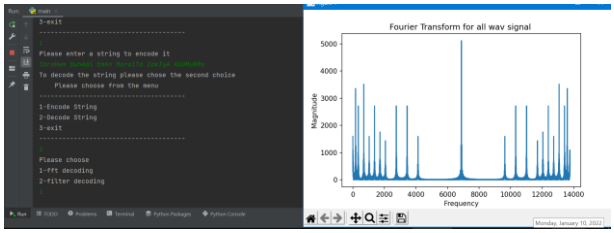

Figure4. 6 (Test-2-fftDecode)

```
The encoded string is =>  IbraHem DuhAdi EmAn MaraITa ZakIyA AbUMuRRa

The accuracy =  100.0
```

Figure4. 7 (Test-2-Result and Accuracy-fftDecode)

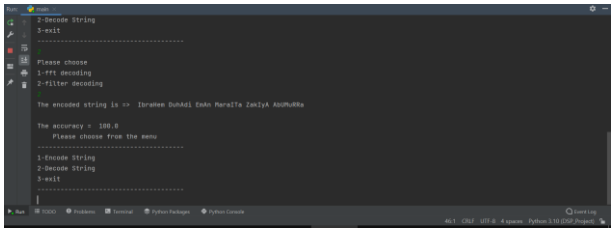

Figure4. 8 (*Test-2-fiterDecode with result and accuracy)*

As can see in the above figures for the second testing example, when the input string to the system was (IbraHem DuhAdi EmAn MaraITa ZakIyA AbUMuRRa) which contains capital and small letters with spaces, the result of encoding this statement using the two methods gave an accuracy of 100%, which means that the system decode the input strings without errors or noise.

7. Development

The system can be develop by making it more general, which means that the encoding and decoding operations designed to be able to work with any characters, commas and numbers, in other word, design a decoder and decoder able to recognize all ASCII characters.

8. Conclusion

In this project, we learnt a lot about how we can send a string message over the network, it let us to think and understand how this operation is done, and how it can be a practical system by designing and implementing two basic communication elements, which are the encoder and decoder, these two elements are very important, they can be implemented in different and various, as described in the previous sections.

9. References

[1] https://www.techtarget.com/searchnetworking/definition/encoding-and-decoding