CS 378 Lab 2

Nimish Milind Manware (22b0944) | Jayesh Jadhav (22b1056)
Daya Swaroop Chavali (22b0939) | Chandrakant Pradhan (22b1023)

September 1, 2024

1 Introduction

In this assignment we are trying to simulate a wireless computer network using sound as a medium. This document contains a brief overview of how we are going to achieve it. It includes

- Identifying the beginning and end of message blocks
- Encoding bitstrings as signal / waveform
- Sound transmission of waveform
- Retrieval of original bitstrings at the receiver end after error correction

2 Preamble

We will use a string of bits before the message which we called as preamble in the class. This would help us in detecting the start and end of the message. Our preamble would look like following

1	0	1	0	1	0	1	0	1	0	1	1						· · · · ·	· · · ·
										END		**Length Bits**					**Message bits**	**CRC**

The first few bits are alternating 0's and 1's that is added as redundancy for handling initial bit loss. The consecutive "11" tells us that the message is about to begin. But before the actual message begins there are 5 bits representing the length of the message which will help us to detect the end of the message. Also we will add CRC to the message which will be useful to detect and correct error on the receiver end.

3 Transmission

For transmitting the bitstring we are going to convert it into waveforms or signal. This waveform will be played by the pyaudio library. This waveform is generated as follows:

- For every set bits in the message we will use a specific frequency in signal for a particular interval.
- For every unset bit we will use some other frequency which will be significantly different from the first frequency.

After playing the sound on sender end and recording it on the receiver end we would do the following to retrieve the message.

- We will FFT to get the most dominant frequencies of the signal in every bit interval
- These frequency will be compared the frequencies corresponding to 1's and 0's

The following figure shows the frequencies detected. As we can see the bits are quite recognizable. The color of the frequency depend on the intensity of that frequencies.

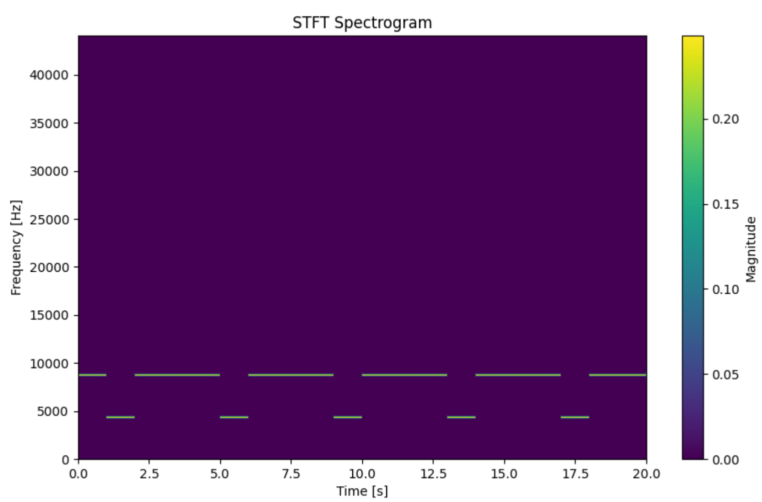

Figure 1: Frequencies

4 Error Detection and Correction

After retrieving the final transmitted message as bitstring. We will use the preamble to get our message back, which will have original message and CRC in it along with some flipped bits. Now to detect and correct the potential errors in the message we will employ a CRC with minimum Hamming distance 5. So that even if 2 bits are flipped we make sure that those errors are detected.

$$\text{Minimum Hamming Distance} = 2 * t + 1 \tag{1}$$

Here t is the number of allowed bit flips. The generator polynomial we have chosen is $x^{10} + x^8 + x^7 + x^6 + x^4 + x^2 + x + 1$ This generator is of length 12 (i.e CRC of length 11).

4.1 Detection

We will use this generator polynomial to get the CRC for the message by dividing the message (after adding 11 0's) by the generator polynomial's bit string. After appending the CRC we will be able to detect error by checking that the remainder is zero or not.

4.2 Correction

To correct errors we will check with flipping every possible combination of single or two bits and get that message which has remainder zero. Whichever bit string gives us zero remainder, we will conclude it to be the orignal message. Then it will parsed accordingly.