CAPACITY AND CODING FOR 2D CHANNELS

A Thesis

by

APARNA KHARE

Submitted to the Office of Graduate Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

December 2010

Major Subject: Electrical Engineering

CAPACITY AND CODING FOR 2D CHANNELS

A Thesis

by

APARNA KHARE

Submitted to the Office of Graduate Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

Approved by:

Chair of Committee,	Henry Pfister
Committee Members,	Jean-Francois Chamberland
	Srinivas Shakkottai
	Natarajan Gautam
Head of Department,	Costas N. Georghiades

December 2010

Major Subject: Electrical Engineering

ABSTRACT

Capacity and Coding for 2D Channels. (December 2010)
Aparna Khare, B. Tech, National Institute of Technology, Karnataka, Surathkal
Chair of Advisory Committee: Dr. Henry Pfister

Consider a piece of information printed on paper and scanned in the form of an image. The printer, scanner, and the paper naturally form a communication channel, where the printer is equivalent to the sender, scanner is equivalent to the receiver, and the paper is the medium of communication. The channel created in this way is quite complicated and it maps 2D input patterns to 2D output patterns. Inter-symbol interference is introduced in the channel as a result of printing and scanning. During printing, ink from the neighboring pixels can spread out. The scanning process can introduce interference in the data obtained because of the finite size of each pixel and the fact that the scanner doesn't have infinite resolution. Other degradations in the process can be modeled as noise in the system. The scanner may also introduce some spherical aberration due to the lensing effect. Finally, when the image is scanned, it might not be aligned exactly below the scanner, which may lead to rotation and translation of the image.

In this work, we present a coding scheme for the channel, and possible solutions for a few of the distortions stated above. Our solution consists of the structure, encoding and decoding scheme for the code, a scheme to undo the rotational distortion, and an equalization method.

The motivation behind this is the question: What is the information capacity of

paper. The purpose is to find out how much data can be printed out and retrieved successfully. Of course, this question has potential practical impact on the design of 2D bar codes, which is why encodability is a desired feature. There are also a number of other useful applications however.

We could successfully decode 41.435 kB of data printed on a paper of size 6.7×6.7 inches using a Xerox Phasor 550 printer and a Canon CanoScan LiDE200 scanner. As described in the last chapter, the capacity of the paper using this channel is clearly greater than 0.9230 kB per square inch. The main contribution of the thesis lies in constructing the entire system and testing its performance. Since the focus is on encodable and practically implementable schemes, the proposed encoding method is compared with another well known and easily encodable code, namely the repeat accumulate code.

ACKNOWLEDGMENTS

First and foremost, I would like to thank my advisor Dr. Henry Pfister, for his valuable guidance, support and encouragement, without which I would have found it difficult to carry out this work. Learning from his research methodology, intuitive thinking, and approach to attack problems, has been a valuable experience for me and I will carry this knowledge with me in the years to come. I would like to thank Dr. Srinivas Shakkottai, Dr. Jean-Francois Chamberland, and Dr. Natarajan Gautam, for being a part of my committee. I would also like to thank Dr. Krishna Narayanan and Dr. Serap Savari for laying the foundations for my understanding in information theory and communications. I would like to thank all my labmates, especially Yung-Yih, who shared a lot of his ideas with me and helped me a lot with my research work. I would like to extend a thank you to my parents and my brother for their constant encouragement, and my friends for making graduate school such a fun experience. A special thanks to my fiancé for all the support, both moral and technical, and for keeping me motivated through the course of my time in graduate school.

TABLE OF CONTENTS

CHAPTER Page

LIST OF TABLES

LIST OF FIGURES

FIGURE Page

CHAPTER I

INTRODUCTION

A. Two-dimensional signals

A signal that can be described mathematically as a function of two independent variables is called a two dimensional (2D) signal. Most signals in communication networks are currents or voltages that vary with time and hence are one dimensional (1D). In practice, a 2D signal is an image. A digital image consists of a 2D grid of pixels, each of which has a value, which may be the intensity of the pixel, for example. For the channel in question, the input signal and the output signal are both 2D signals.

B. The print-and-scan channel

In this work, we would like to present a way to store digital information on paper, and discuss a way to efficiently retrieve it. The channel under consideration for our work consists of a printer and a scanner. The basic communication system is the following: the signal is encoded and printed on a piece of paper. Then it is scanned using a scanner and decoded.

The most conventional application of this channel and system is that of 2D barcodes. One can also think of a few more innovative applications. For example, the emergency information for a building could be posted outside all exits in machine readable

---

The journal model is *IEEE Transactions on Information Theory.*

format. In case of an emergency, this would enable the emergency personnel to use digital aids, like a PDA, to read this information and expedite procedures.

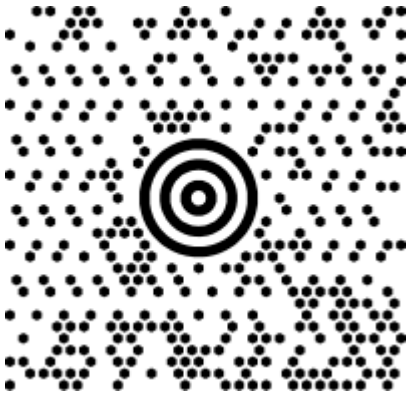
C. Motivation and similar work

The motivation behind the work was to find out the information capacity of paper over the print-scan channel as described above. Paper is a very common mode of communication which we used everyday.

Data transmitted through this channel faces a number of impairments. It can be scaled, translated, rotated, faded and noise can be added to it. Simply encoding the data with error-correcting codes is not adequate to handle all of these distortions. Existing 2D barcodes use a number of techniques to handle these distortions. Figure 1 shows a few 2D barcodes used in practice.

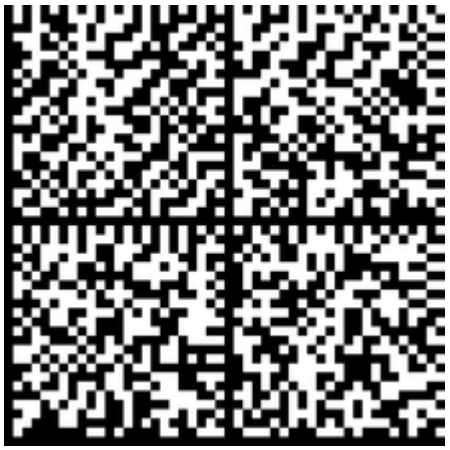
The Aztec code consists of a square bulls-eye pattern, and the encoded data is arranged between the concentric square rings. It has orientation marks on the four corners of the outer square that are meant to help in alignment of the code in case of any rotation or reflection. This code supports a maximum of 1914 bytes of data. A rate-configurable Reed Solomon code is used to encode the data. Similarly, the other codes consist of patterns to recover from various types of distortions. However, most of these codes are design to be very robust against the impairments and support relatively low data rates. The focus of our work is to find out how much data we can pack onto a given area of paper.

(a) Aztec code



(b) Maxicode

(c) QR code



(d) Data matrix

Fig. 1. 2D barcodes used in practice

D. Outline of the dissertation

In Chapter II, we discuss the channel model. First, we consider the model of communication and the flow of signals in the system. Then, we talk about the interference models of the channel based on the physical parameters. Next, we discuss the noise and distortion models introduced by the channel. Finally, we present the overall model used for the system simulation.

In Chapter III, we consider various channel coding schemes. In particular, we discuss low-density parity-check (LDPC) codes and LDPC convolutional codes and propose a code for the system and a simple encoding algorithm for that code. We also discuss the overall belief-propagation decoder used to jointly remove ISI and correct errors

In Chapter IV, we consider a range of techniques for inter-symbol interference removal, and methods for image restoration based on the channel model.

In Chapter V, we present the simulation results for our system and compare it with another easily encodable coding scheme.

Finally, in Chapter VI, we describe the experiments done as a part of this work with the physical print-and-scan channel, the challenges involved, and our observations and conclusions about the system.

CHAPTER II

CHANNEL MODELING

A. Communication model for the channel

The purpose of a communication system is to transmit information from one medium to another via a communication channel that adds some distortion to the transmitted signal.

The signal flow diagram for our communication channel is shown in Figure 2. Let the

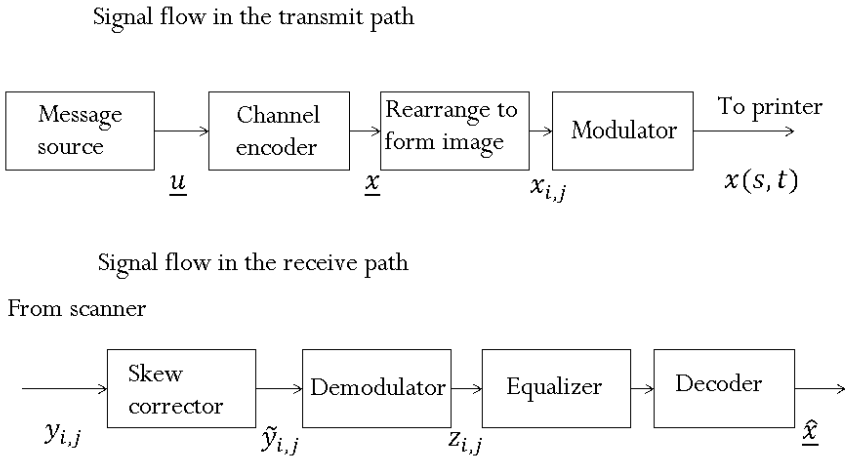


Fig. 2. Signal flow

signal to be encoded be $\underline{u}$. The signal is then encoded and rearranged into a 2D grid to obtain $x_{i,j}$. The signal $x_{i,j}$ then goes through the printer to obtain a continuous signal $x(s,t)$. This image may undergo some distortions. It is then scanned to obtain the discrete image $y_{i,j}$. This signal then undergoes skew correction and decoding to get an estimate of the encoded signal $\hat{x}$ and subsequently the estimate of the original message transmitted $\hat{u}$. The blocks will be described in the following subsections and chapters.

B. Interference model

The interference comes from both the scanner and the printer. While printing out the information on a piece of paper, the ink can spread over from one pixel to another, causing interference. Also, the scanner doesn't have infinite resolution and hence if the pixel size is smaller than the scanner resolution, then the information at one pixel will have components from the neighboring pixels as well. The above factors lead to inter-symbol interference. The actual interference can be estimated by sending known patterns through the channel and observing the output.

For simplicity, the channel is assumed to be finite. Therefore, let it be described by $h(i,j)$, where $i,j \in \mathbb{I}$. For example, a simple model might assume that that the ink from the printer spreads uniformly in every direction, and that the scanner interference is also symmetric. Hence we have a model such that $h(i,j) > 0$ for $-m < i,j < m$, where m is a positive integer. This gives us a finite state model of the channel. Then, the signal $r(i,j)$ obtained after ISI is given by

$$r(i,j) = \sum_{k,l} s(i-k, j-l)h(k,l)$$

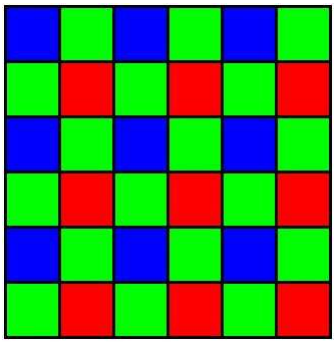


Fig. 3. Bayer pattern

C. Noise modeling

The primary source of noise in the channel is the noise from the scanner. A flatbed scanner consists of a glass pane called the platen which has a source of bright light under it, which illuminates the pane. The panel is closed to avoid ambient light, and a moving optical array of CCD sensors collects the light reflected by the image being scanned. The CCD sensors consist of the red, blue and green sensors that are arranged in the form of patterns, like the Bayer pattern as shown in Figure 3. The data is taken from these sensors via the input port of the computer to obtain the digital image.

The following are the potential noise sources:

1. Ambient noise in the form of light entering the scanning panel.
2. Optical noise while the sensors read the data.
3. A mismatch between the chromatic response and the original data, and the post

processing done by the printer and scanner can cause chromatic aberrations. The mapping of the R,G and B components of the image may not be linear which might lead to errors. These errors are called toning errors.

4. For binary data, the output is a black and white image. However, the sensors still read the data using the RGB sensors. When the image is scanned back and converted back to its original form, this may also lead to chromatic errors.
5. If the image is read into a PDA using a camera, the camera lens could introduce lensing effect. If the lens is very close to the image, then the refraction of light can cause the lens to create spherical distortions in the captured image.
6. Scanning back an image involves an analog to digital conversion. Depending on the resolution of the ADC, quantization noise is introduced.

It is difficult to model each of the above noise sources individually to come up with an exact model for the noise introduced by the channel. As a first approximation, we use an additive white Gaussian noise model (AWGN) with ISI.

D. Alternate models

The noise can be modeled in a few other ways. Consider the noise caused by the ambient light, it may be modeled as a rare event. Hence, the Poisson noise model would be a good model for the channel. It will also be a good model for any spikes that are introduced by the scanner. In this case, the noise n can be mathematically modeled as

$$\Pr(n(D) = k) = \frac{e^{-\lambda|D|}(\lambda|D|)^k}{k!}$$

Here D is the 2 dimensional region of the image over which the noise patterns occur. k is the number of occurrences of noise, and λ is the rate at which the noise occurs. λ

varies depending on the channel. $n(D)$ represents the number of occurrences of noise in the region D.

Another possible noise model could be the Laplace noise model. The noise n can be represented as

$$\Pr(n = k) = \frac{e^{-\frac{\sqrt{2}|k-\mu|}{\sigma}}}{\sqrt{2\sigma}}$$

Here μ is the mean of the distribution, which in this case can be set to 0. σ is the standard deviation of the noise, which is channel dependent. The best estimate for stationary Gaussian noise is its linear average. However, if the system is non linear, the Laplace noise may prove to be a better model because of its nonlinear density function.

An alternate way to model the noise and the interference is empirical estimation, that is, estimate it from the physical channel itself. Known data patterns can be sent over the channel. The difference between the original and the received signal $n = s - s'$ gives us the noise in the channel. A large number of data patterns can be collected and a histogram of the noise can be then plotted. By law of large numbers, the noise distribution hence obtained should be close to the actual noise distribution.

To estimate the interference pattern a few different methods have been tried. Since the channel can be modeled as an FIR filter, adaptive filter estimation could be used to get an estimate of the channel. It can also be modeled as least-square estimation problem. These techniques are described in detail in Chapter VI.

E. Model chosen for the project

For our simulations, we assume a symmetric 2×2 interference model given by

$$h = \begin{bmatrix} 1 & \beta \\ \beta & 0 \end{bmatrix}.$$

The value of β is chosen to be 0.5. The actual channel model may have a larger span but this model is chosen because of its low complexity, and it is a good model to give a proof of concept. The input is assumed to be independent and identically distributed chosen equiprobably from the alphabet $\{1, 0\}$. Let x_{ch} be signed obtained by passing x through the interference model of the channel. It is obtained as

$$x_{ch} = x * h$$

where $*$ represents 2D convolution. The corresponding noisy channel outputs are

$$x_{ch}(i,j) = \underbrace{x(i,j)}_{\text{signal}} + \underbrace{\beta x(i,j-1) + \beta x(i-1,j)}_{\text{interference}} + \underbrace{n(i,j)}_{\text{noise}}$$

The noise is assumed to be Gaussian with a variance σ^2. The Gaussian noise lies between $-\infty$ and ∞. However, a scanned image can have only positive values. Hence the Gaussian noise model needs to be modified for our system. The simplest modification is to use a clipped Gaussian model. The channel output after noise x' is obtained as follows. Let

$$\tilde{x} = x_{ch} + n$$

where n is Gaussian with mean 0 and variance σ^2. Next, let

$$l = min(x_{ch}), m = max(x_{ch})$$

Then,

$$x'(x,y) = \begin{cases} \tilde{x}(i,j) & \text{if } l \leq \tilde{x}(i,j) \leq m; \\ l & \text{if } \tilde{x}(i,j) < l; \\ m & \text{if } \tilde{x}(i,j) > m \end{cases}$$

x' is the input given to the detector for decoding. The signal constellation chosen for the system is $\{0, 1\}$. The matrix operations for obtaining the codeword from the information bits are done over $GF(2)$.

CHAPTER III

CODING FOR THE CHANNEL

A. Low-density parity-check (LDPC) codes

Low-density parity-check codes are a class of linear block codes that were introduced by Gallager in 1962 [1]. As the name suggests, an LDPC code is described by a sparse (low density) parity-check matrix. These codes were largely forgotten because of the high decoding complexity until recently when Mackay re-discovered them in 1996 [2].

LDPC codes are now becoming popular in practical applications including deep space communication and satellite transmission of digital television. They have also been proven to achieve capacity on the binary erasure channel and empirically appear to approach capacity over Gaussian and other channels. An LDPC code can be described as a bipartite graph, known as a Tanner graph. It consists of two sets of nodes, the variable nodes and check nodes. The parity-check matrix describes the edges connecting these nodes. The check nodes represent the parity check conditions. An example is shown in Figure 4. The circles represent the variable nodes, and the squares represent the parity check nodes. Each row of the parity-check matrix is represented by a check node, and each column is represented by a variable node. A non-zero entry in the column (i, j) of the matrix represents a connection between the i^{th} check node and the j^{th} variable node.

A common ensemble of LDPC codes can be described by the degree profiles of its variable and check nodes, which describe the fraction of edges connected to the node of particular type. On the basis of degree profiles, LDPC codes can be classified into

$$H = \begin{bmatrix} 1 & 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 0 & 1 & 1 \end{bmatrix}$$

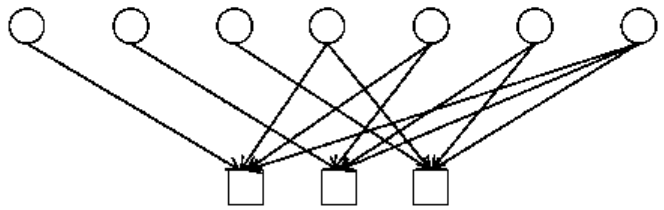


Fig. 4. Example of a Tanner graph

regular and irregular LDPC codes. A regular code is one in which the degree of all the variable and check nodes are same, that is each variable node is connected to the same number of edges and each check node is connected to the same number of edges. For example, in a regular $(3, 6)$ code, each variable node is connected to 3 edges and each check node is connected to 6 edges. Irregular codes have irregular degree profiles. The example shown in Figure 4 is an irregular code. An LDPC code is called a check-regular (or bit-regular) code if the check nodes (or bit nodes) have a regular degree.

B. Protographs and protograph based codes

A protograph is a Tanner graph with a smaller number of nodes. They were first introduced by Thorpe in [3]. They can be thought of as a small blueprint for a longer

LDPC code.

The process of developing an LDPC code can be described with the example below. Consider the protograph shown in the Figure 5(a). Depending on the length of the code, copies of the protograph are taken. At this point, there is no advantage of taking copies of the code since the code as a whole will perform exactly the same as an individual protograph. While keeping the underlying structure of the code, a random permutation of the edges is taken. This operation ensures the randomness in the code while having a regular structure to it. Long random ensembles have proven to be capacity achieving [1], and hence introducing randomness in long codes will improve the code properties.

In [3], Thorpe describes a way to design good protographs based on randomized search algorithm. The advantage of developing a code based on protographs is that the performance of the code can be analyzed by just analyzing the protograph, which reduces the complexity of the analysis because the number of nodes in the protograph is very small as compared to the actual tanner graph of the code. He also talks about how to analyze the protograph based ensembles.

C. LDPC convolutional codes and spatial coupling

LDPC convolutional codes were introduced in [4] and are shown to achieve capacity similar to LDPC block codes. Like the block codes counterpart, they can be defined by sparse parity-check matrices, and can be decoded using iterative decoding similar to the block codes. These codes are shown to be more practical than the block codes for many communication systems. They allow the use of arbitrary block lengths and frame sizes, and are practical in terms of encoding. For example, the encoding can

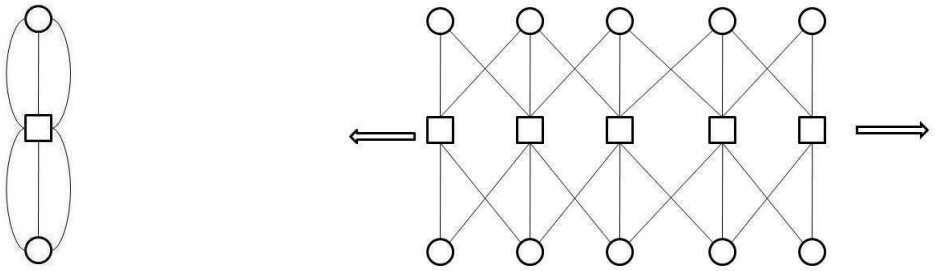
be implemented using shift registers in hardware, which gives a significant advantage over LDPC block codes. A comparison between LDPC block and convolutional codes can be found in [5], which shows that LDPC convolutional codes have many practical advantages.

Based on this concept, *Urbanke et.al.* proved in [6] that the threshold of convolutional-like or spatially coupled codes approach the MAP threshold over the binary erasure channel. This gives us a new approach to the design of capacity approaching ensembles.

D. Coding scheme for the channel

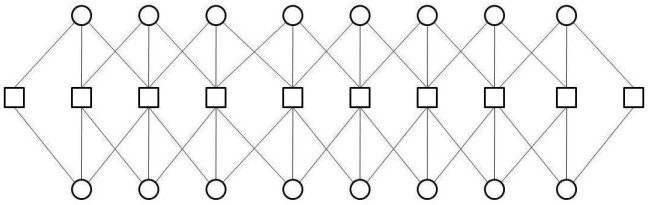
For our problem, we propose a code construction based on the spatially coupled code proposed by Urbanke in [6] and using it as a protograph for the construction of the parity-check matrix. The protograph of the code is shown in Figure 5(b). It can be constructed from a (3,6) regular LDPC code, the protograph of a regular (3,6) code is shown in Figure 5(a). Now we take multiple copies of this protograph and connect each copy to the copies to the left and right of it shown in Figure 5(c). The chain needs to be terminated for which an extra check node needs to be added to each end of the graph. This leads to a rate loss, which goes to 0 as the number of copies approaches infinity. It is proven in [6] that a slight variation of this code construction approaches the MAP threshold over the binary erasure channel with message passing decoding.

As part of the research, we attempt to find the threshold of this protograph over the Gaussian channel over message passing decoding.



(a) Protograph for a (3,6) LDPC code

(b) Spatially coupled version of the (3,6) LDPC protograph



(c) Truncated version of (b)

Fig. 5. LDPC code construction

E. Encoding for the proposed code

Let the number of bits to be encoded be l, and the rate of the code be r. Then the number of output bits $N = l/r$. Let $k = N - l$. The parity-check matrix for the code has the following structure:

$$H^{(k\times N)} = \begin{bmatrix} \tilde{I}_{1,1} & \tilde{I}_{1,2} & & & & & & & & & & & & & \\ \tilde{I}_{2,1} & \tilde{I}_{2,2} & \tilde{I}_{2,3} & \tilde{I}_{2,4} & & & & & & & & & & & \\ \tilde{I}_{3,1} & \tilde{I}_{3,2} & \tilde{I}_{3,3} & \tilde{I}_{3,4} & \tilde{I}_{3,5} & \tilde{I}_{3,6} & & & & & & & & & \\ & & \tilde{I}_{4,3} & \tilde{I}_{4,4} & \tilde{I}_{4,5} & \tilde{I}_{4,6} & \tilde{I}_{4,7} & \tilde{I}_{4,8} & & & & & & & \\ \vdots & \vdots & & & & & & & & & & & & \vdots & \\ & & & & & & & & & & \tilde{I}_{(k-1)(N-3)} & \tilde{I}_{(k-1)(N-2)} & \tilde{I}_{(k-1)(N-1)} & \tilde{I}_{(k-1)(N)} \\ & & & & & & & & & & & & \tilde{I}_{(k)(N-1)} & \tilde{I}_{(k)(N)} \end{bmatrix}$$

$\tilde{I}$ represents an identity matrix with rows randomly permuted. The size of this matrix is equal to the number of the copies of the protograph taken, let it be of size $q \times q$. In order to obtain the generator matrix G for the code from the parity-check matrix, we need to reduce it to a systematic form where $H = [I|P]$ where I is an identity matrix of size $k \times k$. The algorithm used for this uses row and column operations and is described below.

1. First rearrange the columns of H. Let the matrix thus obtained be H'. The first $n/2$ columns of H' are obtained by taking columns $1, 3, 5...N-1$ of H. The columns $N/2+1$ to N are obtained by taking columns $2, 4, 6, ...N$ of H. H' obtained looks as below.

$$H'^{(k\times N)} = \begin{bmatrix} \tilde{I}_{1,1} & & & & & & & \tilde{I}_{1,2} & & & & & & \\ \tilde{I}_{2,1} & \tilde{I}_{2,3} & & & & & & \tilde{I}_{2,2} & \tilde{I}_{2,4} & & & & & \\ \tilde{I}_{3,1} & \tilde{I}_{3,3} & \tilde{I}_{3,5} & \cdots & & & & \tilde{I}_{3,2} & \tilde{I}_{3,4} & \tilde{I}_{3,6} & \cdots & & & \\ & \tilde{I}_{4,3} & \tilde{I}_{4,5} & \cdots & & & & & \tilde{I}_{4,4} & \tilde{I}_{4,6} & \cdots & & & \\ & & \tilde{I}_{5,5} & & & & & & & \tilde{I}_{5,6} & & & & \\ & & & & \tilde{I}_{(k-3),(N-3)} & & & & & & & \tilde{I}_{(k-3),(N-2)} & & \\ & & & & \tilde{I}_{(k-2),(N-3)} & \tilde{I}_{(k-2),(N-1)} & & & & & & \tilde{I}_{(k-2),(N-2)} & \tilde{I}_{(k-2),N} & \\ & & & & \tilde{I}_{(k-1),(N-3)} & \tilde{I}_{(k-1),(N-1)} & & & & & & \tilde{I}_{(k-1),(N-2)} & \tilde{I}_{(k-1),N} & \\ & & & & & \tilde{I}_{k,(N-1)} & & & & & & & \tilde{I}_{k,N} & \end{bmatrix}$$

2. Next, we need to obtain an identity matrix in the first k columns. Consider the first column of H'. It is of the form $\begin{bmatrix} \tilde{I}_{1,1} \\ \tilde{I}_{2,1} \\ \tilde{I}_{3,1} \end{bmatrix}$.

   We need an identity matrix of size $q \times q$ in place of $\tilde{I}_{1,1}$. But $\tilde{I}$ is just a permutation of an identity matrix. Therefore, all we need to do is to permute the first q rows of H' to obtain an identity matrix in place of $\tilde{I}_{1,1}$.

3. Next, the rows below $\tilde{I}_{1,1}$ should be all 0. Since $\tilde{I}_{2,1}$ and $\tilde{I}_{3,1}$ are also permutations of identity matrix, we just need to subtract the appropriate rows in order to make it all zeros. This is explained in the following example.

   Let $q = 3$, hence $\tilde{I}_{m,n}$ is of size 3×3. Let

$$\begin{bmatrix} \tilde{I}_{1,1} \\ \tilde{I}_{2,1} \\ \tilde{I}_{3,1} \end{bmatrix} = \begin{bmatrix} 0\ 1\ 0 \\ 1\ 0\ 0 \\ 0\ 0\ 1 \\ \hline 1\ 0\ 0 \\ 0\ 0\ 1 \\ \hline 0\ 1\ 0 \\ \hline 1\ 0\ 0 \\ 0\ 1\ 0 \\ 0\ 0\ 1 \end{bmatrix}$$

   For step 2, we need to swap the first row with the second, then we have a 3×3 identity matrix in place of $\tilde{I}_{1,1}$. After doing this, we obtain

$$\begin{bmatrix} \tilde{I}'_{1,1} \\ \tilde{I}_{2,1} \\ \tilde{I}_{3,1} \end{bmatrix} = \begin{bmatrix} 1\ 0\ 0 \\ 0\ 1\ 0 \\ 0\ 0\ 1 \\ \hline 1\ 0\ 0 \\ 0\ 0\ 1 \\ \hline 0\ 1\ 0 \\ \hline 1\ 0\ 0 \\ 0\ 1\ 0 \\ 0\ 0\ 1 \end{bmatrix}$$

For step 3, we notice that the 1^{st} row is equal to the 4^{th} and 7^{th}, the 2^{nd} row is equal to the 6^{th} and the 8^{th} and the 3^{rd} row is equal to the 5^{th} and the 9^{th}. Therefore, we need to do the following operations,

$$\begin{aligned} row4 &= row4 - row1 \\ row7 &= row7 - row1 \\ row6 &= row6 - row2 \\ row8 &= row8 - row2 \\ row5 &= row5 - row3 \\ row9 &= row9 - row3. \end{aligned}$$

Hence we obtain

$$\begin{bmatrix} \tilde{I}'_{1,1} \\ \tilde{I}'_{2,1} \\ \tilde{I}'_{3,1} \end{bmatrix} = \begin{bmatrix} I^{3\times 3} \\ 0 \\ 0 \end{bmatrix}$$

4. Repeat steps 2 and 3 for columns 1 to $N/2$ of H'. In this way, we obtain an identity matrix of size $(k-2q)\times(k-2q)$ in the first $k-2q$ columns of H'.
5. For the next $2q$ columns, we use the Gaussian-elimination method to obtain and identity matrix of size $2q \times 2q$ in the sub-matrix formed by rows and columns $k-2q+1 tok$ to obtain $\tilde{H} = [I|P]$.
6. The generator matrix is obtained by $G' = [-P^T|I]$. Since all the operations are done over $GF(2)$ we have $-P^T = P^T$. Therefore, $G' = [P^T|I]$.
7. Undo the row swapping operations on the generator matrix G' to obtain the

generator matrix for the original parity-check matrix H as G.

The advantage of this algorithm over using Gaussian elimination is that the parity-check matrix H has a structure that gives us easier operations in order to obtain G. The complexity of the algorithm is linear in N.

F. Threshold evaluation of protograph-based codes for the Gaussian channel

Protograph-based codes offer a significant advantage over random codes because they allow irregular elements in the ensemble but provide deterministic local neighborhoods. The first significant property of an error correcting code that we need to evaluate is its threshold. The threshold of a code can be defined as the signal-to-noise ratio (SNR) above which reliable decoding can be achieved over a channel if the signal is encoded by the given code with block length tending to infinity, and below that SNR, decoding cannot be achieved.

For this project, we evaluate the threshold of the proposed code over Gaussian approximation [7]. Threshold evaluation is done as follows:

1. The protograph of the proposed code is considered. The codeword used for evaluation is the all zero codeword.
2. Let the noise power be σ^2. Then initialize the pdf of all bits received to be a Gaussian pdf with mean 0 and variance σ^2.
3. Variable node operation: Multiply all the received pdfs, and approximate the result with a Gaussian pdf with mean μ and variance σ'^2. This pdf is then passed onto the check nodes.
4. Check node operation: The pdf obtained from the variable nodes are convolved

at the check nodes. Since all the pdfs are Gaussian, that would just result into adding the means and variances of the incoming pdfs. Carry on the check node and variable node operations for a fixed number of iterations.

5. If the output converges for the given SNR, repeat steps 2 and 3 for a lower SNR until the output doesn't converge. The lowest SNR for which the output converges is the threshold of the code.

The different protographs and their calculated thresholds tried out as a part of this work are listed in Table I. The structure for all the protographs is the design proposed before in this subsection. The number of iterations is fixed to 600.

Table I. Thresholds of spatially coupled protographs

Protograph	Rate	Threshold E_b/N_0 in dB
16 var nodes-10 check nodes	0.375	1.174
20 var nodes-12 check nodes	0.4	1.15
28 var nodes-16 check nodes	0.4286	1.07
32 var nodes-18 check nodes	0.4375	1.02
44 var nodes-24 check nodes	0.4545	0.89
64 var nodes-34 check nodes	0.4688	0.78

G. Message passing decoding of LDPC codes

This subsection describes the decoding technique used for the error correcting codes. For optimal (MAP/ML) decoding of the LDPC code, we need to find the codeword

c for which $\Pr(x''|c) \geq \Pr(x''|c')$ for all other codewords c'. There are 2^k codewords in total, where k is the number of bits being encoded. Generally, 2^k is an extremely large number and hence it is not possible to compute the above probability for all codewords.

Therefore, sub-optimal techniques for decoding of the LDPC code are used. The most popular technique is the iterative message-passing decoding. The parity-check matrix of the LDPC code can be represented by a bipartite graph, with two sets of nodes, the variable nodes and the check nodes. The algorithm consists of a set of messages being passed over the edges to the check nodes and the variable nodes. These messages are passed iteratively from one set of nodes to the other in order to propagate information from all check nodes to all variable nodes and vice versa, which is then utilized to make decisions over all the bits.

This is a soft-input soft-output decoder. The soft information about a bit is in the form of log likelihood ratio of the bit. For example, $I_m = \log_2\left(\frac{\Pr(x_m=0)}{\Pr(x_m=1)}\right)$.

The algorithm is described with the help of an example below. The channel input comes into the variable nodes as shown. Let the input to the m^{th} node be denoted by I_m Let the input coming in from the check nodes to the variable node m in the j^{th} iteration on the i^{th} edge be $\overleftarrow{I}_{mi}^{(j)}$. Let the degree of the node be k. Then the outgoing message on edge q is given by

$$\overrightarrow{I}_{mq}^{(j+1)} = I_m + \sum_{i=1, i\neq q}^{k} \overleftarrow{I}_{mi}^{(j)}$$

Consider a check node of degree p. Let the input coming in from the variable nodes to the variable node n in the $(j+1)^{th}$ iteration on the i^{th} edge be $\overrightarrow{I}_{ni}^{(j+1)}$. Then the

outgoing message on edge q is given by

$$\overleftarrow{I}_{mq}^{(j+1)} = 2 \text{ atanh} \left(\prod_{i=1, i \neq q}^{k} \tanh \left(\frac{\overrightarrow{I}_{mi}^{(j+1)}}{2} \right) \right)$$

In order to decode the code the code, the algorithm is repeated for some iterations J. The final soft output for the m^{th} variable node is given by

$$\hat{x}_m = I_m + \sum_{i=1}^{k} \overleftarrow{I}_{mi}^{(J)}$$

Turbo decoding is used to improve the performance of the system.

CHAPTER IV

IMAGE RESTORATION

A. Assumptions

For our research, we assume that:

1. The noise an interference model is as described in Chapter II.
2. The image obtained is free of any toning or lensing errors.
3. The image obtained is not scaled during the process of transmission of the information.

B. Rotation of the image

When the image is being scanned, using a scanner or a camera, the obtained image may not be aligned because of human errors and we may obtain a rotated version of the desired image. This error needs to be corrected before giving the channel input obtained to the detector and the decoder.

Let the image printed out on paper be $I(x, y)$ which is a continuous signal. Let $0 < x, y < m$, where $m \in \mathbb{R}^+$ and $x, y \in \mathbb{R}$. Rotation of the image by an angle θ gives us $I(x\cos\theta - y\sin\theta, x\sin\theta + y\cos\theta)$. Figure 6 shows an example of rotated grid points. The discrete model of the system $\tilde{I}(\hat{x}, \hat{y})$ can be obtained by sampling. The discrete model can be obtained as

$$\tilde{I}(\hat{x}, \hat{y}) = \int\int_S I(x\cos\theta - y\sin\theta, x\sin\theta + y\cos\theta)dxdy$$

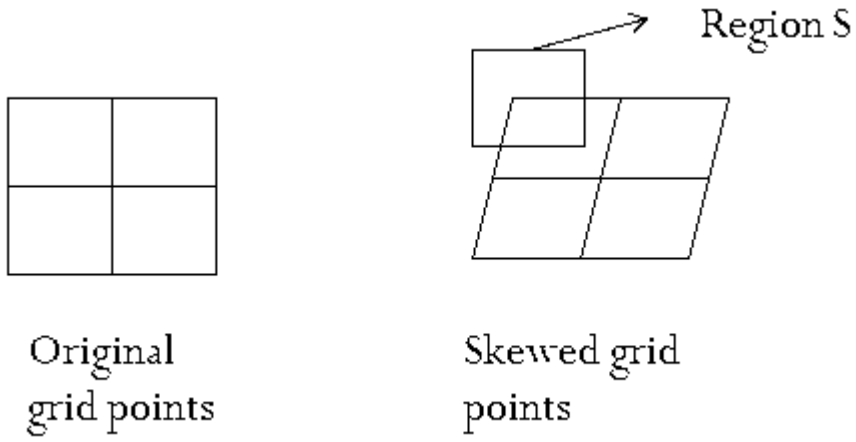


Fig. 6. Example of rotation of grid points

where the region S is as shown in Figure 6. The region S depends on the size of the CCD sensor being used by a scanner. In the process of discretization, the scanner would pick up information from the neighboring pixels as well, hence contributing to the ISI.

The interpolation can be modeled in multiple ways. Let the original pixel points lie on a square grid. Let the point at which data needs to be obtained be $R = (x', y')$. Let the grid points closest to (x', y') be $R_1 = (x_1, y_1), R_2 = (x_1, y_2), R_3 = (x_2, y_1)$ and $R_4 = (x_2, y_2)$.

The first interpolation method is nearest neighbor interpolation. This is a relatively easy model. Let d_i be the distance of R from R_i. Then,

$$\tilde{I}(x', y') = I(R_k) \text{ where } k = \underset{j \in \{1,2,3,4\}}{\operatorname{argmin}} (d_j)$$

The next method is bilinear interpolation. Here, the interpolated value depends on the 4 closest values and is proportional to the distance from these pixels. We can compute the interpolated values as follows:

$$\begin{aligned}\tilde{I}(x',y') &= \frac{I(x_1,y_1)}{(x_2-x_1)(y_2-y_1)}(x_2-x')(y_2-y') \\ &+\frac{I(x_1,y_2)}{(x_2-x_1)(y_2-y_1)}(x_2-x')(y'-y_1) \\ &+\frac{I(x_2,y_1)}{(x_2-x_1)(y_2-y_1)}(x'-x_1)(y_2-y') \\ &+\frac{I(x_2,y_2)}{(x_2-x_1)(y_2-y_1)}(x'-x_1)(y'-y_1)\end{aligned}$$

Higher order interpolation methods can also be used, like bicubic interpolation and higher order spline interpolation but they increase the complexity of the system. Due to non-linearity, they may also lead to higher system noise.

For our system, we assume that the scanner doesn't have infinite resolution and uses bilinear interpolation for calculating the pixel values.

C. Proposed method for de-rotation

An example of the rotated image is shown in Figure 7. In order to determine the angle of rotation, the first step is to determine the location of the information on the image. This could be done by embedding a known pattern in the image. However, to determine the angle of rotation, there have to be a series of patterns. For our system, we chose to embed a pattern on the four corners of the image. In order to determine the angle of rotation the following algorithm is used. Let $a(x,y)$ be the pattern embedded, with $0 < x, y < p$ where p is a positive integer.

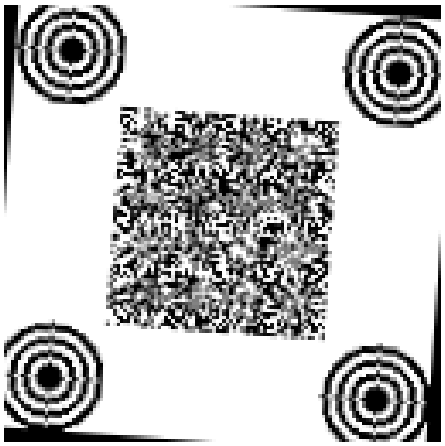


Fig. 7. Received rotated image

1. The obtained channel input is x'. A matched filter is used to determine the positions of the patterns in the image. The discrete filter output F is obtained as the discrete 2D convolution x' and a, represented by $F = x' * a$.
2. To determine the coordinates of the positions of the embedded patterns, the output of the matched filter F is divided into 4 squares as shown in the Figure 8. Then the positions are determined as $(x1, y1) = \text{argmax } A_1(x, y), (x2, y2) = \text{argmax } A_2(x, y), (x3, y3) = \text{argmax } A_3(x, y), (x4, y4) = \text{argmax } A_4(x, y)$
3. The angle of rotation θ is determined as follows:

$$\begin{aligned} \theta_1 &= \arctan\left(\frac{y_2 - y_1}{x_2 - x_1}\right) \\ \theta_2 &= \arctan\left(\frac{y_4 - y_3}{x_4 - x_3}\right) \\ \theta &= \frac{\theta_1 + \theta_2}{2} \end{aligned}$$

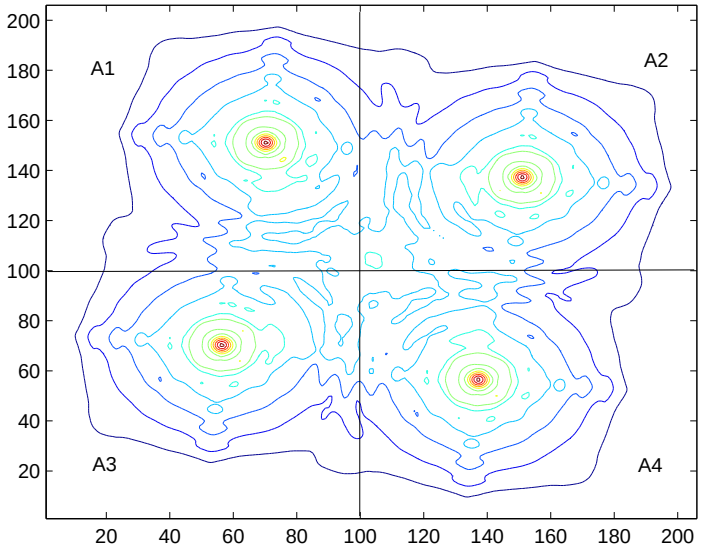


Fig. 8. Contour of the matched filter output

For the system to work, the detection and decoding algorithms just need to be robust enough to handle the noise introduced by this inaccuracy.

Since the angle of rotation can be arbitrary, it makes sense to choose a pattern that is rotationally invariant. The geometric shape that is rotationally invariant is a circle. In order to make the algorithm more robust, the pattern embedded in the image looks like a bull's eye, as shown in Figure 9(a). The size of the embedded pattern used is 32×32, and they are embedded in the image as shown in Figure 9(b). The autocorrelation of a solid circle with itself gives a smooth output which will have a peak, as shown in Figure 10(a). The noise in the system may give rise to a false peak. The autocorrelation of the pattern with itself is shown in Figure 10(b), and it has alternating peaks and lows, which makes the angle determination more robust to noise.

In order to de-rotate the image, bilinear interpolation is used, as described above, to obtain the de-rotated estimate of the signal. The embedded patterns are removed from this signal to get x'', which serves as the input to the channel detector.

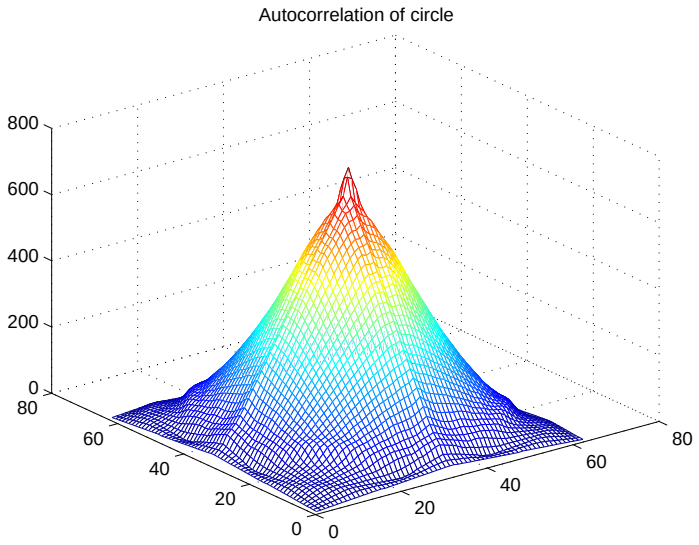
(a) Pattern used (b) Positioning of patterns on the grid

Fig. 9. Patterns embedded for de-rotation

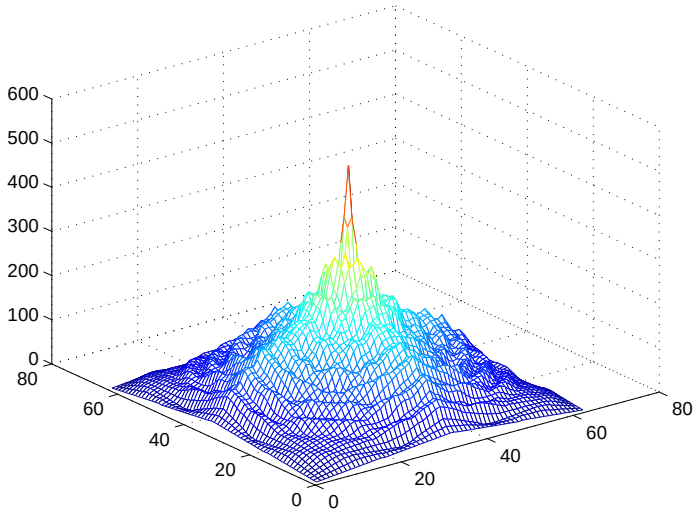
D. Modeling the interpolation noise

In addition to the noise from the scanner, interpolation also adds noise to the signal. In order to improve the performance of the system, this noise needs to be modeled. One method to model the noise is by gathering empirical data. Let the original image be P, and let the image obtained after rotation and de-rotation by an angle θ be $\tilde{P}$. We assume that $-20 \leq \theta \leq 20$. A random pattern of data is generated is rotated and de-rotated using bilinear interpolation. Then, the noise introduced is given by $n = P - \tilde{P}$. This experiment is repeated a large number of times to obtain the distribution of noise.

The observed noise is shown in Figure 11. The best fit obtained was with AWGN of



(a) Autocorrelation of circle



(b) Autocorrelation of proposed pattern

Fig. 10. Matched filter output of embedded patterns

variance 0.38. The probability density function of a normal random variable with the given variance is also shown in the same figure; notice that the noise models match quite well.

The noise models for nearest neighbor and bicubic interpolation are shown in Figure 12 and Figure 13 respectively. The nearest neighbor interpolation has a discrete noise model, this is because the only possibility of an error is when the nearest neighbor of the point of interpolation is not the actual data point itself, in this case it will take on the value of another neighboring pixel, which will lead to an error. With bicubic interpolation, the channel could be modeled as an additive Laplace noise channel, with a smaller variance. Now assuming that the noise introduced by the channel is Gaussian, the noise model in this case is given by the convolution of a Laplacian with a Gaussian. This makes the model more complex. Therefore, for simplicity, we use the bilinear interpolation model.

E. Transmit and receive filters

In a typical communication scenario, modulation is defined as the process of mapping a set of bits onto continuous waveforms. In our system, modulation means mapping a bit (or a set of bits) to pixels that are printed on paper. For our system simulations, we started with a one-to-one bit to pixel mapping, therefore the modulator used is an impulse, which translates to no modulation. Later on, we also mapped 1 pixel to a block of $x \times x$ pixels, where x could be from 1 to 4. Let G be the modulation filter used where

$$G(i,j) = \begin{cases} 1, \text{for } 1 \leq x \\ 1, \text{otherwise} \end{cases}$$

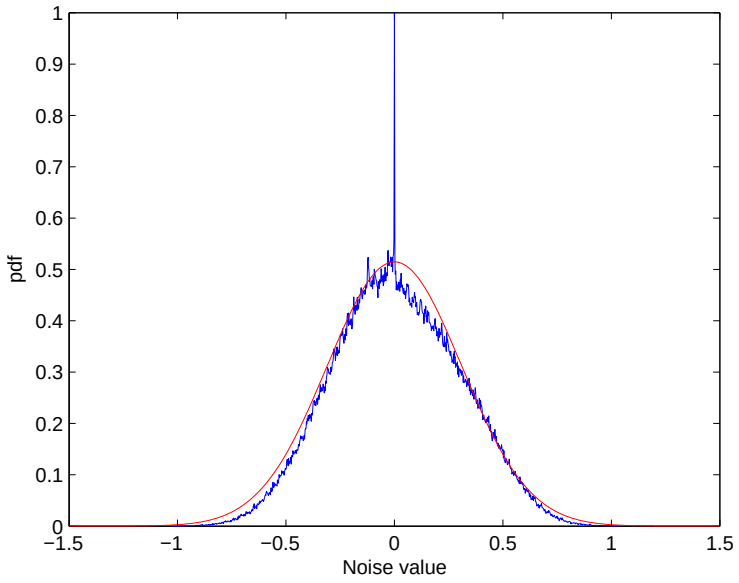


Fig. 11. Estimation of rotation noise using bilinear interpolation

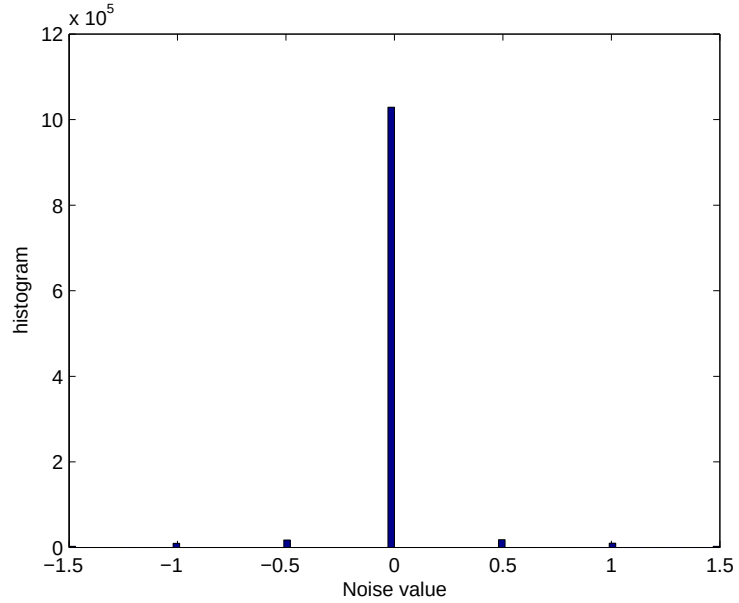


Fig. 12. Estimation of rotation noise using nearest neighbor interpolation

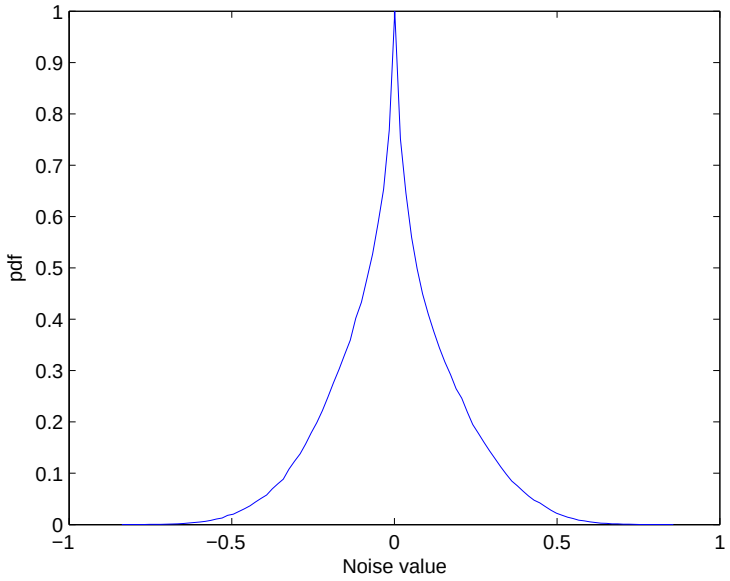


Fig. 13. Estimation of rotation noise using bicubic interpolation

This filter just maps one bit to a square of kxk pixels. Hence, a signal of size mxm bits gets mapped to a signal of kmxkm pixels. Let the scanner image after restoration be $\hat{T}$, the optimal receiving filter for the system is a filter match to the modulator. Therefore, the demodulated output is obtained as $\hat{I}(i,j) = \hat{T} * G$, where $i, j \in \mathbb{I}, 0 < i, j < m$ and sample the output at rate x in both dimensions. This is the optimal detector for the system.

F. Channel detection

Channel detection refers to the process of detection of bits in the presence of interference. Optimal detection of 1-D signals can be done using the Viterbi equalization.

Consider a channel with memory of ν bits. The optimal detector for this channel can be implemented using a trellis with 2^ν states. Consider the output from a 1-D channel with a memory of ν bits $x_1, x_2, \cdots, x_n$. Then $x_{j+\nu+1}, x_{j+\nu+2}, \cdots, x_n$ are independent of $x_1, x_2, \cdots, x_j$ given $x_{j+1}, x_{j+2}, \cdots, x_{j+\nu}$. Hence, even if the length of the 1D signal $N \to \infty$, it can be detected optimally using a trellis with 2^ν states. Therefore, the complexity of optimal 1-D detection depends only on the memory of the channel and not the length of the signal itself. However, this is not the case with 2-D signals. Consider the block to be detected in Figure 14. For optimal detection of the undecoded signal, we need to condition on the bounding strip as shown. Now when the size of the block approaches infinity, the size of this strip becomes infinite, hence the complexity of optimal detection goes to infinity as $N \to \infty$.

The maximum-likelihood equalizer for the 1D and 2D channels can be implemented using the Viterbi algorithm. For a 2D channel, the complexity of this algorithm is quite high. Therefore, sub-optimal methods are used for detection of 2-D signals. Our approach is largely based on the work done by Marrow in [8] Two of the techniques proposed therein are described in the subsection below.

1. Turbo product equalization (TPE)

A finite-complexity 2-D detection scheme can be constructed using two finite-complexity soft-input soft-output (SISO) detectors, one operating on the rows and the other on the columns. The individual 1-D detectors treat the ISI from the other dimension as noise. Because of this the error rate for individual 1-D detectors are really bad. To improve the performance, the extrinsic information from the 1-D detectors from one dimension is passed on to the detector in the other dimension. The equalizer so

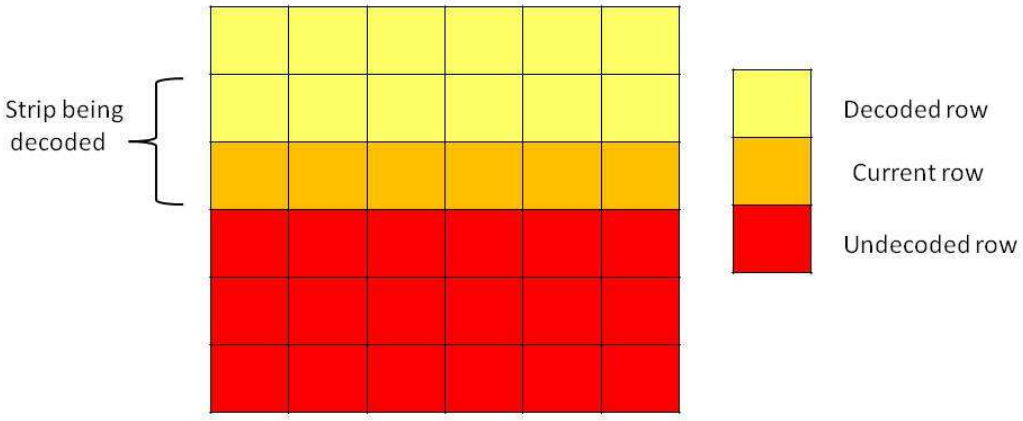


Fig. 14. Signal block for equalization

designed resembles the turbo decoder.

For this project, the 1-D decoder chosen was the BCJR algorithm. In the first iteration, the log likelihood ratio for each bit is calculated by taking one row at a time and using the BCJR algorithm on it. So,

$$I^{(1)}(x,y) = \log_2 \frac{\Pr(c(x,y)|a(x,y)=0)}{\Pr(c(x,y)|a(x,y)=1)}$$

The extrinsic information passed to the second iteration is

$$I^{(1)}_{ext}(x,y) = I^{(1)}(x,y) - L_{ap}(x,y)$$

where $L_{ap}(x,y)$ is the apriori probability of the bit $a(x,y)$. For the second iteration, the columns are taken one by one and the BCJR algorithm is used to obtain the log likelihood ratios in the following way

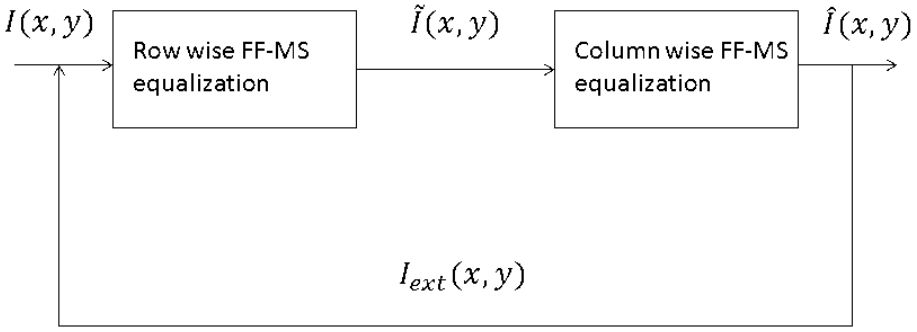
$$I^{(2)}(x,y) = \log_2 \frac{\Pr(c(x,y)|a(x,y)=0, I^1_{ext}(x,y))}{\Pr(c(x,y)|a(x,y)=1, I^1_{ext}(x,y))}$$

The process is continued for a fixed number of iterations.

2. Feed-forward multi-strip equalization (FF-MS)

The next technique used for this project is called feed-forward multi-strip equalization. The schematic diagram for the scheme is shown in Figure 15. The detection algorithm works as follows:

1. The first two rows of the signal block to be detected are taken. Let the rows be denoted by x''_1 and x''_2. The BCJR algorithm, with a 4 state trellis is used to get the soft output $I^{(1)}_2$ for the second row, which is marginalized over the bits of the first row.



$I(x,y)$ = Soft input from the channel

$\tilde{I}(x,y)$ = Soft output from row-wise FF-MS

$\hat{I}(x,y)$ = Soft output after row wise and column wise FF-MS

$I_{ext}(x,y)$ = Extrinsic information fed back = $\hat{I}(x,y) - I(x,y)$

Fig. 15. Block diagram of FF-MS equalizer

2. Next, x_2'' and x_3'' are decoded using the $I_2^{(1)}$ obtained in the first step to get $I_3^{(1)}$. In this way, information is propagated from one row to the next.
3. The above is repeated for all rows to get the soft output for all bits $I^{(1)}$.
4. The same procedure is now repeated by taking columns of x'' and decoding in the same way as the rows were decoded, to obtain $I^{(2)}$.
5. The decoding is done for rows and columns alternately for n iterations to get the final soft output of the detector $I^{(n)}$.

$I^{(n)}$ then becomes the input to the decoder for the LDPC code. More complex models can be used by taking more than 2 strips at a time for decoding, but the complexity of the detector increases.

G. Comparison of FF-MS and turbo product equalization

From the simulation results obtained, the FF-MS equalizer gives lower error rates as compared to the TPE. The results are shown in Figure 16. In the TPE, while decoding in one dimension, the interference from the other dimension is treated as noise, which leads to higher error rates as compared to the FF-MS, where both the dimensions are taken into consideration during symbol detection. FF-MS uses a more accurate trellis for detection as compared to the TPE, hence giving lower error rates.

H. BCJR algorithm for the proposed noise model

The BCJR algorithm was proposed by Bahl, Cocke, Jelinek, and Raviv in [9]. In this subsection, we talk about the outline of the BCJR algorithm and its implementation for our noise model. The detailed proof of the algorithm can be found in [9].

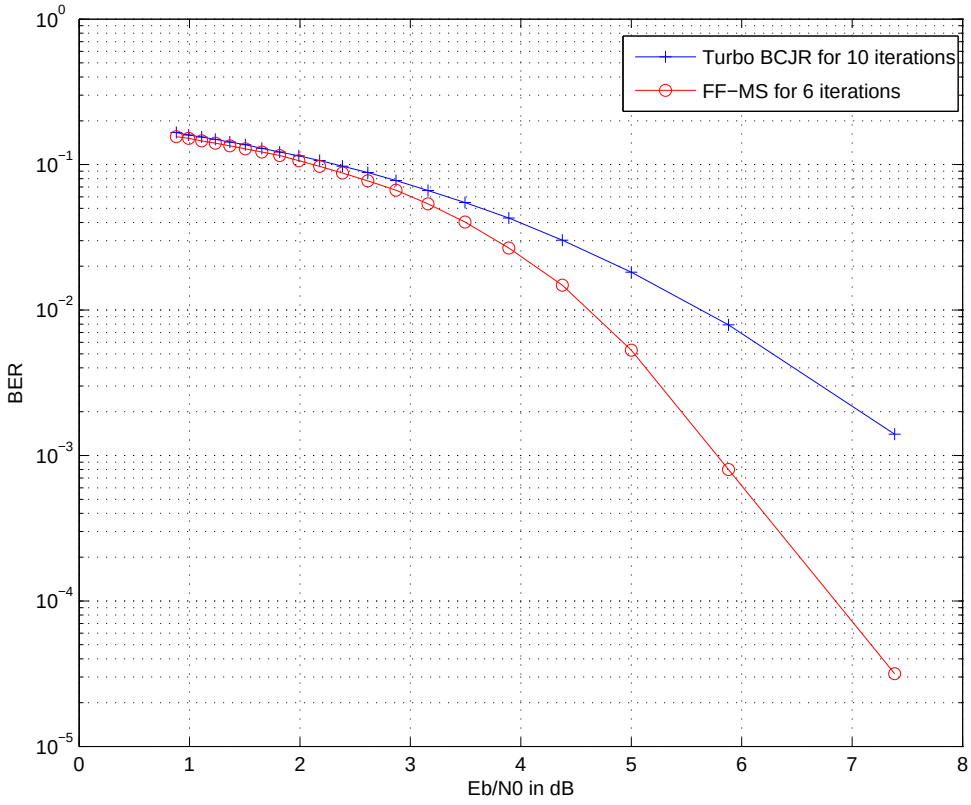


Fig. 16. Performance of TPE and FF-MS over the Gaussian channel

The BCJR algorithm is a soft-output algorithm that can be used with any Markov chain to produce the above aposteriori probabilities in a computationally efficient way. Consider a channel detector that used an n-stage trellis for symbol detection. Let us look at the k^{th} stage in the trellis. Let s_k represent the state of the trellis at stage k, and let u_k be the symbol that we are trying to detect at the k^{th} stage. Let us define the log likelihood ratio as

$$\Lambda[u_k] = \log \frac{\sum_{m',m:u_k=0} P(s_{k-1} = m', s_k = m, y_1^N)}{\sum_{m',m:u_k=1} P(s_{k-1} = m', s_k = m, y_1^N)}$$

where $\sum_{m',m:u_k=0}$ represents the sum over all m and m' for which the state transition occurs when the input symbol $u_k = 0$.

Now define

$$\begin{aligned} \alpha_k(m) &= P(s_k = m, y_1^k) \\ \beta_k(m) &= P(y_{k+1}^N | s_k = m) \\ \gamma_k(m', m) &= P(s_k = m, y_k | s_{k-1} = m') \end{aligned}$$

The BCJR algorithm describes how to compute α_k and β_k efficiently using recursions. The results from [13] are stated below.

$$\begin{aligned} \alpha_k(m) &= \sum_{m'} \alpha_{k-1}(m')\gamma_k(m', m) \\ \beta_k(m) &= \sum_{m'} \beta_{k+1}(m')\gamma_k(m, m') \\ \gamma_k(i, m', m) &= P(s_k = m, y_k = 0 | s_{k-1} = m') \end{aligned}$$

The log-likelihood ratio for the symbol u_k can be then calculated as

$$\Lambda[u_k] = \log \frac{\sum_{m',m} \alpha_{k-1}(m')\gamma_k(0,m',m)\beta_k(m)}{\sum_{m',m} \alpha_{k-1}(m')\gamma_k(1,m',m)\beta_k(m)}$$

For this algorithm, the only this that is dependent on the noise model is the calculation of γ_k. According to the definition, $\gamma_k(m',m)$ is the probability of transition from state m' in stage $k-1$ to state m in stage k given y_k was received. Consider the channel

$$h = \begin{bmatrix} 1 & \beta \\ \beta & 0 \end{bmatrix},$$

and the input alphabet $1, 0$. Let $v = u * h$. Since the model is a clipped Gaussian, the calculation of γ_k is not straight forward. Let $p = \min(v), q = \max(v)$ and let σ^2 be the Gaussian noise variance. Then, γ_k is calculated as follows:

$$\gamma_k(m',m) = \begin{cases} \frac{1}{\sqrt{2\pi\sigma^2}} e^{\frac{-(y_k - v_{m'\to m})^2}{2\sigma^2}} & \text{if } p < y_k < q; \\ Q\left(\frac{v_{m'\to m} - p}{\sigma}\right) & \text{if } y_k = p; \\ Q\left(\frac{q - v_{m'\to m}}{\sigma}\right) & \text{if } y_k = q \end{cases}$$

where $v_{m'\to m}$ is the output when there is a transition from state m' to m, and $Q(x) = \frac{1}{\sqrt{2\pi}} \int_x^\infty \exp\left(-\frac{u^2}{2}\right) = P(n_k > x)$.

The proof is as follows. Let n_k be the Gaussian noise before clipping.

1. In the first case, there is no clipping. Hence,

$$\begin{aligned} \gamma_k(m',m) &= P(y_k = v_{m'\to m}) \\ &= P(n_k = y_k - v_{m'\to m}) \\ &= \frac{1}{\sqrt{2\pi\sigma^2}} e^{\frac{-(y_k - v_{m'\to m})^2}{2\sigma^2}} \end{aligned}$$

2. In the second case, $y_k = p$,

$$\begin{aligned}\gamma_k(m', m) &= P(y_k = v_{m' \to m}) \\ &= P(n_k \leq y_k - v_{m' \to m})\end{aligned}$$

Since the value of y_k cannot be anything lesser than p, that means that the noise could have been anything lesser than $y_k - v_{m' \to m}$.

$$\begin{aligned}\gamma_k(m', m) &= P(n_k \leq y_k - v_{m' \to m}) \\ &= 1 - P(n_k > y_k - v_{m' \to m}) \\ &= 1 - Q\left(\frac{y_k - v_{m' \to m}}{\sigma}\right) \\ &= Q\left(\frac{v_{m' \to m} - p}{\sigma}\right)\end{aligned}$$

since $Q(x) = 1 - Q(-x)$.

3. In the third case, $y_k = q$,

$$\begin{aligned}\gamma_k(m', m) &= P(y_k = v_{m' \to m}) \\ &= P(n_k > y_k - v_{m' \to m})\end{aligned}$$

Since the value of y_k cannot be anything greater than q, that means that the noise could have been anything greater than $y_k - v_{m' \to m}$.

$$\begin{aligned}\gamma_k(m', m) &= P(n_k > y_k - v_{m' \to m}) \\ &= Q\left(\frac{q - v_{m' \to m}}{\sigma}\right)\end{aligned}$$

The BCJR algorithm is then implemented using the γ calculated as above.

CHAPTER V

SYSTEM PERFORMANCE AND COMPARISON

A. Simulation results for proposed LDPC code

The different simulation results obtained for the system and for the clipped Gaussian channel are shown in Figures 17 through 21.

B. Comparison and conclusion

From the simulation results obtained, it can be observed that the repeat accumulate code of the same rate performs better than the proposed protograph based LDPC codes over the Gaussian channel. The results of the simulations of the entire system show that the system that follows the given assumptions within certain limits can be used for transmission of information. For the given system also, the repeat-accumulate codes work better than the proposed spatially coupled codes.

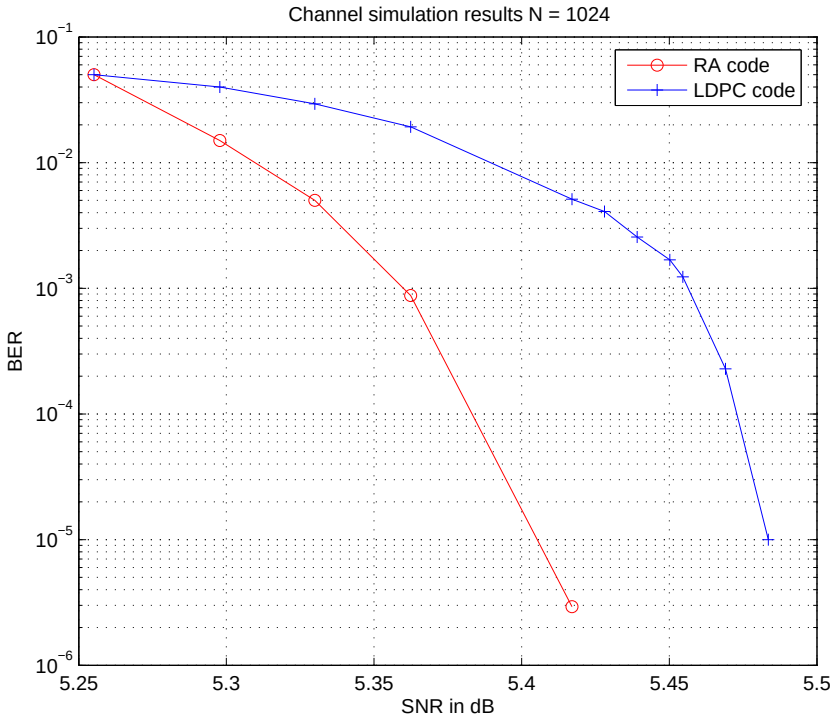


Fig. 17. Simulation results for the system N=1024

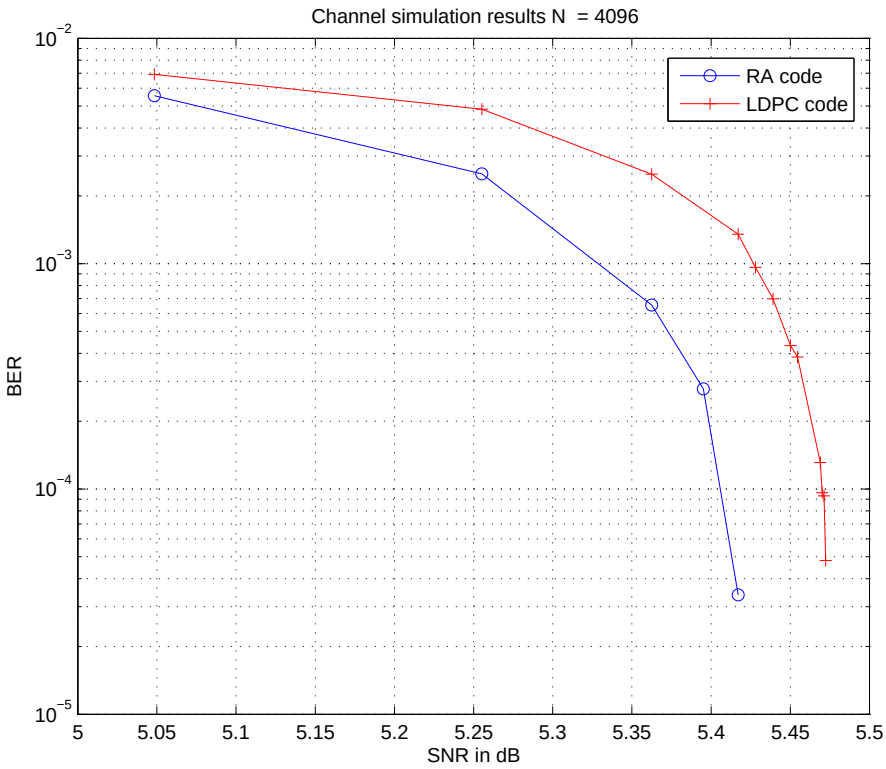


Fig. 18. Simulation results for the system N=4096

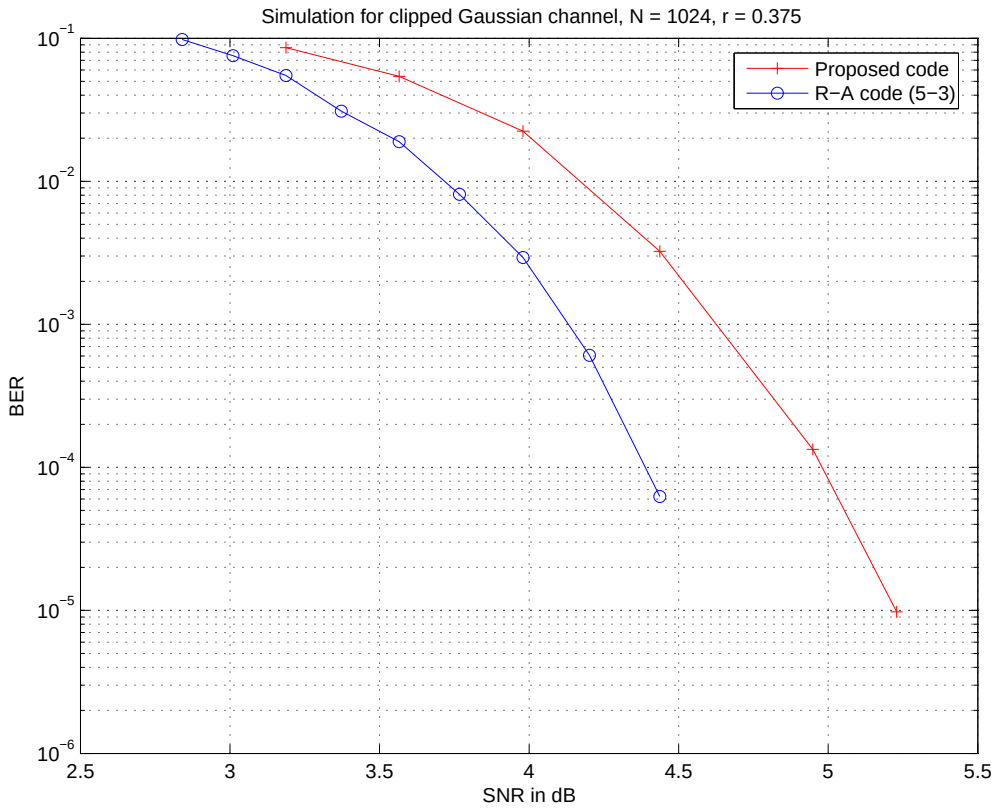


Fig. 19. Simulation results for the clipped Gaussian channel, N=1024

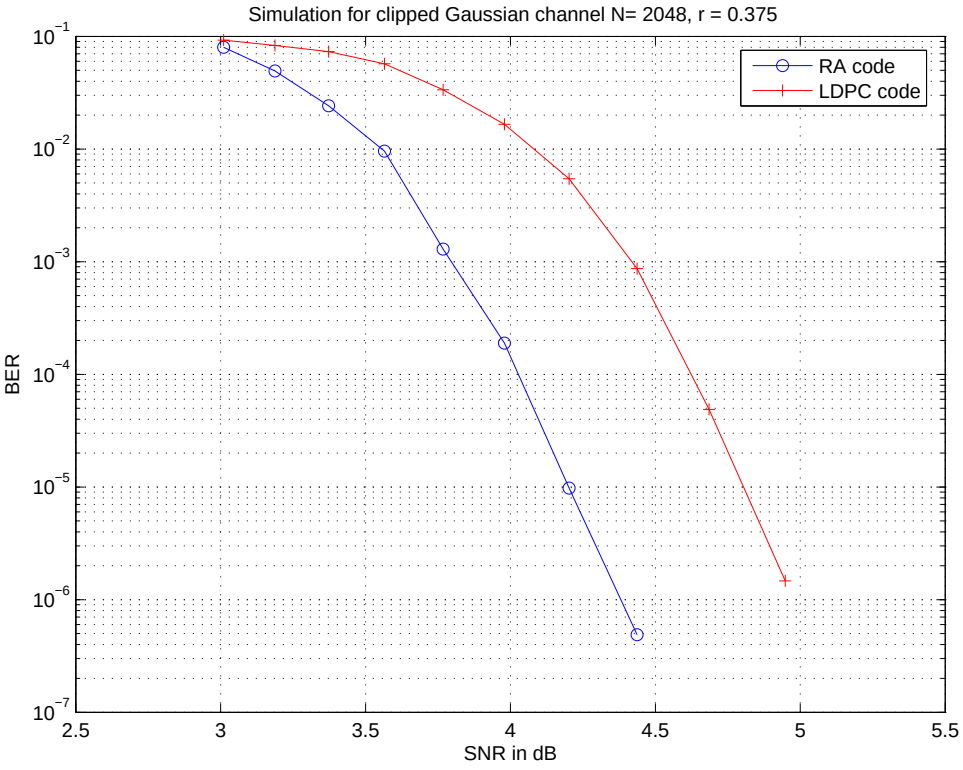


Fig. 20. Simulation results for the clipped Gaussian channel N=2048

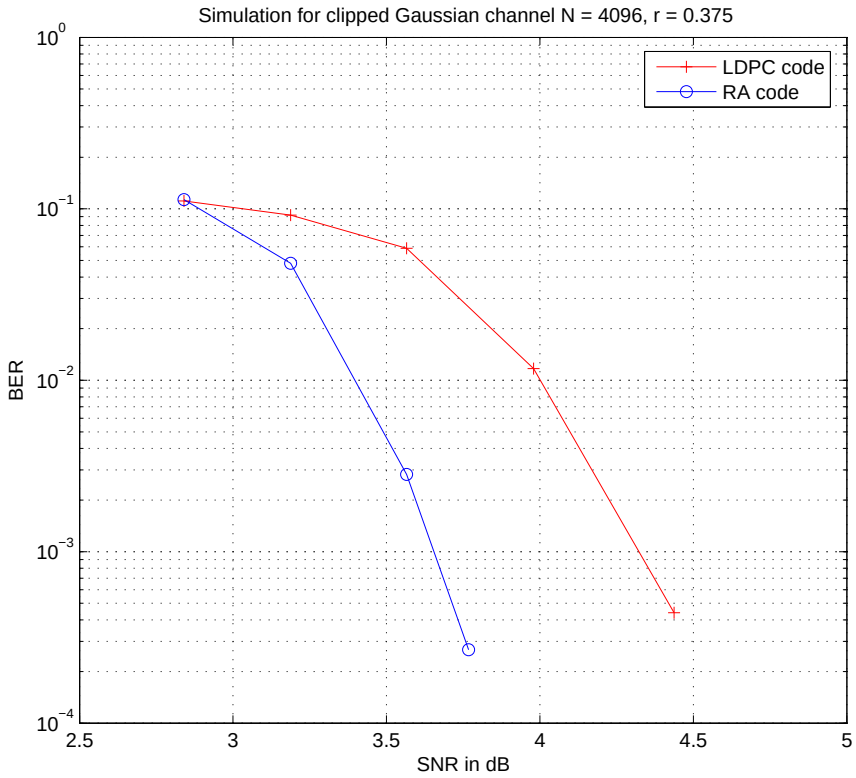


Fig. 21. Simulation results for the clipped Gaussian channel N=4096

CHAPTER VI

EXPERIMENTS CONDUCTED USING THE PHYSICAL SYSTEM AND CONCLUSIONS

We conducted some experiments using a physical printer and scanner system with the intent of finding out an approximate value for the capacity of that channel. The printer chosen was Xerox Phaser 550 printer and the scanner chosen was Cannon CanoScan LiDE200, which is a flatbed scanner. The subsections below describe the experiments conducted.

A. Preparing the data to print

The first step in the process was to take the data to be communicated over the channel. For our case, we used a 9 second long mp3 file with an 8 bit resolution and a rate of 35kbps. Including the header data, the file was 41.435 kb in size. From the simulations results, the capacity of the channel (including just the image distortions) is just less than 0.4. Therefore, the rate used for encoding the information is 0.375. The protograph used for encoding is the proposed spatially coupled protograph with length 16. The block length used for encoding is $N = 1024$, hence, the information bits are divided into blocks of $k = 386$ each. They are then encoded to obtain blocks of length 1024, which are reshaped into squares of size 32×32. Then, 900 of these squares are taken, and arranged into a 2-D image of size 30×30 of these blocks. In order to facilitate equalization of individual blocks, 2 bit-wide strips of blank spaces separate each block.

As discussed in Chapter IV, in order to assess the distortion in the image, we need to embed known patterns in it. The pattern suggested in subection IV.C is embedded around the four corners of the image. However, the results of the practical experiments done led to the conclusion that more data points were required to assess the distortion. Figures 22 and 23 show the data and the different pattern positions used for the experiments.

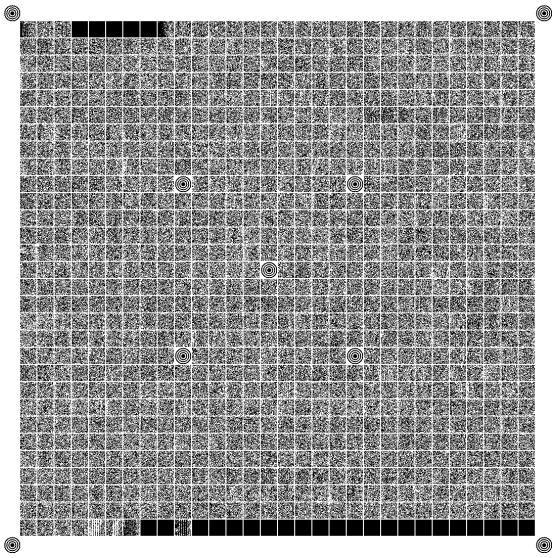
B. Printing the data

The data so prepared is then printed using the Xerox Phaser 550 printer. In order to empirically evaluate a lower limit for the information capacity of paper, we can print the data at different rates on paper and try to decode it. If the data can be successfully decoded, then by Shannon's theorem for information rate, we can say that the capacity of the channel is greater than the rate at which the data is printed.

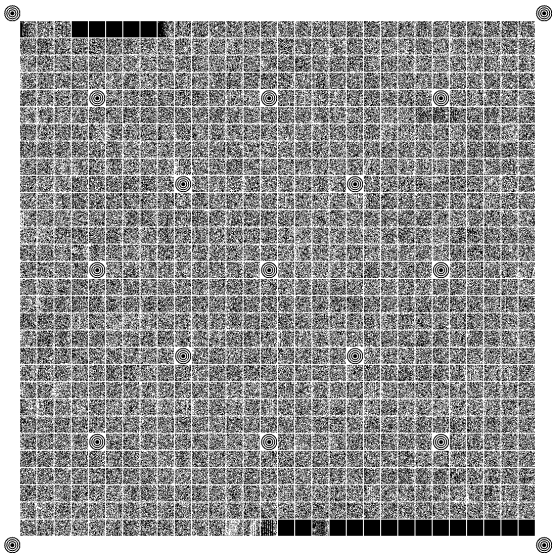
The resolution of a printer is defined by the dots per pixel (dpi) parameter of the printer. The typical resolutions offered by the printer are 150, 300, 600 and 1200 dpi. So the size of the image being printed on the paper depends on the dpi selected. For example, one of the test images was 1080×1080 pixels, therefore, if we print with 300 dpi, and one dot per pixel, we get an image of size 3.6×3.6 inches.

C. Scanning of the image

The printed data is then scanned using the Canon CanoScan LiDE200 flatbed scanner. The scanner is used in grayscale mode to avoid color to grayscale conversion in software. The scanner used gives multiple options for selecting the resolution of the

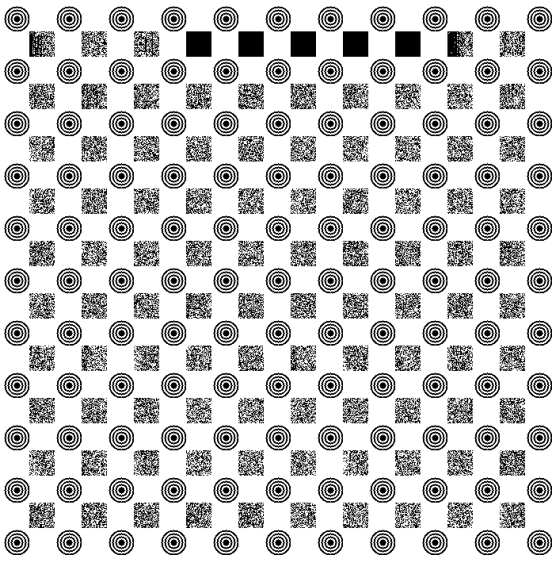


(a) Pattern 1



(b) Pattern 2

Fig. 22. Different patterns embedded for image skew correction (I)



Pattern 3

Fig. 23. Different patterns embedded for image skew correction (II)

image being scanned back. Like the printer, the scanner allows the user to choose the resolution at which the image is to be read. This allows us to oversample the printed data. For the experiments, the image was oversample by 4, 9 and 16. Oversampling gives us more data points and hence a better approximation while restoring the image. We will talk more about this in the following subsection.

D. Image alignment

Image restoration is the biggest challenge in the physical system being used. After passing through the printer and scanner, the image is distorted and the distortion may be non-uniform over the entire image. In order to estimate the original image from the received one, we assume a simple linear model for the distortion of the image. Let the original image be $I(x, y), 0 < x, y < n$. Let the received image be $I'(x', y'), 0 < x', y' < n'$, where x, y, x', y' are all discretized values. We assume that a linear transformation maps I to I'. Let

$$\begin{aligned} x' &= a_1 x + b_1 y + c_1 \\ y' &= a_2 x + b_2 y + c_2 \end{aligned}$$

We can rewrite this in a matrix form as

$$\begin{bmatrix} x & y & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & x & y & 1 \end{bmatrix} \begin{bmatrix} a_1 \\ b_1 \\ c_1 \\ a_2 \\ b_2 \\ c_2 \end{bmatrix} = \begin{bmatrix} x' \\ y' \end{bmatrix}$$

or

$$A\underline{m} = \underline{n} \text{ where } A = \begin{bmatrix} x & y & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & x & y & 1 \end{bmatrix}, \underline{m} = \begin{bmatrix} a_1 \\ b_1 \\ c_1 \\ a_2 \\ b_2 \\ c_2 \end{bmatrix}, \underline{n} = \begin{bmatrix} x' \\ y' \end{bmatrix}$$

In order to get an estimate for $\underline{m}$ we need known values of A and $\underline{n}$. We use the pilot patterns for this purpose. As described in Chapter IV.C, we embed the patterns as described. Then, as described in the same subsection, we get the locations of these pilot patterns by correlating the obtained image with the pilot patterns. So now we have the mapping of 4 of the pilot points from the original image onto the received image. So we now have the original points $(x_1, y_1), (x_2, y_2), (x_3, y_3), (x_4, y_4)$ and the mapped points $(x'_1, y'_1), (x'_2, y'_2), (x'_3, y'_3), (x'_4, y'_4)$. Now we can get a least square estimate for $\underline{m}$ as follows:

$$\underline{m'} = (A^T A)^{-1} A^T \underline{n}$$

$$A = \begin{bmatrix} x_1 & y_1 & 1 & 0 & 0 & 0 \\ x_2 & y_2 & 1 & 0 & 0 & 0 \\ x_3 & y_3 & 1 & 0 & 0 & 0 \\ x_4 & y_4 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & x_1 & y_1 & 1 \\ 0 & 0 & 0 & x_2 & y_2 & 1 \\ 0 & 0 & 0 & x_3 & y_3 & 1 \\ 0 & 0 & 0 & x_4 & y_4 & 1 \end{bmatrix}, \underline{n} = \begin{bmatrix} x'_1 \\ x'_2 \\ x'_3 \\ x'_4 \\ y'_1 \\ y'_2 \\ y'_3 \\ y'_4 \end{bmatrix} \text{ and } \underline{m'} = \begin{bmatrix} a'_1 \\ b'_1 \\ c'_1 \\ a'_2 \\ b'_2 \\ c'_2 \end{bmatrix}.$$

After obtaining an approximation for the pixel mapping, we find an approximation of the pixel map for the original image as mapped onto the received image. We obtain

$$x'' = a'_1 x + b'_1 y + c'_1, y'' = a'_2 x + b'_2 y + c'_2$$

The next step is to obtain the value of the image at the pixels (x'', y''). This can be done using bilinear interpolation as described in Chapter IV. This is the simplest form of image restoration.

While doing the actual experiments, it was observed that the distortions were not uniform over the entire image. In order to overcome this, more patterns were embedded in the image as shown in Figure 23. The linear mapping is then obtained locally for each region in the image, taking into consideration the closest 4 pilot points in the image. This gives us a better approximation of the original image.

E. Estimation of ISI

In order to estimate the ISI of the channel, we could potentially use the pilot pattern. The original pilot patterns are known to us. We consider a 3×3 pixel block for ISI estimation. This is again done using least squares estimation. We have the original image $I(x, y)$ and the restored image $I'(x, y)$, let the ISI pattern be

$$H = \begin{bmatrix} \alpha_{11} & \alpha_{12} & \alpha_{13} \\ \alpha_{21} & \alpha_{22} & \alpha_{23} \\ \alpha_{31} & \alpha_{32} & \alpha_{33} \end{bmatrix}$$

Using this,

$$\begin{aligned} I'(x,y) &= I(x-1,y-1)\alpha_{11} + I(x-1,y)\alpha_{12} + I(x-1,y+1)\alpha_{13} + \\ & I(x,y-1)\alpha_{21} + I(x,y)\alpha_{22} + I(x,y+1)\alpha_{23} + \\ & I(x+1,y-1)\alpha_{31} + I(x+1,y)\alpha_{32} + I(x+1,y+1)\alpha_{33} \end{aligned}$$

This can be modeled into a one dimensional problem as

$$A^T = \begin{bmatrix} I(x-1,y-1) \\ I(x-1,y) \\ I(x-1,y+1) \\ I(x,y-1) \\ I(x,y) \\ I(x,y+1) \\ I(x+1,y-1) \\ I(x+1,y) \\ I(x+1,y+1) \end{bmatrix}, p = \begin{bmatrix} \alpha_{11} \\ \alpha_{12} \\ \alpha_{13} \\ \alpha_{21} \\ \alpha_{22} \\ \alpha_{23} \\ \alpha_{31} \\ \alpha_{32} \\ \alpha_{33} \end{bmatrix};$$

Then,

$$Ap = I'(x,y)$$

The original pilot pattern forms the matrix A, and the received and restored pilot pattern is I'. Then, the least squares estimate of p is given by

$$p = (A^T A)^{-1} I'$$

F. Equalization and decoding of bits

After doing the above, we have an estimate of the data that was send over the channel. Now using the techniques described in Chapters IV and III.G for equalization and decoding, an approximation for the encoded bits $\hat{I}$ is obtained from the restored image.

In order to obtain the information bits from the encoded bits, we need to look at the structure of the generator matrix. As described in Chapter III, the generator matrix of the proposed code contains a permutation of the identity matrix of size $k \times k$, where k is the number of information bits encoded by the LDPC code. Therefore, to get the information bits back, we need to do the following. Let us say we are looking for information bit i. Find the column of the generator matrix that has all zeros but 1 in the i^{th} row, let this column be j . Therefore, the j^{th} bit of the encoded output would be the i^{th} information bit.

For noise estimation, we look at the blank part of the image (which has no data printed on it) and then use that to get an estimate of the noise power. It is done in the same way as is described in Chapter IV.D.

G. Assumptions

There were a lot of challenges involved in the physical experiments. Below is a list of all the assumptions made and how the deviations from those assumptions were handled.

1. For the simulations, the ISI was assumed to be uniform all over the image.

However, because of the printer and scanner attributes, that was not true. In order to handle this, the ISI can be estimated for each small block individually. For this purpose, some pilot bits can be embedded in each block and they can be used to estimate the ISI for that individual block. This was beyond the scope of our research and was not done. For some of the physical experiments, we used the original data that was printed to estimate the ISI.

For future work, we can use more complex methods like decoding the first few rows and columns in each block, and then using the knowledge of these bits to estimate the ISI as we move for further decoding.

2. The noise is also assumed to be uniform over the entire image which may not be the case. Similar to the ISI estimation, the noise estimation can also be done block wise with the help of pilot bits.
3. For the simulations, we assumed that there would be no scaling of the image. In order to find the position of the pilot patterns, we would need to know how the image is scaled, otherwise the matched filter that we use for position detection would not be accurate. For the experiments done, the scaling was determined manually. We believe that this process can be automated with some extra work.
4. The distortion of the image was assumed to be linear. There might be other kinds of distortions introduced, and higher order models could be used for a better approximation.

H. Results obtained and conclusions

The results of the experiments done are tabulated in Table II.

Within the scope of the work, we found that when the data is printed at 4 dots per

Table II. Error rates with different pattern and data rate

Pattern	Pixels per bit	Print dpi	Scan dpi	Bit error rate	Word error rate
1	1	300	300	0.4008	0.9244
1	1	300	600	0.4007	0.9389
1	1	300	800	0.2921	0.9667
1	1	300	1200	0.2721	0.96
2	1	150	300	0.3382	0.8
2	1	300	300	0.4309	0.9833
2	1	300	1200	0.3703	0.8933
3	4	600	1200	0.045	0.3667
3	4	600	2400	0.0026	0.01

pixel, with a code of rate 0.375, it can be successfully decoded. We have not been able to decode successfully with any lower resolution, but that could be attributed to the assumptions listed above. We could successfully decode 41.435 kB of data printed on a paper of size 6.7×6.7 inches. The capacity of paper is clearly greater than 0.9230 kB per square inch, because we could improve upon alignment methods and pack data more efficiently on paper.

I. Future work

As a follow up to this work, the following could be used to improve the performance of the system:

1. Instead of using a codeword with binary alphabets, we could use a code over a

higher-order Galois field.

2. Multi level coding could be used to improve the performance of the system. For instance, we could first encode the bits using an outer code like a Reed Solomon code, and then use an LDPC code as the inner code.
3. Pilot patterns could be embedded in each block, or depending on the quality of the channel, each $n \times n$ block, in order to estimate the ISI of a region.

REFERENCES

[1] R. G. Gallager, "Low-density parity-check codes," *IRE Trans. Inform. Theory*, vol. 8, no. 1, pp. 21–28, Jan. 1962.

[2] D. J. C. MacKay and R. M. Neal, "Near Shannon limit performance of low density parity check codes," *Electronic Letters*, vol. 32, no. 18, pp. 1645–1646, Aug. 1996.

[3] J. Thorpe, "Low-density parity-check (LDPC) codes constructed from protographs," *IPN Progress Report*, vol. 42, no. 154, Aug. 2003. [Online]. Available: http://tmo.jpl.nasa.gov/progress_report/42-154/154C.pdf

[4] A. Felstrom and K. Zigangirov, "Time-varying periodic convolutional codes with low-density parity-check matrix," *IEEE Trans. Inform. Theory*, vol. 45, pp. 2181 – 2191, 1999.

[5] S. B. D. J. Costello, A. E. Pusane and K. S. Zigangirov, "A comparison between ldpc block and convolutional codes," in *Proc. 1st Annual Workshop on Inform. Theory and its Appl.*, San Diego, CA, Feb. 2006.

[6] T. R. S. Kudekar and R. Urbanke, "Threshold saturation via spatial coupling: Why convolutional ldpc ensembles perform so well over the bec," in *Proc. IEEE Int. Symp. Information Theory.* Austin, Texas, USA: IEEE, Jan. 2010.

[7] S. Chung, T. J. Richardson, and R. L. Urbanke, "Analysis of sum-product decoding of low-density parity-check codes using a Gaussian approximation," *IEEE Trans. Inform. Theory*, vol. 47, no. 2, pp. 657–670, Feb. 2000.

[8] M. Marrow, "Detection and modeling of 2-dimensional signals," Ph.D. dissertation, University of California, San Diego, 2004.

[9] L. R. Bahl, J. Cocke, F. Jelinek, and J. Raviv, "Optimal decoding of linear codes for minimizing symbol error rate," *IEEE Trans. Inform. Theory*, vol. 20, no. 2, pp. 284–287, March 1974.

VITA

Aparna Khare was born in Guna, India. She received her Bachelor of Technology degree in electrical and electronics engineering from National Institute of Technology Karnataka, Surathkal in May 2007, and her Master of Science degree in electrical and computer engineering from Texas A&M University in December 2010. Her graduate thesis topic was "Capacity and Coding for 2D Channels." Her primary interests include channel coding, information theory, and image processing.

Ms. Khare can be reached at aparna1909@gmail.com, or at : Department of Electrical and Computer Engineering, Texas A&M University, 214 Zachry Engineering Center, College Station, Texas 77843-3128.