AC 2008-798: A CLASS PROJECT ON AN LDPC-BASED ERROR CORRECTING SYSTEM

Min-Sung Koh, Eastern Washington University

MIN-SUNG KOH obtained his B.E. and M.S. in Control and Instrumentation Engineering in the University of ULSAN, South Korea, and his Ph. D in Electrical Engineering and Computer Engineering in Washington State University. His interests are in the areas of speech and image signal processing, signal processing in communication, photoacoustics and embedded systems.

Esteban Rodriguez-Marek, Eastern Washington University

ESTEBAN RODRIGUEZ-MAREK did his graduate work in Electrical Engineering at Washington State University. He worked as a research scientist at Fast Search & Transfer before transferring to the Department of Engineering & Design at Eastern Washington University. His interest include image and video processing, communication systems, digital signal processing, and cryptographic theory and applications.

Claudio Talarico, Eastern Washington University

CLAUDIO TALARICO received his Ph.D. in the University of Hawaii at Manoa in Electrical Engineering. He is currently an Assistant Professor of Electrical Engineering at Eastern Washington University. His research interests include design methodologies for integrated circuits and systems and complex systems-on-chips.

David Freiberger, Eastern Washington University

DAVID FREIBERGER is an undergraduate student in Electrical Engineering at Eastern Washington University. He plans on obtaining a graduate degree in Electrical Engineering following his graduation, and to pursue interests in hardware design and signal processing.

A Class Project on an LDPC-Based Error Correcting System

ABSTRACT

The low-density parity check (LDPC) code is an error correcting code that closely approaches the information theoretical channel limit, also called channel capacity. LDPC and Turbo codes are the only two currently known codes that are denominated capacity approaching codes, and are extensively used in communication systems requiring high capacity. It was only after several decades of research, sprung from Claude Shannon's seminal work on the mathematics of communication theory, that a capacity approaching code was designed. Developing a capacity approaching code requires the knowledge of a large variety of different error correcting approaches, generally based on advanced mathematic skills. This knowledge typically is taught in classes dealing with coding theory, error correction codes, or information theory etc. Hence, LDPC codes are seldom taught in an undergraduate curriculum, as they are combined in graduate programs with other coding techniques. However, it has been recently found that LDPC code can be understood from factor graphs, which is a dramatically different approach as that used traditionally in coding theory classes. With the factor graph approach, it is possible for undergraduate students to have an introductory experience to error correcting codes in the LDPC family. This paper documents the findings resulting from a project done in a senior-level Digital Signal Processing (DSP) class. The successful class project proves that it is possible for undergraduate students to understand LDPC codes based on factor graphs, without any other traditional coding theory background.

I. INTRODUCTION

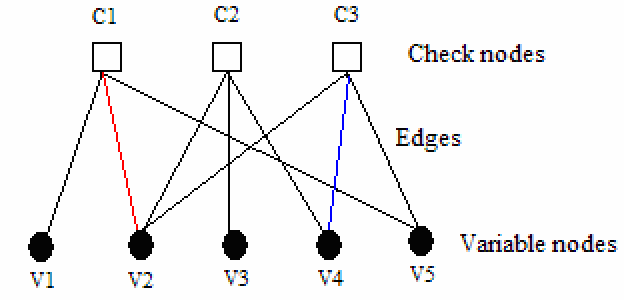
Our lives are a daily succession of conscious information exchanges. We talk to people, listen to the radio, watch TV, browse the Internet, make phone calls, check the stock market with our cell-phones, send text messages, etc. Some information exchange is done without us even realizing it. We swipe our debit card to buy an espresso and a connection is immediately made to transfer the funds from our bank account to that of the vendor. We approach a grocery store entrance and a signal is sent that opens the door automatically. Our cell-phones are constantly sending location information so that we can be located immediately, if needed. We enter a bank and a closed circuit television system is alerted to begin recording a security video. Most of these examples include some form of electronic communication technology. The amazing rate at which information availability is increasing has also increased the number of Internet users to over 1.5 billion people, a 225% increase since the peak of the dot com boom in the year 2000 [1]. Another electronic communication technology whose use has increased significantly in the last few years is mobile-phones. Over 2.4 billion people use cell-phones [2]. Third-world countries show the largest increase in cell-phone usage, since no tangible channel (i.e. copper lines, optical fiber, microwaves, etc.) is needed (other than ubiquitous towers) to enable communication. Although a plateau has not been reached yet, further increases in availability of Internet and cell-phones (and any other communication systems) are limited by one factor: *channel capacity*. Any channel used for electronic communication has an inherent capacity (i.e. transmission limit) associated with it. This leads to a need to add more infrastructure (i.e. new channels). While building new channels solves the problem, it is costly and time intensive. Another way of solving the problem exists that is both cheaper and faster to implement: improve existing transmission algorithms to better utilize channels already in place. These improved transmission algorithms, also called channel codes, or simply codes, will lead to cheaper and faster communication, and at the same time will open the way to modern communication being more easily available to larger segments of the world population.

One of the findings in Shannon's seminal paper in 1948 [3] was the proof of mathematical limits to the capacity of given channels. The immediate consequence of these results was a new area of research called channel coding. The main goal of channel coding is to find codes that approach the theoretical limit predicted by Shannon. The best existing codes are called *capacity approaching* codes, as they achieve reliable communication at a rate approaching Shannon's limit. Since 1948, several efficient channel codes, which can detect and fix errors in various scenarios have been developed by different researchers in the decades following Shannon's findings. It wasn't until 1993, however, that the first capacity approaching code was developed by Berrous et al [4]. These codes were called Turbo codes. After this landmark paper, another capacity approaching code was presented by Mackay and Neal in the 1990s [5, 6]. Mackay and Neal's work was heavily based on resuscitating low-density parity check codes, first introduced by Gallager in 1963 [7]. At the time Gallager had introduced LDPC codes, the algorithm was mostly ignored by the coding community because of its complexity, which was prohibitive with the technology existing at the time. Improvements in semiconductor manufacturing allowed both of these technologies to be implemented. Hence, it is required for undergraduate students having an interest in the communication field to gain some experience in channel codes or error correction codes to understand overall communication systems. Due to continuous development of coding theories to understand capacity approaching codes, it was recently found that Turbo codes and LDPC codes can be explained by one generalized approach called "codes and graphs" [8, 9]. This leads to many other algorithms to be clustered under the "factor graphs" family. Further, one popular LDPC decoding method called the "Sum-Product" algorithm was introduced based on factor graphs. Using this approach, the design of efficient LDPC codes becomes, in essence, the finding of optimal ways of connecting each node in the factor graphs. This approach leads to a way to introduce error correcting code of LDPC to undergraduate students without many advanced technical topics used in traditional coding theory.

Even though LDPC codes can be understood through linear algebra, factor graphs and traditional channel coding, this process is not straight forward. The mathematics involved in channel coding can be overwhelming to an undergraduate student. The large number of channel coding methods and their inherent complexity often results in students losing interest in the topic. This paper introduces a class project that is both intriguing to the student and manageable in a typical undergraduate curriculum based on basic factor graphs and simple linear algebraic operations. Note that a basic knowledge of linear algebra is useful in understanding the algorithms, but not critical within the context of communication systems. Not only is the project tailored to maintain students' interest, but it also helps students to obtain a holistic understanding of a communication system including error correction, without requiring graduate-level classes in coding theory. The project includes a GUI implementation, where different data sources can be selected for encoding with LDPC. The encoded signal is modulated and then contaminated with various types of noise in the simulated channel. The signal recovered in the decoder is compared with the original and a graph for bit error rate (BER) visualized in the GUI. A goal of this class project is to allow students to experience error correction codes and see its functionality within a holistic overview of communication systems.

II. ENCODING OF LDPC CODE

In order to transmit signals over noisy transmission channels such as seen in radio frequency communication and similar applications, it is necessary to incorporate error checking codes. Generally a form of parity-checking is used, in which extra bits are added to the transmission through an encoder, allowing a decoder at the receiver side to perform constraint checks on each bit received. These parity checks allow the receiving device to remove errors from the received signal. Unfortunately, it is not possible to encode a signal that absolutely guarantees equality between sent and received data, as proved by Shannon [3]. However, low-density parity check (LDPC) codes demonstrate high performance capabilities, arbitrarily close to the Shannon limit, and are becoming feasible with today's processing technology [10]. LDPC codes are founded on basic linear-algebra principles. In this paper math is performed in the binary subspace of real-numbers, and hence addition and multiplication are performed in base 2, e.g. addition becomes the XOR operation, and multiplication becomes the AND operation. LDPC codes utilize a *sparse* binary parity-check matrix, H, with dimensions $M \times N$. Matrix H can be either regular, meaning that there are a specific number of 1's per row and column, or irregular, in which there may or may not be a constraint on the number of 1's. In this project we will deal only with an irregular parity-check matrix. The sparseness of H means that there are a very low number of 1's in H, compared to it's total size. Matrix H can be expressed by a factor graph. One example showing the relationship is shown in Figure 1. A graph called Tanner graph (factor graphs can be understood as advanced graphs based on Tanner graph) is shown in Figure 1-(a) and the corresponding H matrix is shown in Figure 1-(b). In Figure 1-(a), there are three check nodes and 5 variable nodes. Hence, the graph can be expressed by the 3×5 H matrix shown in Figure 1-(b). In addition, the first check node in Figure 1-(a) is connected to the 1st, 2nd, and 5th variable nodes, which means there are 1's in the 1st, 2nd, and 5th columns of the first row of the H matrix shown in Figure 1-(b). The 2nd and 3rd rows of H are found in a similar manner. The 1's in each column of H imply the connection between check nodes and variable nodes in the graph of Figure 1-(a).



(a) Tanner graph.

$$
H = \begin{array}{cl} \text{variable nodes} & \\ \begin{bmatrix} 1 & \textcircled{1} & 0 & 0 & 1 \\ 0 & 1 & 1 & 1 & 0 \\ 0 & 1 & 0 & \textcircled{1} & 1 \end{bmatrix} & \text{check nodes} \end{array}
$$

(b) Matrix *H* corresponding to the graph in (a).

Figure 1. The relationship between a graph and matrix *H*.

The parity-check matrix is used to encode a message or codeword **x**. Both the sending and receiving device have a copy of *H*, which for example can be implemented as a lookup table in hardware memory. The columns in *H* correspond to the bits of the codeword, and the rows in *H* correspond to the parity-checks on the codeword [10]. A valid codeword will satisfy $Hx = 0$. The meaning of the equation, $Hx = 0$, can be explained as illustrated in Figure 2. When a codeword x={1 0 1 1 1} is received, each check node performs the binary XOR operation with the corresponding bit in *x*. If all check nodes generate a 0, then it means that it is a valid codeword, as shown in Figure 2.

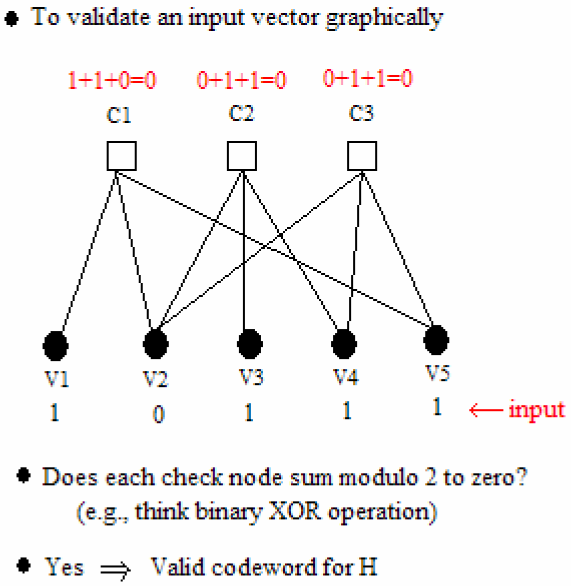

• To validate an input vector graphically
1+1+0=0
0+1+1=0
0+1+1=0
C1
C2
C3
V1
V2
V3
V4
V5
1
0
1
1
1
← input
• Does each check node sum modulo 2 to zero?
(e.g., think binary XOR operation)
• Yes ⇒ Valid codeword for H

Figure 2. Pictorial representation of how valid codeword can be check by a graph

Several algorithms for the encoding of *x* are available in [10]. The method used in this demonstration creates parity check vector bases using sparse LU decomposition. The algorithm attempts to create a lower-triangular and upper-triangular matrix that is as sparse as possible, and that satisfies for equation, $Hx = 0$ [11,12]. The algorithm is described in Radford M. Neal's presentation Sparse Matrix Methods and Probabilistic Inference Algorithms [12] and consists of the following basic steps:

*1) Partition H into an invertible M ×M left part **A,** and a M ×N right part **B**.*
*2) Partition the codeword x into M check bits **c** and N-M source bits **s**.*

$$[A \,|\, B]\begin{bmatrix} c \\ - \\ s \end{bmatrix} = 0$$

$$\Rightarrow Ac + Bs = 0$$

$$\Rightarrow c = A^{-1}Bs$$

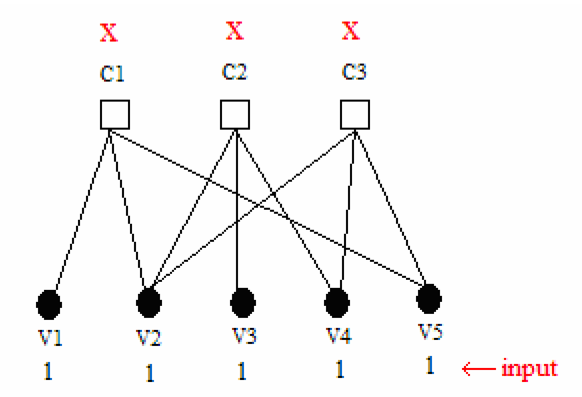
*3) Set **z = Bs** and row reduce **Ac** = **z** to find **Uc = y**, where **U** is the upper triangular matrix.*
*4) Record the reduction as the solution to **Ly = z**.*

All of this is shown in [12]. Combining the check bit vector **c** and the codeword *x* gives a vector **[***x***|c]** that can be transmitted to the receiver for decoding.

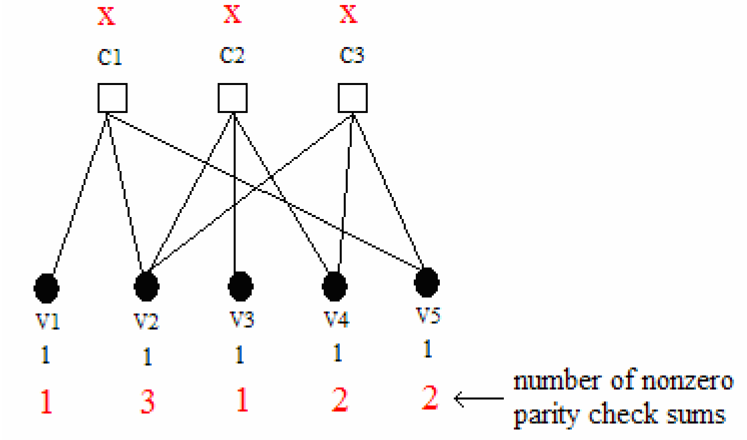
III. DECODING OF LDPC CODE

Decoding for this demonstration is achieved using four different sum-product algorithms as implemented in MATLAB in [13]. The basic theory behind these instances of sum-product algorithms was first developed by Robert Gallager in the 1960's, and is known as belief propagation or probability propagation. Belief propagation can be implemented in several ways, all of which attempt to calculate the probability of a message bit being 0 or 1, using probability and likelihood ratios [11]. The main idea behind simple LDPC decoders is widely known as the Bit-Flipping decoding algorithm [7]. This algorithm can be explained with Figure 1 or Figure 2. Assume a received codeword is *x* = {1 1 1 1 1} in Figure 2. This means that there is an error, because the check nodes in Figure 2 are *not* all zeros (i.e., $Hx \neq 0$) for *x* = {1 1 1 1 1}. Comparing the *x* = {1 0 1 1 1} (i.e., error-free codeword) given in Figure 2 with *x* = {1 1 1 1 1}, we know that there is an error on the second bit of the received codeword, *x*. The underlying idea for decoding LDPC can be explained as follows.

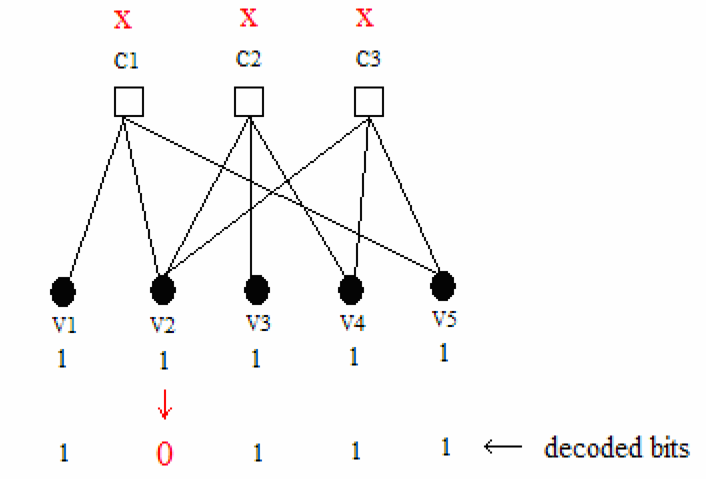
1) *Compute each check node sums with the given codeword. (i.e., x = {1 1 1 1 1}). It is shown in the following figure, where "X" denotes that parity check sum is not zero.*



2) *For each variable node, count the nonzero parity check sums.*



3) *Flip the bit of the variable node having the largest number of nonzero parity check sums.*



4) *Repeat steps 1) through 3) until all parity check sums are zero.*

The above concept can be modified using different mathematical methods for calculating probability rates. This leads to various decoding algorithms for LDPC, such as log domain, simplified log domain, and probability domain sum-product decoding algorithms [15]. Further, the algorithms can be iterated to increase accuracy, and some of them have been shown to reach bit-error rates (BERs) close to the Shannon limit [5, 15].

IV. GRAPHIC USER INTERFACE SIMULATION OF LDPC CODES

To simulate the error correction functionality of LDPC codes, the simple communication system shown in Figure 3 was implemented using a Graphic User Interface (GUI) environment developed with MATLAB. The communication system is shown to include source formatting/deformatting, LDPC encoder/decoder, modulation/demodulation, and a loop eliminator. Binary phase shift keying (BPSK) is used for the modulation and demodulation blocks. To observe the performance of the LDPC code, the results of error correction code of LDPC is compared with the results *without* any error correcting mechanisms. This is shown in Figure 3 as the bypassing branch of LDPC encoder and decoder. Figure 3 also shows a loop elimination block. Loops are related to the performance of LDPC [14], and generally refer to any closed path starting from one check node and ending at the same starting check node in a graph. The loop elimination algorithm to remove any loops composed by 4 paths, or ones, in a graph or *H* matrix is implemented in the class project. The loop elimination algorithm is performed off-line (i.e., before simulation) so that the resulting *H* matrix that does not have loop 4. The matrix obtained by the loop eliminating algorithm is used in the simplified communication system shown in Figure 3. To simulate a noisy channel, noise is added through software to achieve a desired SNR.

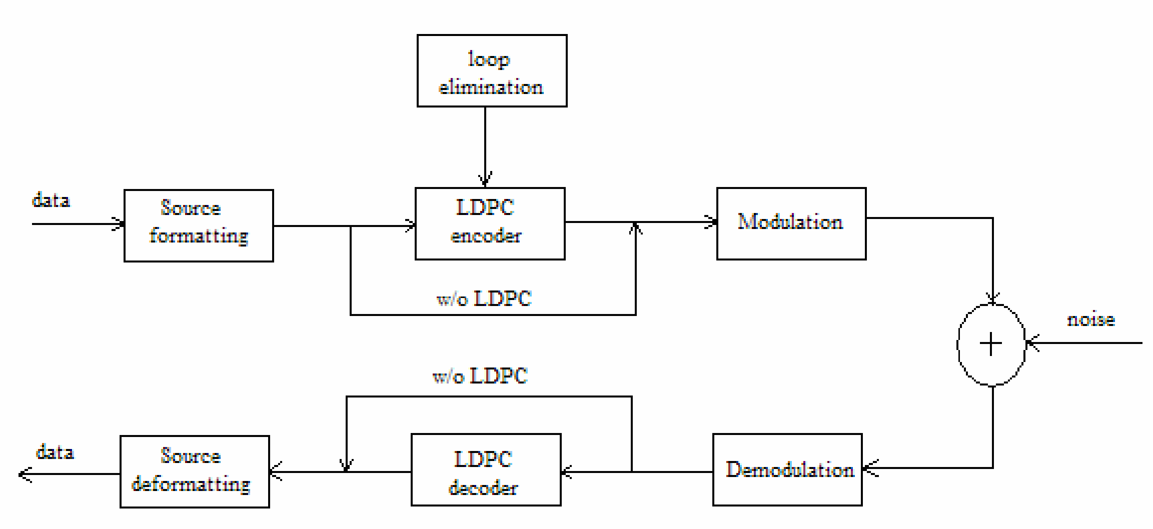
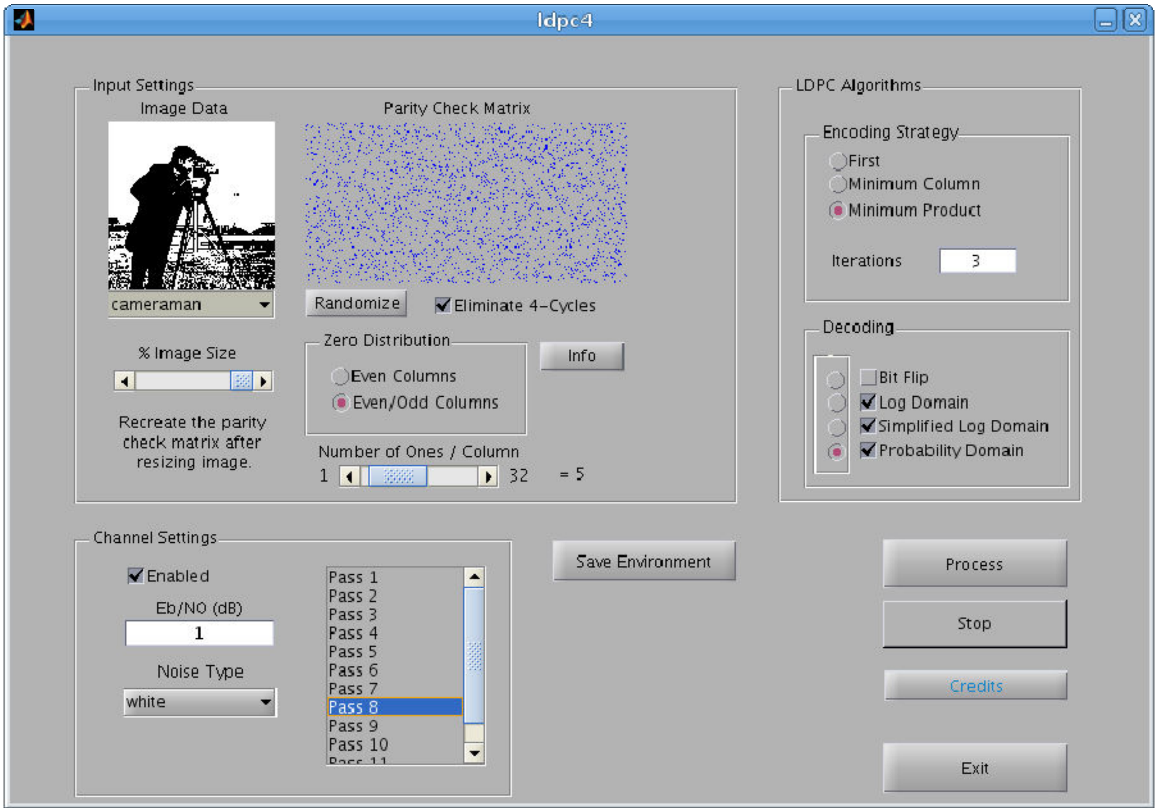


Figure 3. A simple communication system to simulate the LDPC code

The main GUI screen of the finalized class project is shown in Figure 4. The software allows the user to adjust parameters for the encoding, transmission, and decoding processes. The user can choose input data in image form, generate a random parity check matrix based on the size of the image data, choose an encoding strategy, select decoding algorithms, input noise and the type of noise, and finally produce a BER plot and step-by-step process plots. In Figure 4, four different decoding algorithms written by Bagawan S. Nugroho [13] are incorporated into the overall system. The dot plot under "Parity Check Matrix" in Figure 4 is for matrix *H*, where the blue dots are 1s in the obtained matrix. Figure 5 presents another GUI window that shows the process of the implemented LPDC code simulation. A user can observe how error correction codes help a communication system. The results with LDPC and without LDPC are shown in Figure 5 at the given SNR. By integrating all blocks in Figure 3 into a GUI simulation system, students can have a holistic view of communication systems, as well as get a flavor of the importance of an error correcting system. Several parameters given by the GUI system help to better understand the various processes of communication systems.


ldpc4

Input Settings

Image Data: cameraman

Parity Check Matrix

☑ Eliminate 4–Cycles

% Image Size

Recreate the parity check matrix after resizing image.

Zero Distribution
○ Even Columns
● Even/Odd Columns

Number of Ones / Column 1 … 32 = 5

LDPC Algorithms

Encoding Strategy
○ First
○ Minimum Column
● Minimum Product

Iterations 3

Decoding
○ □ Bit Flip
○ ☑ Log Domain
○ ☑ Simplified Log Domain
● ☑ Probability Domain

Channel Settings

☑ Enabled

Eb/NO (dB) 1

Noise Type white

Pass 1
Pass 2
Pass 3
Pass 4
Pass 5
Pass 6
Pass 7
Pass 8
Pass 9
Pass 10
Pass 11

Figure 4. Main screen of MATLAB demonstration software

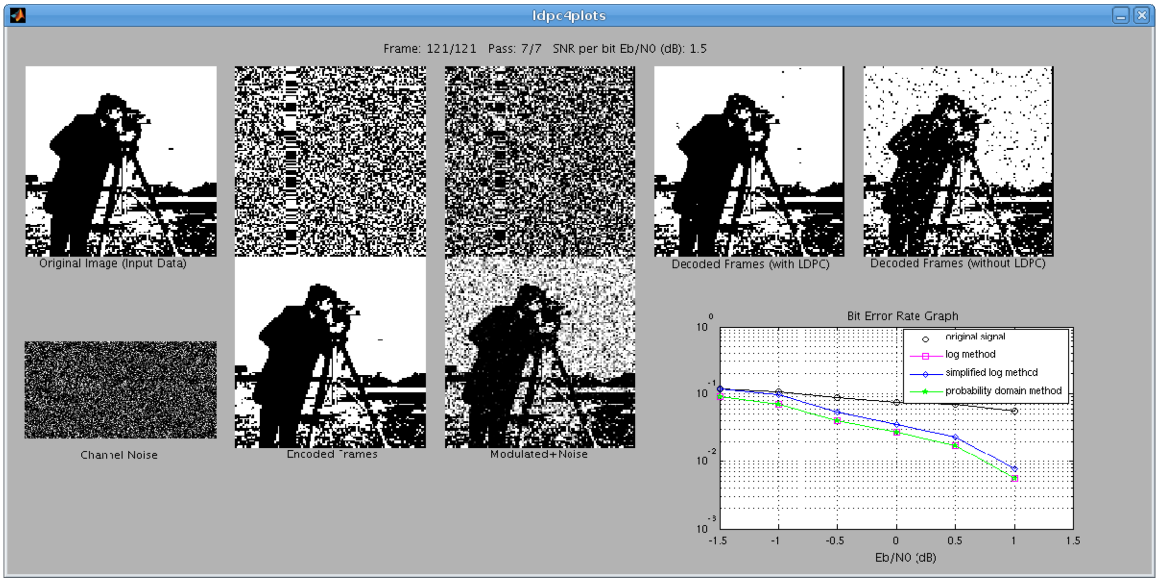


Figure 5. Process step plots, including BER graph

As a final comparison, a BER plot is given in the bottom right of Figure 5 which shows the BER curves for the non-LDPC encoded case, and for the log domain, simplified log domain, and probability domain sum-product decoding algorithms, provided in [13]. The different options in the communication system of Figure 3 are chosen in the GUI main window shown in Figure 4. Data size for the images used is 256×256. The number of iterations for encoding and decoding is set at 5. From the simulation results shown in Figure 5, the functionality of error correcting codes within the context of communication systems can be better understood.

V. CONCLUSIONS AND FUTURE WORK

This paper describes a senior-level undergraduate class project that simulates a communication system using LDPC as the error correction code. Usually, LDPC codes are introduced in coding theory, error correction code, or information theory classes at graduate level. However, the LDPC code itself as a particular example of error correction code can be explained without using advanced mathematics because it can be understood by basic factor graphs and simple linear algebraic operations. Hence, a LDPC error correction simulation is a good sample project that intrigue students and entices them to further the study of coding theory. Moreover, it generates a better understanding of overall communication systems. The class project was successfully implemented based on a GUI environment, and it correctly allows the student to understand the usefulness of error correcting codes within a larger communication system. The successful completion of the project proves that a simple introduction of LDPC codes with graphs and basic linear algebra is enough for undergraduate students to realize the functionality of error correction coding techniques in communication systems. Empirical data shows that students' interest in communication systems was piqued by this class project. Throughout the final presentation, students stated their interest in furthering their studies in the field. Moreover, students suggested that more projects along the lines of the one hereby presented should be included throughout the curriculum.

The Department was encouraged by student comments regarding the project and has decided to include similar projects in upper division classes. We intent to include the data gathered from these projects as another way to measure the achievement of program outcomes related to the "lifelong learning" ABET criterion.

REFERENCES:

[1] "*Internet World Stats*" http://www.internetworldstats.com/stats.htm

[2] "*Why Africa?*" Entrepeneurial Programming and Research on Mobiles. http://eprom.mit.edu/whyafrica.html

[3] C. E. Shannon, "*A mathematical theory of communication*", Bell Syst. Tech. J., vol 27, pp. 379–423 and 623–656, 1948.

[4] C. Berrou, A. Glavieux, and P. Thitimajshima, "*Near Shannon limit error-correcting coding and decoding: Turbo codes*", in Proc. 1993 Int. Conf. Communication, Geneva, Switzerland, pp. 1064–1070, May 1993.

[5] D. J. C. Mackay and R. M. Neal, "*Near Shannon limit performance of low-density parity-check codes*", Elect. Letter, vol. 42, no. 11, 1645–1646, Aug. 1996.

[6] D. J. C. MacKay, "*Good error-correcting codes based on very sparse matrices*", IEEE Trans. Information Theory, vol. 45, no.3, pp. 399–431, Mar. 1999.

[7] R. G. Gallager, *Low-Density Parity Check Codes*, MIT press, Cambridge, MA, 1963

[8] N. Wiberg, H.-A Loeliger, and R. Kotter, "*Codes and iterative decoding on general graphs*", Europ. Trans. Telecommunication, vol. 6, pp. 513–525, Sep./Oct. 1995

[9] N. Wiberg, *Codes and decoding on general graphs*, Linkoping Studies in Science and Technology, Ph.D. dissertation No. 440, Univ. Linkoping, Sweden, 1996.

[10] Amin Shokrollahi, "*LDPC Codes: An Introduction*", Digital Fountain Inc , Fremont, 2003

[11] Radford M. Neal, http://www.cs.toronto.edu/~radford/ftp/LDPC-2006-02-08/

[12] Radford M. Neal, "*Sparse Matrix Methods and Probabilistic Inference Algorithms*", http://www.cs.toronto.edu/~radford/ftp/ima-part1.ps

[13] Bagawan S. Nugroho, http://bsnugroho.googlepages.com/ldpc

[14] J. A. McGowan and R. C. Williamson, "*Loop Removal from LDPC codes*", IEEE Information Theory Workshop, pp. 230–233, Mar/Apr. 2003

[15] W.E. Ryan, "An introduction to LDPC codes," in CRC Handbook for Coding and Signal Processing for Recoding Systems (B. Vasic, ed.), CRC Press, 2004.