Digital Communication Mini project

C Srichandra-13EC30008

15 April 2016

1 Objective:

To design a communication system where input file(text/image) is to be communicated via a noisy channel and received at the output.

2 Design Steps:

2.1 Source Encoder:

We have found the probability distribution of characters in text file(pixel values in image). .

Using these probability values we have performed **Huffman encoding** and generated bitstream of 1's and 0's. In case of image, we have reshaped the matrix having the pixel values into single array by concatenating the rows. We have created a cell containing huffman codes and used this codes we have encoded the whole file(text/image). This code is included in the source$_encoder.mfile$.

2.2 Modulation and Noise:

Given specifications and constraints are:

- Bandwidth = 12MHz
- Transmission time should be less than 30ms

We have found the bitstream lengths to be

- Text file : 1480608
- Image file : 1957756

From [1], we have

- Symbol rate = a x Bandwidth
- Bit rate = k x Symbol rate

where a is proportionality constant and k is no.of bits per symbol. In practice a=1.8. If we choose 16-QAM we have

- M = 16
- k = 4
- Bandwidth = 12MHz
- Symbol rate = 21600000 symbols per sec
- Bit rate = 86400000 bits per sec
- Transmission time for text file = 17.13 msec
- Transmission time for image file = 22.659 msec

Clearly, all constraints are met and we have used 16-QAM modulation. We have divided the bitstream into symbols each of 4 bits. We have assigned real part and imaginary part to each symbol. Realpart and imaginary part values are (-3,-1,1,3). Distance between any two points in the constellation is 2. We have then passed this modulated stream through additive white gaussian noisy channel of various SNRs. This code is in qam.m file.

2.3 Receiver(Demodulation):

For the demodulation, we have first generated the reference points of the constellation. We have then compared the euclidean distance of the received symbols with the reference points and assigned it to the nearest reference point. We have calculated the actual bit error rate and expected bit error rate [2]. We have plotted these for various SNR per bit values of 5dB, 10dB and 15dB . This code is in qamdemod.m file

2.4 Source decoder:

Using the huffman code dictionary, we have created a binary tree. We have then traversed through the bit stream and then using tree created we have decoded the signal. We have then generated an outfile text file(image file) and also an error file. In case of error file for the text, we have used * wherever error occured. In case of error file for the image, we have blackened the pixel value, wherever error occurred. This code is in sourcedecoder.m file

3 Results:

Results for the text file have been formulated in the table below.

SNR	Expected BER(QAM)	Actual BER(QAM)	Errors in Decoded file	Total chars in file
5	0.041892303181263	6.75398214787438E-07	305829	326559
10	0.001754150617893	6.75398214787438E-07	301262	326559
15	1.84185551109448E-07	0	0	326559

Results for the image file have been formulated in the table below.

SNR	Expected BER(QAM)	Actual BER(QAM)	Errors in Decoded img	Total pixels in img
5	0.041892303181263	6.75398214787438E-07	260299	262144
10	0.001754150617893	5.10788882782124E-07	251461	262144
15	5.10788882782124E-07	0	0	262144

For 10dB text file,

- Total time * No.of Errors = 5160.61

4 Plots:

We have plotted Expected BER and Actual BER versus SNR per bit for text file after QAM demodulation.

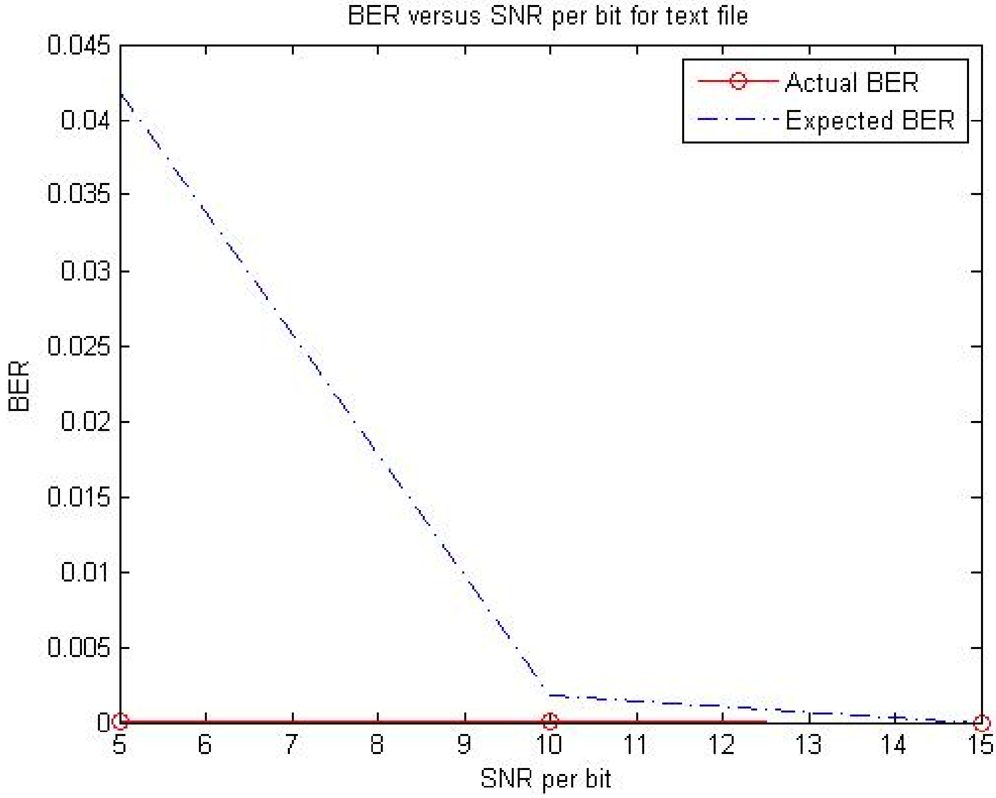

Figure 1: BER versus SNR per bit after demodulation for text file

We have plotted Expected BER and Actual BER versus SNR per bit for image file after QAM demodulation.

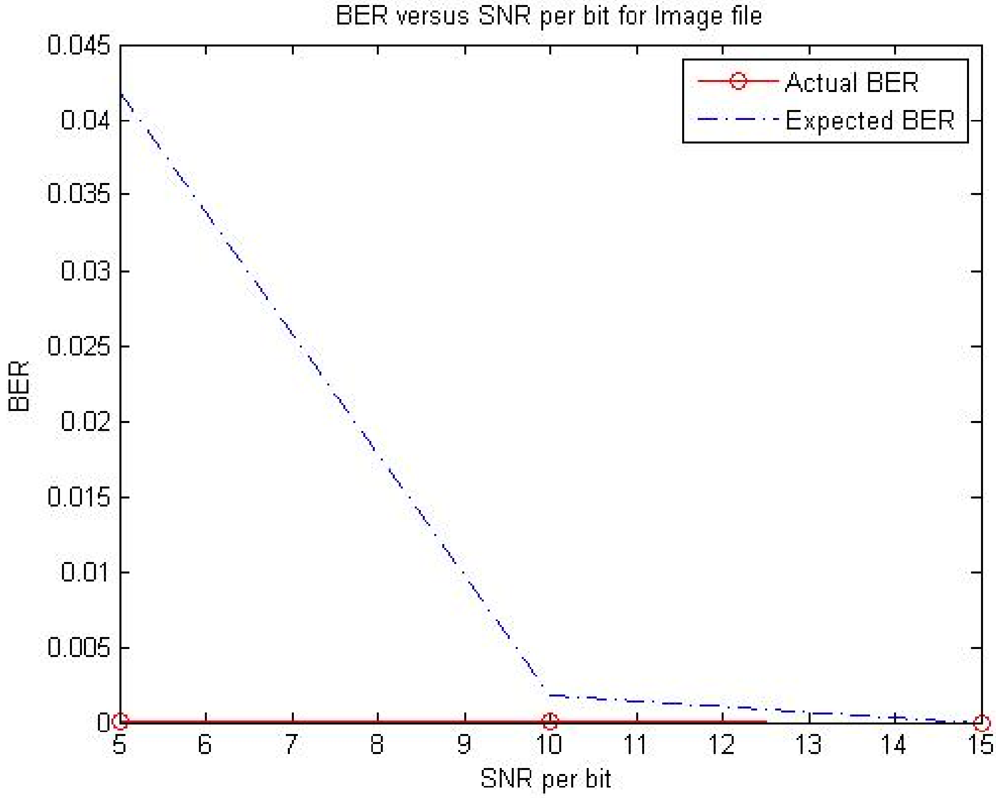

Figure 2: BER versus SNR per bit after demodulation for image file

We have plotted error probability after source decoding for text and image.

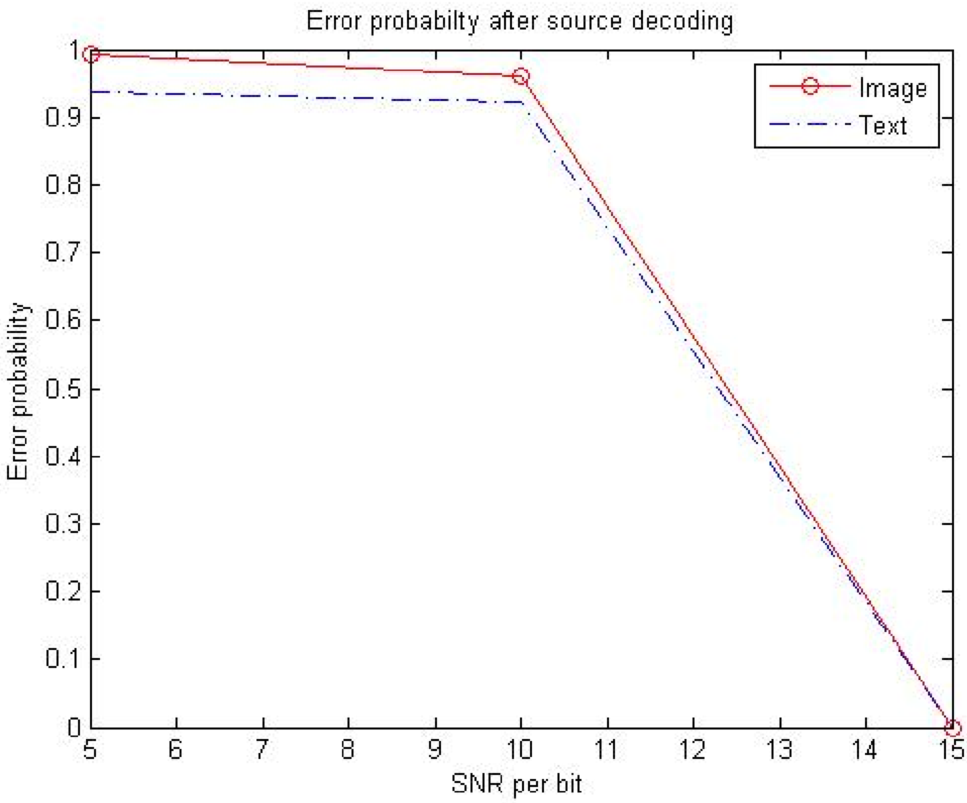

Figure 3: Error probability versus SNR per bit for source decoded file

5 Assumptions:

- While reading a text file file in matlab, three extra ambiguous characters are being added, we have ignored these extra characters.
- We have used QAM-16, in which encoded bitstream is divided into chunks of 4bits. For the original lena.bmp image, we have found that the encoded bitstream length was not a multiple of 4. So, we have modified pixel value at (1,1) co-ordinate to 25(initially it was 162). After modification, encoded bitstream length was a multiple of 4 and we could perform QAM.

6 Discussions:

- Bit error rates of the QAM modulation cannot be used for expecting the error rate after decoding. Error rate after decoding depends on the bit change position leading to the false prediction of corresponding characters.
- Bit error rate for SNR per bit of 15dB was found to be zero in case of both text

and image and correspondingly, outputs were same as input. Also error file was blank in case of text file and error file was pure white in image file, meaning no error.

- Bit error rate was found to be same for 10dB and 5dB in case of text and image. However, after decoding, more information was lost for 5dB case.

- Important information like the edges region pixel values wasnt lost in image for 10dB while most of the information was lost for 5dB. This can be observed from the output image files.

References

[1] URL http://www.eecs.berkeley.edu/~messer/netappc/Supplements/20-digicom.pdf/.

[2] URL http://www.dsplog.com/2012/01/01/symbol-error-rate-16qam-64qam-256qam/.