ECE 340: Digital Communication Systems | Due Date: Nov. 30, 2021

Assignment

- The institute academic code of conduct will be strictly enforced. The first instance of plagiarism or other unfair means would lead to a grade reduction. The second instance would lead to the student failing the subject.
- The assignment can be implemented in either MATLAB or Python. Thus, wherever 'MATLAB' is mentioned below, you may use any programming framework instead of it.
- Write a separate function for each of the following steps.

The aim of the assignment is to design an end-to-end digital communication system implementing each of the 'blocks' discussed in the class. The steps to implement the same can be as follows:

1. Generate an input analog/digital signal, e.g., by any of the following.
    - Record an audio signal using any microphone app of your mobile phone and store it into one of the standard audio file formats.
    - A text file obtained from the internet.
    - A snippet of a video file that can act as the input to your MATLAB program.
2. Import the signal into MATLAB and display the characteristics of the signal in the time domain (amplitude and phase of the signal) and in the frequency domain (the spectrum). Use filters wherever necessary to restrict your signal/spectrum to a finite domain. Develop a function *input()* to do this.
3. Find out the maximum frequency component of the spectrum (after possible filtering), and sample the signal using i) a frequency less than the Nyquist rate, and ii) a frequency greater than the Nyquist rate. Display the spectrum of the sampled signal for both the cases. Develop a function *source()* to do this.
4. Choose a reasonable (up to your discretion) number of levels for quantizing the sampled outputs. For your choice of levels calculate the MSE of the quantization. Plot an MSE vs number of quantization levels graph for different values of quantization levels. Develop a function *quantize()* to do this.
5. For your chosen value of quantization levels (say = 16 or 32 or higher etc.) encode the samples into bit-streams. For example, if you have 32 quantization levels, choose 5 bits to represent each bit. Find out the total bits used in this representation.
6. For the chosen number of quantization levels, find out an empirical probability distribution of the occurrence of the symbols coming out of the quantizer by simply counting the number of times each symbol is occuring dividing by the total number of symbols. Use this probability distribution to do a Huffman coding of the symbols. Find out the number of bits in the encoded bit-stream. Develop a function *encode()* to do this.
7. Choose a PAM line-code, i.e., 0 is represented by -1 and 1 is represented by +1. Design a raised-cosine filter (don't use built-in functions) that takes the inputs as the output of your Huffman coded data and converts them into subsequent pulses. Vary the roll off factor of the filter to visualize the output of three consecutive input bits (i.e., plot it in the frequency and time domain) and comment on the ISI. Develop a function *pulseshaping()* to do this. In the time domain, it will consist of a sequence of modified sinc functions.

8. **Bonus: 5** Marks Assume that the transmission is only in base-band and noise-free. Thus the output of the previous step is directly the output of your channel. Design an appropriate sampler to sample the output and store the sampled values (which can have any value due to ISI). Design a decoding rule to get back the transmitted bits. Perform Huffman decoding and subsequently a reconstruction operation to generate the output waveform. Convert it back to the video/audio/text file and compare the input and output. Develop functions *receivedecode()*, *tablelookup()*, etc. to do this.

Write a report showing the outputs of each step as mentioned and the methodology used.