Constrained Coding and Signal Processing for Holography

A Thesis
Presented to
The Academic Faculty

by

Shayan Garani Srinivasa

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy

School of Electrical and Computer Engineering
Georgia Institute of Technology
August 2006

Constrained Coding and Signal Processing for Holography

Approved by:

Dr. Steven W. McLaughlin, Advisor
School of Electrical and Computer Engineering
Georgia Institute of Technology

Dr. John R. Barry
School of Electrical and Computer Engineering
Georgia Institute of Technology

Dr. Aaron Lanterman
School of Electrical and Computer Engineering
Georgia Institute of Technology

Dr. William T. Rhodes
School of Electrical and Computer Engineering
Georgia Institute of Technology

Dr. John A. Cortese
Georgia Tech Research Institute
Georgia Institute of Technology

Date Approved: 28 June 2006

To my parents.

ACKNOWLEDGEMENTS

I am really fortunate to work with Prof. Steven W. McLaughlin. He has been more than just a great advisor to me. He initiated a wonderful problem for my thesis, gave me complete freedom to explore my thoughts, and provided insightful comments throughout my research. For his patience, encouragement, and support, I sincerely thank Steve. In many aspects of life, other than just information theory, I would like to emulate him.

I thank Profs Barry, Rhodes, and Lanterman for serving on my reading and defense committees and for many helpful comments. I would like to thank Dr. John Cortese from GTRI for stimulating discussions on quantum information theory and for serving on my defense reading committee.

I would like to thank Prof. Adibi, Prof. Fekri, Omid and Arash of the ultramem group for many helpful discussions on optics related issues of holographic systems. The multidisciplinary nature of research discussions during ultramem meetings has been very useful from a holographic systems perspective.

I thank Dr. Patrick Lee from Western Digital for giving me an opportunity to work with him during summer 2005 on post-ECC modeling of magnetic recording channels. I will cherish his insightful discussions on many topics of technical and non-technical interest and for his warm friendship.

GCATT 562 has been an interesting lab for many reasons. I would like to thank my lab mates for a fun-filled atmosphere. I would like to thank my good friend Dr. Aravind Nayak for wonderful discussions on many topics outside academics and for patiently reviewing and commenting on my thesis work. I am thankful to Ms. Patricia Grindel for helpful editorial comments.

My family has been very supportive of me from my childhood. I greatly thank my parents Prof. G. R. Srinivasa and Mrs. Vathsala for their care, love, encouragement, and support. I also thank my sister Ranju and grandmother Smt. Ranganayaki for being

affectionate and cheering me up during agitated moments.

I would like to thank all my close friends and relatives who have been associated with me. Lastly, I should take this opportunity to thank all my teachers from nursery till my doctoral studies for teaching me many valuable things in life and motivating me to accomplish.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

SUMMARY

The increasing demand for high density storage devices has led to innovative data recording paradigms like optical holographic memories and patterned media. In holographic memories, data is stored in the form of two-dimensional pages within the volume of a recording material. These memories promise ultra-high volumetric densities ($1Tb/cm^3$) and ultra-fast readout speeds ($1Gb/s$) and could be the future hard drives based on nanoscale technology. To realize the potential of these devices, significant research needs to be done in many multi-disciplinary areas, such as material science, optics, information theory, and signal processing. Consequently, coding theorists and signal processing practitioners are interested in developing efficient two-dimensional constrained codes, error correcting codes, and signal processing algorithms for holographic systems.

Holographic data storage is just one application where two-dimensional coding theory can be put into practice. From a theoretical perspective, the study of two- and higher-dimensional constrained systems is an active area of research in symbolic dynamics, a branch of mathematics dealing with discrete dynamical systems. The applications of this theory have deep consequences in other areas, such as mathematical physics, finite automata theory, and languages in computer science. Thus, it is important to understand the fundamental limits of two-dimensional constrained systems.

The theory behind one-dimensional constrained channels is well known. There are a number of algorithms for constructing codes with rates as close to capacity as desired. However, computing the capacity and constructing codes for higher-dimensional constrained systems is an open problem. There are a few cases where tight bounds for two-dimensional capacity are known. Also, there are hardly any efficient algorithms for constructing two-dimensional codes.

In this thesis, we propose tiling algorithms for constructing a few classes of two-dimensional runlength-limited codes on a rectangular lattice and derive bounds for the channel capacity. We present sequential nested-block coding algorithms with rates close to the derived capacity lower bounds. The proposed tiling algorithms are constructive and have low encoding complexities. Motivated by recent advances in localized holography, we generalize our bounds and coding algorithms for two-dimensional M-ary runlength-limited channels.

The storage and retrieval of information from a holographic drive can be modeled as data transmission over a noisy communications channel. We derive a lower bound for the capacity of holographic channels and analyze the density versus multiplexing trade-off. This result is useful for deciding the number of recorded pages and for choosing the right code for maximizing the volumetric storage density.

The pixel misregistration problem is an important topic in signal reconstruction theory. In a holographic system, the spatial light modulator (SLM) and the detector arrays are not perfectly aligned. This leads to interpixel crosstalk. We develop a channel model and propose an algorithm for recovering data bits in the presence of pixel misregistration and noise.

CHAPTER I

INTRODUCTION

The storage and retrieval of digital information is an important aspect of digital communication theory and practice. Information theory provides a basis for understanding the fundamental limits for reliable data storage. Using advanced coding theory techniques, practical algorithms can be designed which approach these theoretical limits. Thus, computing limits for achievable data rates in storage channels and designing practical codes that approach these limits constitute the key steps for realizing devices with ultra-high storage densities.

Almost all storage channels in magnetic and optical recording are imperfect. In other words, channel limitations and imperfections coupled with media noise make it unsuitable for writing information bits directly into these channels. We need a combination of modulation codes and error correction codes to work with these systems. Modulation codes modify the information bits to suit the channel requirements. For example, a modulation code can be used to minimize the amount of spectral power at zero frequency. Error correction codes (ECC) are used for protecting against the channel errors by adding redundancy to the data so that the noise-corrupted data patterns can be retrieved correctly.

Many electronic devices, like magnetic discs, compact discs (CDs), and digital video discs (DVDs), popularly use a class of constrained modulation codes called runlength-limited (RLL) codes. These codes are used for reducing the intersymbol interference (ISI) and for ensuring timing recovery during detection by maintaining the runlength of zeros between any two consecutive ones within a specified range. RLL codes are also used for spectral shaping. In addition to constrained codes, powerful algebraic error correcting codes, such as Reed Solomon (RS) codes, are used to combat the effects of media noise and erasures resulting from scratches. Recently, iterative codes, such as low-density parity-check codes (LDPCs) and turbo codes, have been proposed for high-density data storage applications.

The advent of novel optical recording technologies, like volume holographic memories, has sparked significant research interest in the coding theory and signal processing communities. In a holographic system, data is recorded in the form of two-dimensional (2-D) pages within the volume of a holographic material. Since an entire page of data can be retrieved at once, this storage technique promises ultra-fast data access rates. Since many pages can be recorded within the medium, ultra-high storage densities can be realized. Consequently, there is an increased theoretical interest in developing 2-D constrained codes and error correcting codes for holographic applications. Two-dimensional constrained codes can be used for overcoming the effects of interpixel interference (IPI) and for ensuring timing recovery during detection. The capacity analysis of 2-D constrained channels and the construction of 2-D rate-efficient constrained codes are some of the important challenges in this field.

The storage and retrieval of holographic data can be viewed as data transmission over a noisy channel that distorts information either in a deterministic or in a stochastic manner. Computing the capacity of noisy holographic channels and designing 2-D error correcting codes for correcting burst errors of a given geometrical shape are some of the important problems in this area.

In a much broader context, analyzing the capacity of higher-dimensional constrained channels and designing multi-dimensional constrained codes are fundamental problems in symbolic dynamics and in the emerging field of multi-dimensional information theory. The theory behind constrained coding has many applications in other fields, like finite automata theory, statistical mechanics, and theoretical chemistry.

In this dissertation, we focus on the constrained coding and signal processing aspects for holography. On the information-theoretic front, we examine the noiseless capacity of 2-D binary runlength-limited channels. We derive bounds for the capacity of a few RLL channels and propose 2-D codes that achieve the lower bounds. The derived bounds and coding techniques are constructive and can be easily generalized to the multi-dimensional case. Motivated by recent developments in localized holographic recording, we examine the capacity of multi-level 2-D RLL constrained channels for high-density applications. We generalize our bounds and coding algorithms on the binary RLL constraints to 2-D M-ary

RLL constraints. We also derive a lower bound for the capacity of a noisy holographic channel and use this result for optimizing the number of pages that can be stored within the medium. This result has some interesting theoretical and practical implications.

Signal processing is an essential component in any data storage system design. Some of the signal processing challenges for holography include modeling, 2-D equalization for IPI compensation, and noise removal. A practical holographic device can have many shortcomings despite careful engineering. Some of these issues arise because of the limitations in the fabrication and design of the optical components. The inherent effects of the band-limiting aperture, diffraction, misfocus, optical aberration, material shrinkage, and mechanical motion of optical components lead to interpixel crosstalk. Signal processing algorithms are needed for recovering the data by removing the residual energy from unintended pixels. By designing efficient algorithms for overcoming media noise and interpixel crosstalk, the signal-to-noise ratio (SNR) can be improved. This facilitates increased data storage densities. On the signal processing front, we propose algorithms for signal recovery resulting from combined 2-D translational and rotational misalignments. Our technique can be applied to other optical imaging systems with square apertures.

The remainder of the thesis is organized as follows. In chapter 2, we discuss the physics of holographic storage. We present an overview of the information-theoretic concepts related to constrained channels. This overview serves as background information for the succeeding chapters. We also outline the signal processing challenges in holography and review some of the existing techniques and algorithms. In chapter 3, we present two algorithms for constructing $(1, \infty, d, k)$ 2-D RLL arrays and derive bounds for the capacity of these constrained channels. In chapter 4, we present two tiling algorithms for constructing a class of $(d_1, \infty, d_2, \infty)$ and $(0, k_1, 0, k_2)$ 2-D RLL arrays. Using these constructions, we derive the capacity bounds and coding algorithms for these constrained channels. In chapter 5, we extend our constructions and capacity bounds on the binary RLL constraints to a more general class of M-ary 2-D RLL channels. In chapter 6, we derive bounds for the capacity of noisy holographic channels and present an analysis of the fundamental trade-off between the storage density and multiplexing of holograms. In chapter 7, we address some of the signal processing issues in a holographic channel. In particular, we develop a technique for signal recovery resulting from 2-D pixel misalignment. Finally, in chapter 8, we present conclusions and ideas for future work. Detailed derivations and additional results are relegated to the Appendices.

CHAPTER II

BACKGROUND INFORMATION

In this chapter, we present an overview of the principles of holographic recording, highlight the coding and signal processing issues in holographic systems, review the basic concepts relating to constrained channels, and discuss previous work related to the contents of this thesis.

The increasing demand for higher storage densities at lower costs necessitates evolutionary storage technologies. Conventional magnetic and optical recording [3] are reaching a point where physical constraints limit the storage of tiny individual bits on the surface of the medium. The advent of holographic data storage offers many advantages and could be a viable alternative to conventional data recording technologies. Holographic storage is a volumetric approach in which information is recorded as an optical interference pattern within the holographic material. Since data is stored within the volume of the medium as opposed to on the surface, intriguingly high storage densities can be realized. We need innovation in the fabrication of optical holographic materials and components, the development of advanced coding and signal processing algorithms, and the design of efficient system architectures to realize a practical holographic device.

In section 2.1, we give a brief overview of holographic recording. We present the physics of holographic recording, highlight the advantages of holographic memories, and present the configuration of a working holographic system. In section 2.2, we discuss the media requirements for a holographic device and point out some important optical considerations. In section 2.3, we discuss the coding and signal processing aspects of holography. In section 2.4, we introduce constrained channels and discuss their information-theoretic aspects. In section 2.5, we highlight the pixel misregistration problem. We summarize our discussions in section 2.6.

2.1 Holographic Recording

In holographic storage, a page of information is stored as an optical interference pattern by intersecting two coherent laser beams at a spot within the volume of the medium, as shown in Figure 1. The stored interference pattern is called a hologram. Depending on the material properties, several holograms can be stacked within the medium. The theoretical limits for data storage density could be tens of terabits per cubic centimeter [3]. In subsection 2.1.1, we discuss the actual storage and retrieval mechanism in greater detail.

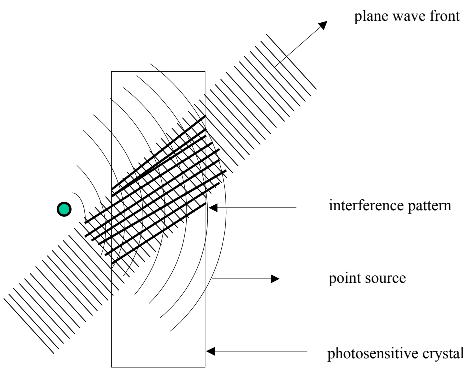


Figure 1: Schematic of a grating pattern produced by the interaction of a point source with a reference beam. The interference pattern is stored in a photosensitive crystal.

2.1.1 Physics of Holographic Storage

In a coherent holographic setup, a laser beam called the object beam intersects another coherent plane wave front called the reference beam to produce a grating pattern. The light and dark regions of the grating pattern interact with the photosenstive material, causing the transportation and trapping of electrons within the medium. This results in local changes in the physical/chemical properties of the material such as refractive index, absorption, or thickness of the medium. Thus, grating patterns are replicated and stored as local changes in the material properties. This is how holograms are stored within the medium. To recover one of the two interfering beams, the stored grating pattern in the material is illuminated with the other beam. In other words, by illuminating the grating pattern with the reference beam, the object wave can be reconstructed. This is illustrated in Figure 2.

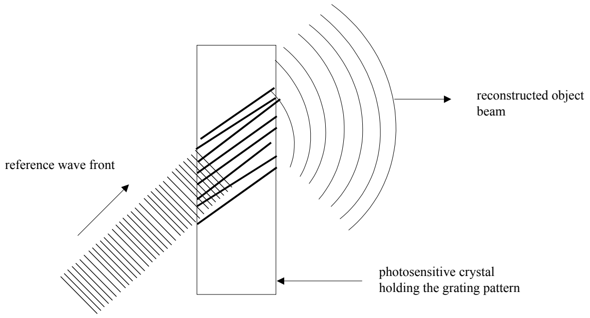

reconstructed object beam
reference wave front
photosensitive crystal holding the grating pattern

Figure 2: Schematic of re-creating the object beam by illuminating the stored interference pattern with a reference beam.

By changing the wavelength of the source, or by changing the reference angle (angle multiplexing), several holograms can be stacked within the material, depending on its properties. The retrieval of holograms can be done independently by illuminating the stored grating pattern with the reference beam that was used for creating it.

Figure 3 shows a basic holographic setup. A laser source modulates a programmable pixelated grid array called a spatial light modulator (SLM) to form an object beam. The SLM is typically a liquid crystal panel found in most electronic displays. The information-bearing object beam interferes with the reference beam, creating a grating pattern within the medium. The original object beam is later retrieved by illuminating the stored pattern with the reference wave that created it. The re-created object beam is projected on a high-quality pixelated array of photodetectors called a charge-coupled device (CCD) to recover the digital data. The imaging process must be of very high quality for replicating the original data. Despite a well-engineered imaging system, there will always be minor optical aberrations as a result of diffraction and misfocus, resulting in interpixel crosstalk. It is possible to avoid having an expensive imaging system by using phase-conjugated readout. In phase-conjugated readout, the reconstructed object beam gets backpropagated through the same optics that created it, thereby compensating for some of the imaging imperfections. However, the imaging systems must be properly aligned. Optical and mechanical distortions such as translation and rotational misalignments coupled with magnification errors introduce additional errors. This requires special coding and signal processing techniques for data recovery. We discuss these problems in section 2.2.

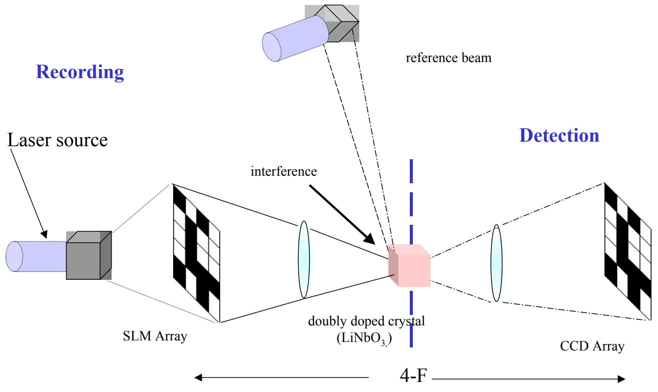


Figure 3: Basic holographic system configuration: recording and detection.

2.1.2 Advantages of Holographic Memories

Holographic memories offer a number of advantages. In this subsection, we discuss some of the benefits of holographic storage.

- Ultra-high Capacity: Data is recorded in the form of two-dimensional pages. Several thousand pages can be stored within the volume of the medium, depending on the material properties. Thus, holographic memories promise very high densities. However, the ultimate limits for storage density are dictated by the available SNR that the material can provide. By designing sophisticated codes that can achieve these limits, coding gains can be realized. Ultra-high density in holographic memories can be realized through a combination of better media and advanced coding techniques.

- Ultra-fast Data Rates: Unlike disk drives that require electro-mechanical actuators for accurate positioning, optical laser beams do not have any inertia [3]. This saves a considerable amount of time in reading and writing. Most important, data is inherently stored and retrieved in a pagewise manner. This amounts to massive parallelism, a feature not found in conventional storage. The overall system can be viewed as a high rate data channel realized from relatively slower low-cost parallel channels.

- Associate Retrieval: This is one of the novel features of holographic storage and is akin to content addressable memories found in neural networks. To understand what this means, consider the problem of indexing a hologram without knowing its address. By embedding a search pattern on the object beam and illuminating the stored patterns, we can reconstruct all the reference beams that were used to create the patterns. The intensity that is diffracted by each grating into its corresponding reference beam is proportional to the search pattern and the content of the data page. By choosing the reference beam with the highest intensity and reading the corresponding page with this reference, we can obtain the closest match to a given search pattern. This concept has an interesting application as a parallel search algorithm in very large databases.

While all these advantages are evidently foreseen, the cost of lenses and lasers is a limiting factor for the mass production of these systems at present.

2.1.3 Components and Configurations of a Basic Holographic System

The important components of a holographic setup are shown in Figure 3 [3]. A laser source such as a krypton laser (676nm) is used for producing a coherent plane wavefront for creating grating patterns. The SLM modulates digital information onto a laser beam, producing an object beam. The two lenses are used for imaging the data. A storage material such as doubly doped lithium niobate or barium titanate records the holograms. The CCD array is used to collect data from the reconstructed object beam. The SLM and CCD arrays are usually a few tens of micrometers in pitch size and hold around one mega pixel. Other optical hardware components such as collimators, beamsplitters, and waveplates are needed. Collimators are used for producing a fine point beam. Beamsplitters split the laser beam into two parts; one beam creates the object wavefront, and the other beam is used for reference. Waveplates are needed for controlling the amount of polarization. Special hardware is needed for aligning the SLM and CCD. Depending on the type of multiplexing, additional hardware is needed. For example, in the case of angle multiplexing, a beam-steering system is required for changing the angle of the reference beam. For a wavelength multiplexing system, a fast tunable laser source is needed.

We now discuss different configurations for a holographic storage system. The most popular configuration is the '4-F' configuration, as shown in Figure 2.3. In this setup, the two lenses are separated by the sum of their focal lengths, and the SLM and CCD are four focal lengths apart. Each lens produces a Fourier transform in two dimensions [3]. The recorded hologram within the medium is in fact the Fourier transform of the SLM data. At the detector, the second lens Fourier transforms the recovered object beam, and the original data is imaged on the CCD. The imaging geometry, i.e., the angle between the reference and object beams is another factor that influences system performance. It is reported in [57] that limiting this angle to less than 90 degrees achieves better system performance compared to fixing the angle at 90 degrees.

Other holographic configurations, like localized recording [34], have been recently explored. In localized holography, the holographic crystal is divided into a number of slices. The reference beam is used to selectively sensitize each slice of the crystal. Using this technique, one can record a few hundred holograms compared to a thousand holograms in the angle multiplexed case. However, from a storage standpoint, localized holography provides improved SNR properties that can be exploited to design multi-level codes for maximizing the ultimate storage density. Localized holography offers many advantages, like the selective erasure of holograms and increased readout persistence. These features are not present in angle multiplexed holography.

In the following section, we highlight the media requirements for improving the dynamic range of holographic systems.

2.2 Media Requirements

We have discussed the basic principles of holographic storage. We need to explore materials that can actually meet the practical requirements. Ideally, it is desired to have a compactly sized material holding many holograms. In practice, there are many trade-offs between the size and the material properties affecting the overall system performance. In this section, we outline some of the desirable media properties and give examples of such recording materials.

The imaging of information from the SLM to the CCD must be accurate so that the information conveyed by a pixel at the SLM is accurately received at the intended detector pixel. Since the holographic material is inherently part of this imaging process, the recording medium must be fabricated in a homogenous way throughout its entire volume so that different areas accessed during readouts produce nearly the same image quality. The amount of noise produced as a result of the scattering of light by the material determines the fundamental limits on the data storage density and the bit error rate performance of the system. It is reported in [3] that inorganic crystals have a lower scattering level than the best organic media.

A rather obvious choice to improve absolute storage is to increase the thickness of the medium. However, limitations in the physical and chemical properties of the recording process prevent an arbitrary scaling of thickness.

In the next subsection, we review some basic terms referred to in holographic recording.

2.2.1 Sensitivity and Dynamic Range

The term sensitivity [3] refers to the amount of refractive index modulation per unit exposure. The diffraction efficiency (η) dictates the amount of energy in the readout signal and is proportional to the square of the modulation index and thickness. We can express the recording sensitivity S (in cm/J) as

$$S = \frac{\eta^{\frac{1}{2}}}{Ilt}, \tag{1}$$

where I is the total intensity, l is the medium thickness, and t is the exposure time. The normalized sensitivity $S_n = Sl$ can be used to compare different materials of varying thicknesses.

It is desired that the readout signal power be maximized and the readout times be minimized.

The term dynamic range refers to the overall response of the medium with many stored holograms in it. This is often expressed in terms of a quantity called $M/\#$, introduced in [33]. For angle multiplexed holography, the diffraction efficiency (η_{angle}) is related to $M/\#$ and the number of holograms P as

$$\eta_{angle} = \left(\frac{M/\#}{P}\right)^2. \tag{2}$$

For localized holography, the diffraction efficiency is given by

$$\eta_{local} = \left(\frac{M/\#}{P}\right). \tag{3}$$

The parameter $M/\#$ can be measured experimentally [33] and depends on the physical properties of the photorefractive crystal such as impurity doping level, oxidation state, absorption coefficient, electro-optic coefficient and photoconductivity. Other factors, such as grating period, modulation depth, and stability of the interference pattern influence $M/\#$. The equation for characterizing $M/\#$ is given by

$$M/\# = \frac{A_0 \tau_e}{\tau_r} \tag{4}$$

where A_0 is the saturation grating strength, τ_r is the recording time constant, and τ_e is the erasure time constant.

Stabilizing holograms is an important issue in holographic storage. Many organic photopolymer materials are subjected to aging because of the induced stresses during recording and also because of the residual reactive substances left within the material. This causes erasures over a long period of time. Erasures are also likely to occur because of the thermal diffusion of molecules that record the hologram and can be restored by trapping the mobile charge carriers either by thermal or electronic fixing.

2.2.2 Read-write Holographic Storage

Holographic memories must be non-volatile and read-writeable for commercial applications. This requires the selective erasure and recording of holograms. The selective erasure of holograms can be accomplished by rearranging the trapped charges in the photorefractive material by light illumination. Selective charge re-excitations can often lead to the undesirable [10] erasures of other holograms during normal readouts. To overcome such effects, alternative methods for achieving non-volatile storage need to be explored. In one such method, recording is done at a wavelength of light that gets absorbed only in the presence of a third 'gating' beam of different wavelength. This third beam exists only during recording and is switched off when data is being read. Photorefractive materials like lithium niobate can be optimized for gated two-color recording by doping the crystal with two dopants such as manganese and iron, or by changing the ratio of lithium and niobium in the compound [10].

In the case of impurity dopants, one trap, such as Mn, creates a deep trap near the middle of the band gap, while the other dopant, such as Fe, creates a shallower trap near the valance band. Gating occurs in the deep trap. The shallower trap provides an intermediate level for gated recording. Charges excited in the shallower trap persist longer in the dark and can be populated with low-power laser beams.

To summarize, information is recorded by creating a grating pattern in the presence of a sensitizing beam such as a monochromatic beam of light in the visible region. Non-destructive readout is accomplished by using just the writing beam.

2.3 Coding and Signal Processing

In the previous sections, we discussed how innovative optics and material choice can lead to increased dynamic range and sensitivity. From a communications standpoint, the entire process of sending information, storing it as a hologram, and receiving it at the detector is just another instance of a noisy communications channel. The ultimate limit for the storage and transmission of information is limited by the noise floor in the channel. We need to compute information-theoretic limits for predicting the amount of data storage and for designing codes that can achieve those limits. We also need equalization and signal detection algorithms for compensating channel distortions and for recovering the data from detected samples. In this section, we discuss various coding and signal processing aspects of holography. Parts of this section are explored in greater detail in succeeding chapters.

2.3.1 Channel Modeling and Equalization

Signal intensity is a detected signal at the CCD. This must be transformed to digital data. The first step toward this task is channel equalization to undo the effects of distortion. To equalize the channel distortion, it is beneficial to have a channel model that accurately represents the recording and reading mechanism. Several models for holographic channels have been proposed. Heanue, Gurkan, and Hesselink [23] proposed a 2-D space-invariant interpixel interference channel model. In their channel model, data gets filtered by a two-dimensional channel transfer function. The filtered signal gets corrupted by additive white noise. The horizontal span L of the 2-D filter determines the length of the ISI memory. The inputs take values from a multi-level alphabet of size M, and detection is accomplished row by row by constructing a 2-D Viterbi algorithm with M^L states. This model works fine as long as there is a constant lateral misalignment in two dimensions. Chugg, Chen, and Neifeld [13] developed minimum mean squared error (MMSE) equalizers for two-dimensional finite contrast space-invariant ISI channels and studied the improvement of storage densities after equalization. Vadde and Kumar [53] considered two different channel models, one linear in amplitude and the other linear in intensity, and studied the effect of post-equalization for characterizing bit-error rates in these models. Keskinoz and Kumar [28] proposed the discrete magnitude-squared channel model by considering quadratic non-linearity in holographic systems. Other popular equalization techniques include adaptive 2-D equalizers using the least mean squares (LMS) algorithm.

Figure 4 shows the schematic of a physical model [28] for volume holographic memories. The input $x_{i,j}$ takes values from a discrete set of M-ary alphabets and gets filtered by the 2-D impulse response corresponding to the SLM shape $p(x, y)$. The resulting signal $s_1(x, y)$ is Fourier transformed by the first lens, filtered by the aperture shape transfer function $h_A(x, y)$, and finally replicated in the medium. During readout, the stored hologram is inverse Fourier transformed by the second lens. This signal is then focused on the CCD, which integrates the magnitude squared of the readout signal $s_2(x, y)$ over its region of support and outputs the intensity $I_{i,j}$. This model is closer to a physically realistic holographic channel model than the previous models [53], [23], [13]. We note that the system is assumed to be linear between the SLM and the CCD [28]. In reality, there could be channel non-linearities, in which case the classical notion of linearity and convolution do not apply. However, in practice such non-linearities can be ignored to develop a tractable mathematical framework using linear systems theory for the best approximation of the underlying process.

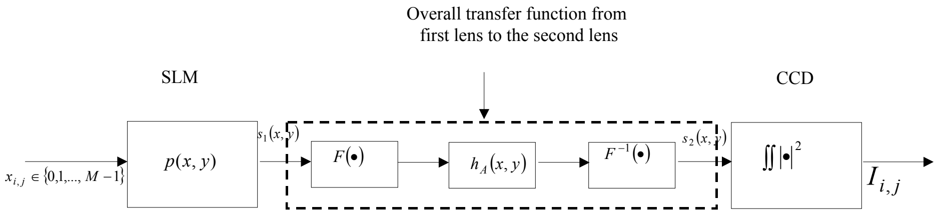


Figure 4: Physical model for a holographic system: linear system approximations.

2.3.2 Signal Detection

Signal detection is the next step after modeling and equalization. The goal of a signal detection algorithm is to recover information bits from the detected and equalized signal values. Detection techniques can be as simple as threshold detection [5] to more sophisticated maximum likelihood-based techniques [5].

- Threshold Detection:

  In threshold detection, the input pixels take values from an M-ary alphabet corresponding to the different gray-level pixel values. Based on the probability distribution of the received intensity as a function of the gray-level, optimum threshold levels are chosen for decoding the detected pixel values by minimizing the probability of bit-error rates. An illustration of threshold detection is shown in Figure 5. In cases where combined noise effects occur as a result of scattering and detector electronics, deciding optimum thresholds is often difficult. When spatial variations in intensity occur, threshold detection can perform poorly. In such cases, modulation codes can be beneficial to facilitate threshold detection.

- Maximum Likelihood:

  In maximum likelihood detection, information bits are sequentially decoded based on a sequence of observations. We briefly explain how maximum likelihood sequence detection is done in the 1-D case and then discuss 2-D detection done in [23]. ISI

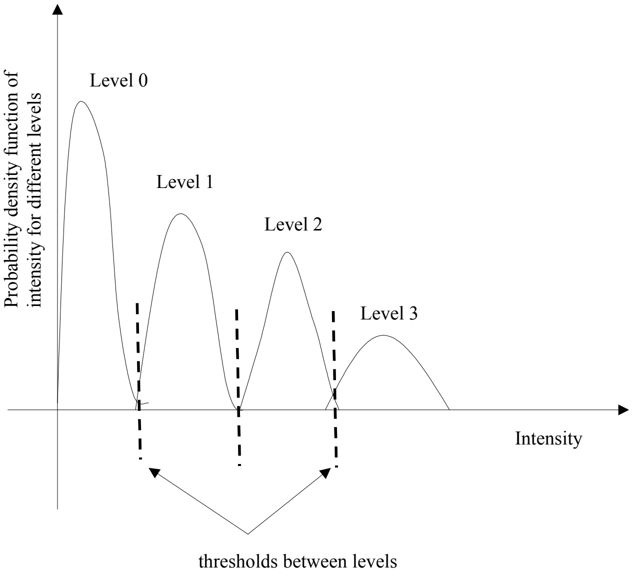


Figure 5: Example of a four level threshold detection scheme.

channels have memory. This means the received signal at any given instant depends on the previous samples. The operation of an ISI channel can be visualized by a trellis diagram shown in Figure 6.

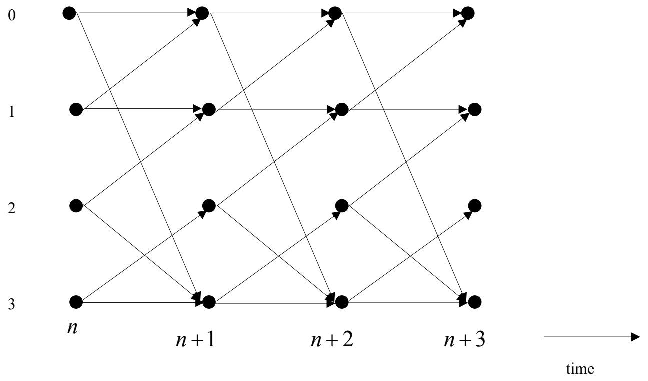


Figure 6: Example of a trellis diagram: There are four states in the trellis, each state transition results in the emission of a binary symbol.

Let us suppose that the channel has D delay elements in its impulse response and takes on binary input. The channel memory of D elements corresponds to 2^D states. For the sake of illustration, we choose two delays corresponding to four possible states '0', '1', '2', and '3', as shown in Figure 6. Depending on the binary input values, the channel outputs a received signal and moves to the next state, as shown in the trellis. As data symbols are transmitted, the system traces a path through the trellis. The sequence detection problem can be framed as follows. Given a sequence of k received values $\mathbf{r} = (r_1, r_2, ..., r_k)$, we are interested in obtaining the data sequence $\mathbf{x} = (x_1, x_2, ..., x_k)$ that maximizes the conditional probability

$$P(\mathbf{r}|\mathbf{x}) = P(r_1, r_2, ..., r_k | x_1, x_2, ..., x_k). \tag{5}$$

Since the channel memory is D, (5) can be simplified as

$$P(\mathbf{r}|\mathbf{x}) = \prod_{i=1}^{D} P(r_i | x_{i-D}, x_{i-D+1}, ..., x_i). \tag{6}$$

Depending on the noise statistics, (6) can be simplified. Using the Viterbi algorithm [5], the optimum data sequence that maximizes the maximum likelihood (ML) metric in (6) can be obtained.

Though ML detection for 1-D channels is well understood, extension to two dimensions is not straightforward. Two-dimensional Viterbi detection with decision feedback equalization (DF-VA) was studied in [23]. In the DF-VA algorithm, the Viterbi detector operates on the 2-D channel transfer function. For example, with a 2-D channel matrix of size $L \times L$, the ISI memory corresponds to length $L - 1$. The algorithm proceeds by decoding symbols row by row. In each iteration, it is assumed that data in the previous rows are known and correctly decoded. The next row is decoded by subtracting the information about the previous row. Similar detectors are used in multi-track channels in magnetic recording. The DF-VA technique cannot accomodate the time-varying changes as a result of rotational misalignments and non-uniform material shrinkages. More powerful techniques and generalizations are needed for handling time-varying channel effects. Error propagation is another limitation of this technique. Any error made in the decisions in the previous rows will affect the decisions in decoding the current row. Such error propagations can be catastrophic, leading to a large number of decoding errors, especially under low SNRs.

More sophisticated iterative-based detection is explored in [31].

2.3.3 Channel Codes

Channel codes can be classified into two categories, namely, modulation codes and error correction codes. Modulation codes are used to combat ISI and facilitate detection. Error correction codes introduce controlled redundancy across data pages to identify and rectify bit errors. In this section, we highlight the need for such channel codes through examples.

A. Modulation codes

The role of modulation codes is to shape the coded data according to the channel characteristics so that data is less prone to errors. Choosing an appropriate modulation code facilitates detection and decoding. We pointed out in subsection 2.2.2 that threshold detection performs poorly when there are frequent spatial variations in the data pages. An alternative to threshold decoding is to use balanced arrays (dc-free array codes) together with a simple detection scheme. Since the variation in pixel intensity is not much in a small local neighborhood of the detector array, coding data patterns using balanced arrays facilitates simple detection algorithms. For instance, by reading N binary coded pixels, $N/2$ pixels with the highest intensity can be declared '1' and the rest as zeros [3]. The real challenge is to design asymptotically unity rate 2-D balanced codes, i.e., codewords that have the least amount of redundancy. Algorithms for constructing such balanced arrays are reported in [36] and [52].

Sometimes we need codes for shaping the power spectrum of the channel data. Such codes are called spectral shaping codes. Low-pass filtering codes [4] (codes that eliminate patterns with rapidly changing ones and zeros, i.e., having high spatial frequency) and spectral-null codes [26] (codes that exhibit a null at zero frequency) are examples of spectral shaping codes. Compensating for IPI and ensuring timing recovery in 2-D detectors is an important application of modulation codes like runlength-limited codes [25] and checkerboard codes. Constructing efficient 2-D RLL codes is a challenging problem. We explore constrained codes in detail in the succeeding chapters.

B. Error Correction Codes

In holographic channels, errors occur in the form of 2-D bursts of a certain geometrical shape and are not independent and identically distributed. Further, error rates can vary over data pages [3]. For example, bits at the center of a page are less likely to be in error than those near edges. Designing simple and efficient 2-D burst error correcting codes for different error rate regions is an interesting problem.

Several authors [1], [9], [24] have investigated the problem of designing optimal burst error correcting codes on a rectangular lattice. To correct burst errors, optimal interleaving strategies must be adopted. The problem of optimal interleaving in one-dimension is straightforward. Designing interleaving strategies for higher-dimensional constraints is a challenging problem. Blaum, Bruck, and Vardy [8] developed efficient two- and three-dimensional interleaving schemes requiring the smallest possible number of distinct codes without repetitions. Etzion and Vardy [18] have investigated two-dimensional interleaving schemes with repetitions. A procedure for constructing two-dimensional burst error correcting codes for an arbitrary geometry and burst size and building efficient decoding algorithms is an open problem.

2.3.4 Capacity and Storage Density

We pointed out in the beginning of the thesis that the holographic channel is just another instance of a digital communication channel. Computing the Shannon limit [40] for maximum information transfer in a holographic channel provides insight for achievable storage densities in the system. This limit is a function of the signal and noise powers of the system and is insightful for developing codes. In fact, one can either develop new coding algorithms or modify a plethora of existing coding algorithms [29] to suit the channel requirements.

The overall density is a function of the available SNR in the medium and can be maximized by appropriately choosing the number of stored pages and the number of gray levels. Experimental capacity estimation is reported in [3]. In this thesis, we derive lower bounds for the capacity of the holographic channel. Using existing techniques, we can guarantee codes that can achieve the lower bounds.

2.4 Constrained Channels

Many discrete communication channels, such as those in magnetic and optical recording, [25] allow only a fixed set of input patterns. Such channels are called constrained channels.

The reference to constrained channels dates back to Shannon's classic paper [40]. In this section, we discuss 1-D constrained channels and extend the definitions to 2-D constraints.

2.4.1 One-dimensional Constraints: Definitions and Preliminaries

The maximum information rate for noiseless constrained channels was defined by Shannon [40] as follows:

Definition 2.1. *The capacity C of a constrained channel is given by $C = \lim_{T\to\infty} \frac{\log_2(N(T))}{T}$, where $N(T)$ denotes the total number of allowed signals of duration T.*

It was also proved by Shannon that there exists a coding scheme with an average rate R for reliably encoding the source information across the channel with capacity C defined above. The following theorem [40] relates the coding rate, channel capacity, and source entropy for reliable data transmission.

Theorem 2.1. *Let a source have an entropy H (bits per symbol) and a channel capacity C (bits per second). It is not possible to transmit at an average rate R (symbols per second) greater than $\frac{C}{H}$.*

An example of a constrained channel is the runlength-limited (RLL) channel frequently encountered in magnetic and optical recording [25]. The RLL channel accepts binary sequences with restrictions on the runlength of the number of zeros between consecutive ones and is defined below.

Definition 2.2. *A binary sequence $\{0,1\}^m$ satisfies the (d,k) constraint if there are at least d zeros and at most k zeros between any two ones in the sequence.*

The d constraint helps in mitigating intersymbol interference and the k constraint ensures adequate timing to the detector circuitry.

Sequences satisfying (d,k) constraints can be obtained by a random walk on the graph shown in Figure 7.

Theorem 2.2. *The capacity of a (d,k) constrained channel is given by [40]*

$$C_{(d,k)} = \log_2(\lambda_{max})$$

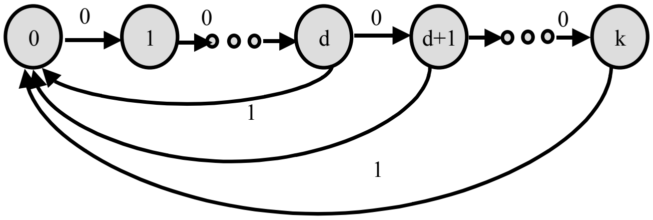


Figure 7: Constrained graph $\mathcal{G}_{(d,k)}$.

where λ_{max} is the largest real root of the characteristic equation $z^{k+1} - z^{k-d} - z^{k-d-1} - .... - z - 1 = 0$.

We can alternatively formulate computing capacity by a maxentropic random walk on $\mathcal{G}_{(d,k)}$, shown in Figure 7. The adjacency matrix for the graph $\mathcal{G}_{(d,k)}$ is given as $\mathbf{A} = [a_{ij}]$, where $a_{ij} = 1$ if state j can be reached from state i and zero otherwise. The following result is equivalent to Theorem 2.2 [40].

Remark 2.1. *Theorem 2.2 is equivalent to computing $\log_2(\lambda_{max})$ where λ_{max} is the largest eigenvalue of the adjacency matrix of $\mathcal{G}_{(d,k)}$.*

Codes designed for a RLL-constrained channel are called RLL codes. These codes can be either fixed length or variable length. A fixed-length (d, k) constrained code is a one-to-one mapping $f : \{0, 1\}^p \rightarrow \{0, 1\}^q$ of p information bits to q channel bits such that the rate $\frac{p}{q} < C_{(d,k)}$. There are coding schemes [30] that approach as close to $C_{(d,k)}$ as desired. There are rate-efficient variable length codes [7] that achieve $C_{(d,k)}$ for several choices of d and k.

2.4.2 Higher Dimensional Constraints

Holographic channels and channels for patterned media are classic examples of two-dimensional storage channels. Encoding data over 2-D channels can be visualized as writing sequences on a plane with correlations in both dimensions. Constrained channels need not be restricted to storage. Relationships to the entropy of formal languages and the probability of constructing crosswords are instances of the 2-D constrained coding problem [25]. The study of two- and higher-dimensional constraints has important applications in areas of mathematical physics and chemistry, such as statistical mechanics and lattice packings. For example, quoting Baxter [6], the arrangements of dimers on an $m \times n$ 2-D lattice can be uniquely defined by specifying if chemical bonds linking adjacent lattice sites are occupied by a dimer. The specification of chemical bonds in 2-D can be mathematically replicated to 2-D constraints. These specifications can be further generalized to multi-dimensional (multi-D) constraints when dealing with arrangements of dimers in solids. Thus, the theory of higher-dimensional constrained channels is a fundamental problem with many applications of interest in basic science and engineering.

There are several 2-D constraints of interest. Examples of some 2-D constraints include RLL constraints and checkerboard constraints. These constraints can be defined over lattices of different geometrical shapes such as a rectangle, hexagon etc.

We now introduce some definitions for 2-D constrained channels by extending the definitions in the 1-D case.

Definition 2.3. *The noiseless capacity of a 2-D constrained channel over a hyper rectangular lattice is given by* $C_{2D} = \lim_{m,n \to \infty} \frac{\log_2(N_{m,n})}{mn}$ *where* $N_{m,n}$ *is the total number of 2-D constrained sequences on a rectangular lattice of size* $m \times n$*.*

We are interested in the special case of 2-D RLL constraints for holography applications. An array satisfying 2-D runlength-limited constraints is defined as follows.

Definition 2.4. *A 2-D binary array satisfies* (d_1, k_1, d_2, k_2) *RLL constraints, if there are at least* d_1 *zeros and at most* k_1 *zeros between any two ones in all the rows, and at least* d_2 *zeros and at most* k_2 *zeros between any two ones in all the columns.*

Figure 8 shows an example of a 2-D runlength-limited constraint. The horizontal runlength of zeros is either one or two, and the vertical runlength of zeros is between one and three.

Computing the capacity C_{2D} for runlength-limited channels is a well-known open problem in the field. Using sub-additivity, it was proven in [27] that the limit in Definition 2.4 exists. However, there is no closed-form solution for exactly computing capacity. Bounds for the capacity of 2-D RLL channels are known in a few cases. Calkin and Wilf [12] obtained

1	0	1	0	0
0	1	0	1	0
0	0	1	0	0
0	1	0	0	1
1	0	0	1	0

(1,2,1,3) Runlength-limited constraint

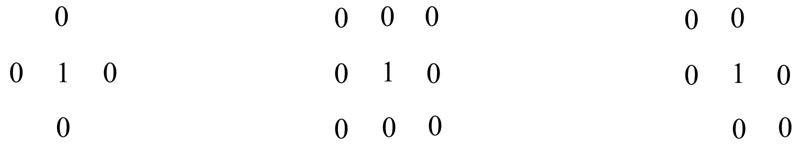
Figure 8: Example of a $(1, 2, 1, 3)$ runlength-limited constraint on a 5×5 rectangle.

tight bounds for the capacity of the hard square model, i.e., the $(1,\infty,1,\infty)$ constraint, based on an extension of Engel's results [16]. Kato and Zeger [27] derived capacity bounds in a few cases and demonstrated the existence of zero capacity regions in some cases. Forchhammer and Justesen [19] introduced a new class of 2-D random processes called cylindrical processes and used this concept to derive entropy bounds for 2-D constrained random fields.

Very few efficient algorithms exist for constructing 2-D RLL constrained codes. Siegel and Wolf [41] developed bit-stuffing lower bounds for symmetric (d,∞) constraints. Their construction is simple and effective for constructing variable rate two-dimensional codes for symmetric constraints. Roth, Siegel, and Wolf [39] derived an exact analysis of the bit-stuffing algorithm for the hard-square constraint and developed an efficient coding scheme for the same. Halevy et al. [21] extended the basic bit-stuffing bound idea to obtain improved bounds for some 2-D constraints. We explore 2-D RLL constraints in more detail in the suceeding chapters.

There are other non-RLL constraints, like the checkerboard constraints shown in Figure 9, in which zeros surrounding a one satisfy a certain geometric rule. For example, in the hexagonal constraint, a one must be surrounded by zeros in the north, south, east, west, and the diagonals in the northeast and southwest directions.

Weeks and Blahut [56] obtained numerical bounds for the capacity of some non-RLL checkerboard constraints and used Richardson interpolation to conjecture tighter bounds.



(a) Diamond constraint (b) Square constraint (c) Hexagonal constraint

Figure 9: Examples of checkerboard constraints. (a) One is surrounded by four zeros in a diamond geometry. (b) One is surrounded by zeros in a square geometry. (c) One is surrounded by six zeros in a hexagonal geometry.

Nagy and Zeger [35] derived the asymptotic capacity of open convex symmetric non-RLL checkerboard constraints and proved that channels with square, diamond, and hexagonal checkerboard constraints have the same capacity. In general, computing the capacity of 2-D constrained channels is a difficult problem. There is no known general solution for exactly computing capacity and for constructing codes for these channels.

2.5 Pixel Misregistration

Pixel misregistration is a well-studied problem in image processing [38]. There are many applications such as compressed video decoding, optical imaging, and magnetic resonance imaging where image misregistration problems are frequently encountered. Depending on different signal models, several misregistration compensation techniques have been developed.

In a holographic system, the signal received at the detector suffers from both pixel blurring and interpixel crosstalk. Deviations in the lateral pixel alignments of the SLM and CCD and effects of rotation and magnification can lead to significant interpixel cross talk, leading to reduced signal-to-noise ratios. Hence, we need algorithms for compensating the effect of misalignments before decoding any information.

Figure 10 shows the schematic of a lateral misalignment [11] between the imaging arrays.

The SLM and CCD have a fractional horizontal offset σ_x. Following the physical model in Figure 4 [28], the observed signal at the detector pixel is intensity (I) from the intended

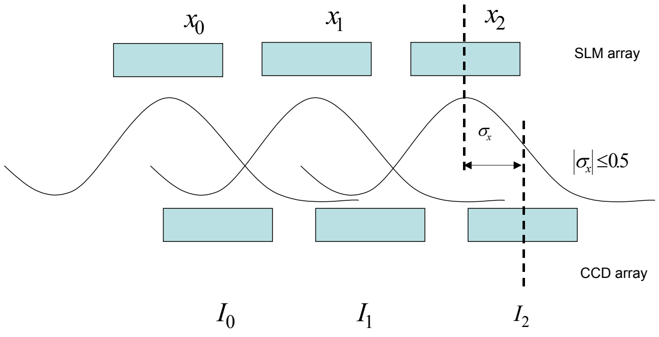


Figure 10: Schematic of lateral pixel misalignment.

pixels and cross talk from neighboring pixels. The goal of the problem is to recover the information bits (x) from observed samples. Burr and Weiss [11] developed non-linear compensation techniques for lateral pixel misregistration by sucessive interference cancellation. They showed improved error performance after compensation. However, time-varying interpixel cross talk resulting from combined translation and rotation has not been addressed in [11]. In chapter 6, we extend the algorithm of [11] for the combined translation and rotation case based on the assumption that the misalignment parameters are constant over different pages.

2.6 Summary

In this chapter, we presented an overview of the holographic system and its basic ingredients. We highlighted various system-level issues and challenges for realizing a holographic storage device. To benefit from holographic storage, a practical system should have the right choice of storage material, carefully designed optical components, efficient channel codes, and signal processing algorithms working in tandem.

In our current work, we look at holographic channels from a communications and information-theoretic perspective. We are interested in developing constrained modulation codes and signal processing algorithms for holographic channels. Though our work relating to 2-D constrained codes has practical applications to holography, it is a theoretical study on one of the important problems in mathematical physics. In the suceeding chapters, we examine 2-D constrained channels in detail. We present theoretical limits for the achievable storage density in holographic channels and suggest the application of multi-level codes. Finally, we address the pixel misregistration problem due to combined translation and rotation and present signal recovery algorithms.

CHAPTER III

CONSTRUCTIONS AND CAPACITY BOUNDS FOR $(1, \infty, D, K)$ RUNLENGTH-LIMITED CONSTRAINED CHANNELS

In chapter 2, we presented an overview of 2-D constrained channels. In this chapter, we examine the simplest type of 2-D RLL constraints, i.e., $(1, \infty, d, k)$ constraints. We present two algorithms for constructing $(1, \infty, d, k)$ binary RLL arrays. The first algorithm is based on the adjacency approach. The second algorithm is based on an iterative construction. We derive capacity bounds and present numerical results.

This chapter is organized as follows. In section 3.1, we present two constructions for obtaining 2-D RLL arrays satisfying the constraints. In section 3.2, we present bounds for the capacity of the constraints. Numerical results are discussed in section 3.3. The results are summarized in section 3.4.

3.1 Constructions

One-dimensional constrained sequences can be realized by a random walk on a constrained graph, as shown in Figure 7. However, such graphical representations are not known for 2-D RLL constraints. Hence, it is important to develop algorithms describing the construction of 2-D constrained arrays to gain insight for computing the capacity and for designing efficient codes. Etzion [17] described rules for merging any two arbitrary 2-D RLL arrays to an array that satisfies the given constraints and discussed the Hamming distance of such patterns. Such a merging approach in [17] requires constructing RLL arrays in the first place. In this section, we present two simple algorithms for constructing 2-D RLL arrays satisfying $(1, \infty, d, k)$ constraints on an $m \times n$ rectangular grid. The first approach is based on the well-known adjacency method [30]. The second approach is called the iterative approach

[45]. Both of these approaches are equivalent representations and are useful to understand the structure of 2-D constrained arrays with an eye towards computing the capacity. We will now describe the adjacency construction in detail.

3.1.1 Adjacency Approach

The main idea behind the adjacency approach is to somehow reduce the 2-D problem to a 1-D framework and apply known techniques for determining the capacity. This type of construction is well-known [30], [12] in coding theory. The construction is outlined in the following steps:

Outline of the Adjacency Construction

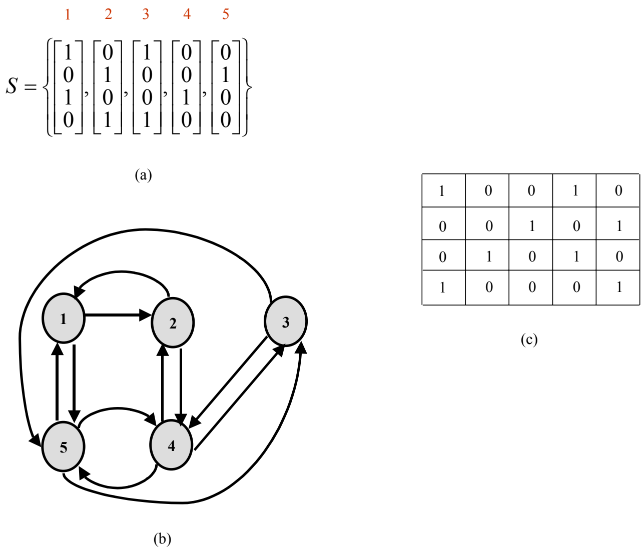
1. Construct an exhaustive set $\mathcal{S}$ of all column vectors of length m, satisfying the 1-D (d, k) constraint.

2. Let $\mathcal{V}$ denote the set of vertices of a graph $\mathcal{G}$ such that $|\mathcal{V}| = |\mathcal{S}|$. Construct a one-to-one function $f : \mathcal{S} \rightarrow \mathcal{V}$ for mapping each vector $x \in \mathcal{S}$ to a vertex $v \in \mathcal{V}$.

3. For any two vertices $v_i, v_j \in \mathcal{V}$, construct a directed edge e_{ij} iff $x_i \cdot x_j = 0$. The $(\cdot)$ operator is the dot product of the vectors x_i and x_j in the real field.

Following the above three steps, we have constructed a first-order Markov chain representing the graph of a (1,∞,d,k) RLL constraint. It is easy to observe that an $n-$step random walk on this graph will result in a 2-D binary array of size $m \times n$ that satisfies the constraints.

We will illustrate the procedure through an example. Suppose we want to construct a $(1, \infty, 1, 2)$ RLL array on a 4×5 grid, we first start with a set of all $(1, 2)$ column constrained sequences of length 4, as shown in Figure 11 (a). We associate each column constrained vector to a state in a graph. For instance, the column vector $[1001]^T$ is associated with state 1 and so on. Following Step 2 of the adjacency construction, we construct a graph, as shown in Figure 11 (b). It is easy to see that the dot product in Step 3 of the algorithm ensures that the row constraint is satisfied, i.e., we are forced to write a zero after writing a



$$S=\left\{\begin{array}{ccccc} 1 & 2 & 3 & 4 & 5 \\ \begin{bmatrix}1\\0\\1\\0\end{bmatrix}, & \begin{bmatrix}0\\1\\0\\1\end{bmatrix}, & \begin{bmatrix}1\\0\\0\\1\end{bmatrix}, & \begin{bmatrix}0\\0\\1\\0\end{bmatrix}, & \begin{bmatrix}0\\1\\0\\0\end{bmatrix}\end{array}\right\}$$

(a)

(b)

1	0	0	1	0
0	0	1	0	1
0	1	0	1	0
1	0	0	0	1

(c)

Figure 11: Schematic of the adjacency approach for constructing valid $(1, \infty, d, k)$ 2-D arrays on a 4×5 grid. (a) An exhaustive set of column vectors of length 4 satisfying $(1, 2)$ constraints. (b) Constructing a constrained graph $\mathcal{G}$ for generating valid 2-D arrays. (c) Schematic of a 2-D array obtained by doing a random walk along the transitions $3 \rightarrow 4 \rightarrow 5 \rightarrow 1 \rightarrow 2$.

one in any particular row. Finally, by doing a random walk on the graph, shown in Figure 11 (b), we can construct all valid 2-D arrays, like the one in Figure 11 (c).

The above construction is intuitively appealing and can be used to derive bounds for the channel capacity. From a practical perspective, generating 2-D arrays of an arbitrary large column length m is difficult due to the combinatorial explosion in the number of states as a function of m. We need other constructions that can overcome this spatial complexity. In the following subsection, we describe an equivalent construction called the iterative approach.

3.1.2 Iterative Approach

We highlight an alternative strategy for writing $(1{,}\infty, d, k)$ constrained arrays on an $m \times n$ rectangular grid. The key ideas behind the approach are as follows:

- A sequence of phrases (a phrase is a sequence of zeros followed by a one) is constructed by a random walk on a 1-D $(d{,}k)$ constrained graph $\mathcal{G}_{(d,k)}$, shown in Figure 7.

- For every '1' that occurs along the first column, a '0' is placed adjacent to it (i.e., along the next column). This ‘bit-stuffing’ ensures that the (1,∞) row constraint is satisfied.

- (d,k) sequences are constructed in the second column such that the overall column constraints are satisfied consistent with the ‘bit-stuffing’ in the previous step. This is accomplished by doing a random walk from an appropriate pre-determined state on $\mathcal{G}_{(d,k)}$ to satisfy the column constraints.

Unlike the previous method, the iterative approach does not require encoding in blocks of column vectors and is not restricted by combinatorial complexity of generating column vectors.

Outline of the Iterative Construction

There are a total of $k+1$ states in the graph $\mathcal{G}_{(d,k)}$, shown in Figure 7. Let us denote them sequentially as $0, 1, ..., k$.

1. We introduce the following variables:

   $c \equiv$ column number

   $f \equiv$ look ahead free position where a bit can be written (anticipation)

   $p \equiv$ position where a ‘1’ was most recently output (memory)

   $a \equiv$ look ahead occupied position of a ‘0’ (anticipation)

2. Along the first column, construct a sequence of phrases by doing a random walk on $\mathcal{G}_{(d,k)}$. For every ‘1’ that occurs in this column, stuff a ‘0’ adjacent to it.

   Initialize: $c = 2$;

3. Start in the c^{th} column. Set $f \leftarrow 1$; $p \leftarrow f$.

   - Locate a.
   - Starting from state 1, make a random walk of length $a - f$ on $\mathcal{G}_{(d,k)}$. Let this sequence be Y.

4. Update the following parameters:

   - $p \leftarrow$ position of the most recent '1' output in Y.
   - $f \leftarrow$ free position where a bit can be written.
   - $a \leftarrow$ immediate position where there is an occupied '0'.

5. If $f - p < d$,

   - Stuff $d - f + p$ zeros to ensure that the d constraint is satisfied. Update f.
   - Perform a random walk of length $a - f$ starting from the state $s = d$. Let this sequence be Y. (Note that starting from state d ensures that the k constraint is not violated.)

   Else

   - Make a random walk of length $a - f$ starting from the state $s = f - p$ on $\mathcal{G}_{(d,k)}$. Let this sequence be Y. Starting from this state ensures that the k constraint is satisfied.

   End

6. Loop over Step 4 until all the m rows are filled.

7. For every '1' occurring in this column, stuff a '0' adjacent to it.

8. $c \leftarrow c + 1$.

9. Loop over Step 3 until all the n columns are filled.

10. Stop.

The iterative algorithm clearly precludes the necessity of storing random vectors of length m and building a complicated graph for generating the 2-D arrays. This algorithm constructs a 2-D $(1, \infty, d, k)$ code.

3.2 Capacity Bounds

The constructions presented in the previous section are helpful for analyzing the information-theoretic rate of the constraints. In this section, we derive the capacity bounds for $(1, \infty, d, k)$ constraints.

Recall from Step 2 of the adjacency construction, the adjacency matrix $\mathcal{A}_{(m)} = [a_{ij}]$ (subscript m indicates that the adjacency matrix is a function of the column length m) is such that for any two column vectors x_i and x_j, $a_{ij} = 1$ iff $\boldsymbol{x_i . x_j} = 0$. Here the dot product is on the real field.

We note the following observations with respect to the $(1, \infty, d, k)$ RLL constraints.

Observation 3.1. *The following statements are trivially true:*

1. $|\mathcal{V}|$ *is at least* $2^{mC_{(d,k)}}$

2. *The adjacency matrix* $\mathcal{A}_{(m)}$ *is always symmetric.*

The first part of Observation 3.1 indicates that we have an exponential increase in the number of states of the Markov chain as a function of m. The second part of Observation 3.1 is helpful for deriving a lower bound for the capacity of these constraints. Calkin and Wilf [12] obtained a tight lower bound on the asymptotic growth rate of largest eigenvalue of $\mathcal{A}_{(m)}$ for the $(1, \infty, 1, \infty)$, i.e., the hard-square constraint based on the maximum principle. We sketch the proof of their result in the following theorem. We note that this result holds true for the $(1, \infty, d, k)$ case as well.

Theorem 3.1. *The capacity of* $(1, \infty, d, k)$ *constrained channels is lower bounded by*

$$C_{(1,\infty,d,k)} \geq \log_2 \left(\sqrt[p]{\tfrac{\lambda_{p+2q}}{\lambda_{2q}}} \right),$$

where λ_m *is the largest eigenvalue of the adjacency matrix formed by column vectors of length* m.

Proof. Let $g(m, n, x)$ be the number of binary arrays of size $m \times n$ with the rightmost column x. After one step on the Markov chain $\mathcal{G}$, we have

$$g(m, n+1, y) = \sum_{x \in S} \mathcal{A}_{(m)}(y, :) g(m, n, x) \qquad (7)$$

where, $\mathcal{A}_{(m)}(y,:)$ denotes the row vector of $\mathcal{A}_{(m)}$ corresponding to the column vector y.

After a total of n steps on the graph $\mathcal{G}$, the total number of binary arrays on an $m \times n$ grid is

$$g(m,n) = u^T A_{(m)}^n u, \tag{8}$$

where u is a column vector of all ones.

The adjacency matrix is real and symmetric from Observation 3.1. Thus, for any positive integer p, $\mathcal{A}_{(m)}^p = \mathcal{A}_{(p)}^m$. Using the maximum principle we have,

$$\lambda_m^p = \frac{u^T \mathcal{A}_{(m)}^p u}{u^T u}, \tag{9}$$

where λ_m^p is the largest eigenvalue of $\mathcal{A}_{(m)}^p$.

Since (9) holds true for any linear transformation of the vector u, for any integer q, we can rewrite (9) as

$$\lambda_m^p = \frac{(\mathcal{A}_{(m)}^q u)^T \mathcal{A}_{(m)}^p \mathcal{A}_{(m)}^q u}{(\mathcal{A}_{(m)}^q u)^T \mathcal{A}_{(m)}^q u} = \frac{u^T \mathcal{A}_{(m)}^{p+2q} u}{u^T \mathcal{A}_{(m)}^{2q} u}. \tag{10}$$

The above equation holds true for every positive integer q. Since $\lim_{m\to\infty}(\lambda_m)^{\frac{1}{m}}$ exists [27], (10) can be simplified as,

$$\lim_{m\to\infty}(\lambda_m)^{\frac{1}{m}} \geq \sqrt[p]{\frac{\lambda_{p+2q}}{\lambda_{2q}}}. \tag{11}$$

We have derived a lower bound for the largest eigenvalue of the adjacency matrix $\mathcal{A}_{(m)}$ when $m \to \infty$. From Remark 2.1, the capacity of a 1-D RLL constrained channel is $\log_2(\lambda)$, where λ is the largest eigenvalue of the adjacency matrix of the constrained graph. Using this remark, the theorem follows. □

To obtain an upper bound for the capacity, we invoke the following definition of a Markov process [15].

Consider a stochastic process $X_1, X_2, ..., X_n$ indexed by a sequence of discrete random variables. Let $P(X_1 X_2 ... X_n)$ denote the joint probability distribution of the sequence of random variables.

Definition 3.1. *A discrete stochastic process $X_1, X_2, ...$ is said to be a first-order Markov process if, for all $n = 1, 2, ...$*

$$P(X_{n+1} = x_{n+1}|X_n = x_n, X_{n-1} = x_{n-1}, ..., X_1 = x_1) = P(X_{n+1} = x_{n+1}|X_n = x_n).$$

In the following theorem, we derive an expression for computing an upper bound for the capacity.

Theorem 3.2. *The capacity of $(1, \infty, d, k)$ constrained channels is upper bounded by*

$$C_{(1,\infty,d,k)} \leq \lim_{m\to\infty} \tfrac{1}{m} H(X_2^{(m)}|X_1^{(m)}),$$

where $H(X_2^{(m)}|X_1^{(m)})$ is the conditional entropy of the Markov chain $\mathcal{G}$.

Proof. From the adjacency construction, the graph $\mathcal{G}$ is a first-order Markov process. The number of states in the Markov chain is a function of the column length m. For a given m, the conditional probability $P(X_{n+1}^{(m)}|X_n^{(m)})$ is invariant with time, i.e., the probability of being in a certain state at time $n+1$ given the previous state at time n is independent of time. Choosing a stationary probability distribution for the states, we have $P(X_{n+1}^{(m)}) = P(X_n^{(m)})$.

Computing the maximum entropy rate for the stationary Markov process representing $\mathcal{G}$,

$$C_{(1,\infty,d,k)} \leq \lim_{m,n\to\infty} \sup_{P(X_1^{(m)} X_2^{(m)} ... X_n^{(m)})} \frac{1}{m} H(X_1^{(m)}, X_2^{(m)}, ..., X_n^{(m)}). \tag{12}$$

Using the chain rule for entropy and expanding the joint entropy term in (12),

$$C_{(1,\infty,d,k)} \leq \lim_{m,n\to\infty} \frac{1}{m} \sup_{P(X_1^{(m)} X_2^{(m)} ... X_n^{(m)})} \sum_{i=1}^{n} H(X_i^{(m)}|X_{i-1}^{(m)}, ..., X_1^{(m)}). \tag{13}$$

Using stationarity and Markovity of the underlying random process, we can simplify (13) as

$$C_{(1,\infty,d,k)} \leq \lim_{m\to\infty} \sup_{P(X_2^{(m)}|X_1^{(m)})} \frac{1}{m} H(X_2^{(m)}|X_1^{(m)}). \tag{14}$$

□

It should be noted that the above result in Theorem 3.2 holds true for asymptotic values of m. Computing the conditional entropy for finite m is always an estimate.

3.3 Numerical Results

We will compute numerical results based on the theorems we presented in the previous section. Table 1 shows the computed lower bounds for a few constraints. For example, to compute the bound for $(1, \infty, 3, \infty)$ constraints, we choose $p = 3, q = 2$. The largest eigenvalue of the adjacency matrix obtained by splicing all $(3, \infty)$ column vectors of length $p + 2q = 7$ was evaluated as 10.5334. Similarly, the largest eigenvalue corresponding to the adjacency matrix obtained by splicing all $(3, \infty)$ column vectors of length $2q = 4$ was evaluated as 4.2361. Using Theorem 3.1, we compute $C_{(1,\infty,3,\infty)} \geq 0.4381$.

Table 1: Lower bounds for $(1, \infty, d, k)$ constraints

$(1, \infty, d, k)$	`Lower Bound`
$(1, \infty, 2, \infty)$	0.4995
$(1, \infty, 3, \infty)$	0.4381
$(1, \infty, 1, 4)$	0.4094
$(1, \infty, 2, 5)$	0.2848

Table 2 shows the upper bound computations for the same constraints considered in Table 1. The upper bounds were evaluated for a column length of 7 in all cases.

Table 2: Upper bound estimates for $(1, \infty, d, k)$ constraints

$(1, \infty, d, k)$	`Upper Bound Estimate`
$(1, \infty, 2, \infty)$	0.5263
$(1, \infty, 3, \infty)$	0.4853
$(1, \infty, 1, 4)$	0.5068
$(1, \infty, 3, 5)$	0.4196

3.4 Summary

In this chapter, we presented two simple algorithms for constructing arrays satisfying $(1, \infty, d, k)$ RLL constraints. The class of $(1, \infty, d, k)$ constraints are relatively the simple to analyze. We presented bounds for the capacity of the constraints by applying existing approaches. The computation of the capacity bounds is useful for a code designer to know the limits for achievable code rates. In the next chapter, we examine other RLL constraints, derive capacity bounds and present coding algorithms.

CHAPTER IV

CAPACITY BOUNDS AND CODING ALGORITHMS FOR ASYMMETRIC (D, ∞) AND $(0, K)$ RUNLENGTH-LIMITED CONSTRAINED CHANNELS

In the previous chapter, we presented two simple algorithms for generating $(1, \infty, d, k)$ RLL arrays and derived bounds for the capacity of $(1, \infty, d, k)$ constrained channels. However, the constructions and bounding techniques that we presented in the earlier chapter cannot be straightforwardly generalized for analyzing the capacity of other RLL constraints. The lack of graph-based structures makes this combinatorial problem challenging.

The capacity of 2-D asymmetric RLL constrained channels is hardly known. In a few cases, Kato and Zeger point out the existence of positive/zero [27] capacity regions. There are very few efficient coding algorithms for constructing 2-D arrays. Siegel and Wolf [41] developed the bit-stuffing bounds for 2-D symmetric (d, ∞) and $(0, k)$ RLL constraints. Their bound is constructive for obtaining variable-rate coded arrays satisfying the constraints. Roth, Siegel, and Wolf [39] analyzed the bit-stuffing algorithm in detail for the hard-square constraint. Improved bit-stuffing bounds for symmetric (d, ∞) constraints are reported in [21]. In this chapter, we examine the capacity of two classes of asymmetric RLL constraints and present coding algorithms for mapping the raw bits to 2-D RLL arrays.

This chapter is organized as follows. In section 4.1, we present two tiling algorithms for constructing 2-D $(d_1, \infty, d_2, \infty)$ RLL arrays. We examine the Hamming weight structure of these arrays and derive capacity bounds based on a combination of probability and combinatorial approaches. In section 4.3, we present an algorithm for constructing sequentially nested block codes for $(d_1, \infty, d_2, \infty)$ constraints and present numerical results in section 4.4. In section 4.5, we present an algorithm for constructing $(0, k_1, 0, k_2)$ RLL arrays. We derive bounds for the capacity of $(0, k_1, 0, k_2)$ constrained channels in section 4.6 and present a coding algorithm in section 4.7. Numerical results for the capacity of $(0, k_1, 0, k_2)$ constrained channels are discussed in section 4.8, followed by a chapter summary in section 4.9.

4.1 *Asymmetric* (d, ∞) *Constraints*

In this section, we describe two simple algorithms for constructing an ensemble of all 2-D binary arrays satisfying the $(d_1, \infty, d_2, \infty)$ RLL constraints on an $m \times n$ rectangular grid. In the first approach, (d_2, ∞) column vectors are spliced according to certain merging conditions so that the resulting 2-D array satisfies the overall constraints, in a block by block construction. In the second approach, RLL arrays are generated by sequentially writing column constrained sequences resulting from a random walk on the column constrained graph such that the overall 2-D constraints are satisfied. We will explain the two algorithms in detail and also review the bit-stuffing technique [41] for purposes of comparison.

4.1.1 Tiling Algorithm-A

We first present a few facts for sequentially writing binary patterns satisfying the 2-D constraints. The algorithm easily follows from these facts. Throughout this chapter, we refer to a valid array as a 2-D binary array satisfying the constraints on an $m \times n$ grid.

Let $\mathcal{S}_m^{(d_2,\infty)}$ denote the set of all (d_2,∞) column vectors of length m. Every column in a valid 2-D array is an element of the set $\mathcal{S}_m^{(d_2,\infty)}$. Let $z_i^{(m)} \in \mathcal{S}_m^{(d_2,\infty)}$ represent the i^{th} column vector of a valid array.

In Fact 4.1, we prove that the sequence of column vectors of a valid 2-D array has a memory d_1. The proof of this property follows from a straightforward extension of the ideas presented in [30] and is helpful for analyzing the structure of the 2-D constraints.

Fact 4.1. *The vector sequence* $\{z_i^{(m)}\}_{i=1}^n$ *over the alphabet set* $\mathcal{S}_m^{(d_2,\infty)}$ *has a finite memory* d_1*.*

Proof. Let $u = z_1 z_2 ... z_n$ be a block of column vectors whose block length is greater than d_1. Let us imagine a device that reads the column vectors sequentially from z_1 through z_n to

[1] A derivation on the exact size of the set $\mathcal{S}_m^{(d_2,\infty)}$ based on combinatorics is outlined in the next section.

ascertain if u satisfies the 2-D constraints. At any point i, the device will need a memory of past d_1 columns. Thus the vector sequence formed by $\{z_i^{(m)}\}_{i=1}^n$ is a d_1-step memory process. □

The following lemma provides the necessary and sufficient conditions for creating valid 2-D arrays.

Lemma 4.1. *Let $z_i^{(m)}, z_j^{(m)} \in \mathcal{S}_m^{(d_2,\infty)}$ denote any two arbitrary column vectors on an $m \times n$ grid satisfying the constraints. The vectors $z_i^{(m)}$ and $z_j^{(m)}$ $(i < j)$ satisfy the set of orthogonality conditions iff*

$$z_i^{(m)} \cdot z_j^{(m)} = 0, 1 \leq i \leq n \& \;\; i+1 \leq j \leq i+d_1, \tag{15}$$

where $(\cdot)$ is the usual vector inner product on the real field.

Proof. We prove that these conditions are necessary and sufficient for sequentially writing $\{z_i^{(m)}\}_{i=1}^n$. To prove that these conditions are necessary, let us assume that one of these conditions fail. Without any loss of generality, suppose that for some fixed i and j in the ranges specified above, $z_j^{(m)} \cdot z_i^{(m)} \neq 0$. This implies that there is a '1' in column i that is not at least d_1 away from a '1' in the j^{th} column. Hence, the horizontal constraints are violated.

To prove that these conditions are sufficient, recall from Fact 4.1 that $z_j^{(m)}$ depends only on the previous d_1 columns. In other words, for every '1' occurring in the column vector $z_j^{(m)}$, there must have been at least d_1 zeros to its left. Thus, for a fixed i, $z_j^{(m)} \cdot z_i^{(m)} = 0$ for values of j in the range $i+1 \leq j \leq i+d_1$.

□

Fact 4.1 and Lemma 4.1 are summarized in a simple algorithm as follows:

Outline of the Algorithm - A

1. *Initialize:* $j = 2$. Choose any random vector $z_1^{(m)} \in \mathcal{S}_m^{(d_2,\infty)}$ for the first column.

2. Start from the j^{th} column. Choose a vector $z_j^{(m)} \in \mathcal{S}_m^{(d_2,\infty)}$ such that $z_j^{(m)} \cdot z_i^{(m)} = 0$ and $\max\{1, j - d_1\} \leq i \leq j - 1$.

3. $j \leftarrow j + 1$.

4. Loop over Step 2 until all the columns are filled.

The set of merging conditions in Lemma 4.1 leading to tiling Algorithm-A can be visualized as concatenating a set of valid rectangular arrays of size $m \times d_1$ so that the resulting array satisfies the overall constraints. This procedure is somewhat similar to the adjacency construction described in section 3.1.1.

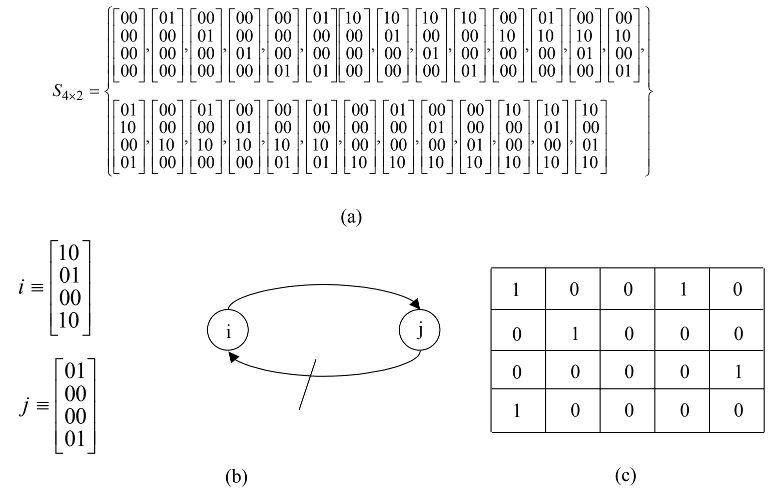
In Figure 12, we illustrate the construction of $(2, \infty, 2, \infty)$ arrays on a 4×5 grid. First, we form a set of all 4×2 arrays satisfying the $(2, \infty, 2, \infty)$ constraint. Each such array represents the state of a directed graph $\mathcal{G}$. We have not explicitly illustrated the graph $\mathcal{G}$ since the number of states of this graph is 27, corresponding to the number of all 4×2 arrays, as shown in Figure 12 (a). There is a directed edge from state i to state j on $\mathcal{G}$ if the merger of two arrays is a valid array. It is an important point to note that the adjacency matrix of $\mathcal{G}$ is no longer symmetric. This observation is illustrated in Figure 12 (b). Hence, Theorem 3.1 does not hold for these constraints.

Before concluding this subsection, we would like to discuss the complexity of Algorithm-A. Note that the time complexity of the algorithm is proportional to the number of columns sequentially written. Calculating the conditions specified by Lemma 4.1 are the only computations needed. However, the space complexity of this approach is exponential in m since we need to store all of the column vectors of length m.

4.1.2 Tiling Algorithm - B

In this subsection, we highlight a second tiling algorithm for generating valid 2-D arrays satisfying (d_1,∞,d_2,∞) RLL constraints. The key ideas behind this approach are summarized as follows:

- A sequence of phrases is constructed along the first column by doing a random walk on a 1-D (d_2, ∞) constrained graph $\mathcal{G}_{(d_2,\infty)}$.



$$S_{4\times 2} = \left\{ \begin{matrix} \begin{bmatrix}00\\00\\00\\00\end{bmatrix}, \begin{bmatrix}01\\00\\00\\00\end{bmatrix}, \begin{bmatrix}00\\01\\00\\00\end{bmatrix}, \begin{bmatrix}00\\00\\01\\00\end{bmatrix}, \begin{bmatrix}00\\00\\00\\01\end{bmatrix}, \begin{bmatrix}01\\00\\00\\01\end{bmatrix} \begin{bmatrix}10\\00\\00\\00\end{bmatrix}, \begin{bmatrix}10\\01\\00\\00\end{bmatrix}, \begin{bmatrix}10\\00\\01\\00\end{bmatrix}, \begin{bmatrix}10\\00\\00\\01\end{bmatrix}, \begin{bmatrix}00\\10\\00\\00\end{bmatrix}, \begin{bmatrix}01\\10\\00\\00\end{bmatrix}, \begin{bmatrix}00\\10\\01\\00\end{bmatrix}, \begin{bmatrix}00\\10\\00\\01\end{bmatrix}, \\ \begin{bmatrix}01\\10\\00\\01\end{bmatrix}, \begin{bmatrix}00\\00\\10\\00\end{bmatrix}, \begin{bmatrix}01\\00\\10\\00\end{bmatrix}, \begin{bmatrix}00\\01\\10\\00\end{bmatrix}, \begin{bmatrix}00\\00\\10\\01\end{bmatrix}, \begin{bmatrix}01\\00\\10\\01\end{bmatrix}, \begin{bmatrix}00\\00\\00\\10\end{bmatrix}, \begin{bmatrix}01\\00\\00\\10\end{bmatrix}, \begin{bmatrix}00\\01\\00\\10\end{bmatrix}, \begin{bmatrix}00\\00\\01\\10\end{bmatrix}, \begin{bmatrix}10\\00\\00\\10\end{bmatrix}, \begin{bmatrix}10\\01\\00\\10\end{bmatrix}, \begin{bmatrix}10\\00\\01\\10\end{bmatrix} \end{matrix} \right\}$$

(a)

$$i \equiv \begin{bmatrix}10\\01\\00\\10\end{bmatrix}$$

$$j \equiv \begin{bmatrix}01\\00\\00\\01\end{bmatrix}$$

(b)

1	0	0	1	0
0	1	0	0	0
0	0	0	0	1
1	0	0	0	0

(c)

Figure 12: Schematic of the adjacency approach for constructing valid $(d_1, \infty, d_2, \infty)$ 2-D arrays on a 4×5 grid. (a) Illustration of an exhaustive set of 2-D arrays of size 4×2, satisfying the $(2, \infty, 2, \infty)$ constraint. (b) Illustration of a case where directed edges between any two states need not be reflexive. In this example we can go from state i to state j, but not the other way. (c) Schematic of a 2-D array satisfying the $(2, \infty, 2, \infty)$ constraint.

- For every '1' that occurs along the first column, d_1 zeros are placed adjacent to the rightside of the current column. This 'bit stuffing' ensures that the (d_1, ∞) row constraint is satisfied.

- (d_2,∞) sequences are constructed in the second column consistent with the 'bit stuffing' in the previous step by doing a random walk on $\mathcal{G}_{(d_2,\infty)}$. The above steps are iteratively repeated for the subsequent columns.

Tiling scheme-B does not require writing sequences in blocks of column vectors and circumvents the need for storing any column vectors. The above ideas are summarized in an algorithm below.

Consider the 1-D (d, ∞) constrained graph, as shown in Figure 13.

Outline of Algorithm - B

1. *Initialize:* $i = 1$, $d = d_2$.

2. Start from the i^{th} column. Along the vacant spaces in the i^{th} column, do a random walk on $\mathcal{G}_{(d_2,\infty)}$, shown in Figure 13, by starting from the last state, i.e., d_2.

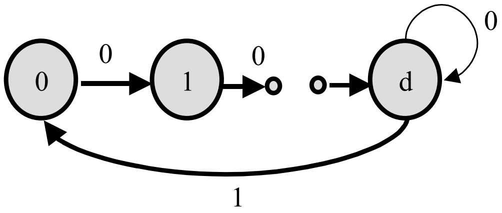


Figure 13: Constrained graph $\mathcal{G}_{(d,\infty)}$.

3. Identify the row locations where a '1' occurs in the i^{th} column. Stuff d_1 zeros horizontally along the next d_1 columns of the row locations containing the ones in the i^{th} column.

4. $i \leftarrow i + 1$.

5. Loop over Step 2 until all the columns are filled.

It is clear that the vacant spaces along the first column will be the entire column length m, but for the subsequent columns, the vacant space is dictated by the zero bit-stuffing because of the previous d_1 columns. It is easy to observe that the random walk process in Algorithm-B is shift invariant. Further, this is not a maxentropic random walk because of the zero bit stuffing induced from the previous d_1 columns.

In Lemma 4.2, we show that both tiling schemes are equivalent. In other words, both schemes produce valid 2-D arrays.

Lemma 4.2. *Tiling Algorithms A and Algorithm-B are equivalent.*

Proof. We will prove this by using induction on the number of columns.

Case $n = 1$:

When the number of columns is one, Algorithm-A chooses a valid column vector of length m from the set $\mathcal{S}_m^{(d_2,\infty)}$. This is identical to a random walk of length m according to Algorithm-B. Thus, the result is verified to be trivially true for $n = 1$.

Case $n = p$:

Let us assume that the result holds true for $n = p$ and look into the case for $n = p + 1$.

Case $n = p + 1$:

At this point, recall that Algorithm-A picks a column vector $z_n^{(m)} \in \mathcal{S}_m^{(d_2,\infty)}$ according to the conditions specified in Lemma 4.1. To verify the equivalence, we note that writing sequences in the n^{th} column according to Algorithm-B satisfies the orthogonality conditions because of the horizontal bit stuffing. Furthermore, the random walk process starting from state d_2 in Algorithm-B ensures that no sequences are lost. Thus, the ensemble of all arrays produced by both algorithms is the same. Since both of these schemes are sequential in nature, they must have the same maxentropic rate.

□

4.1.3 Bit-stuffing Algorithm

In this subsection, we discuss the bit-stuffing algorithm by Siegel and Wolf [41] for constructing symmetric (d, ∞) constrained arrays. The bit-stuffing encoder converts an arbitrary binary sequence into a sequence of statistically independent $p-$biased bits. The probability of a '1' in the converted sequence is p. This conversion is accomplished by using a distribution transformer at a rate penalty of $h(p)$, where $h(p)$ is the binary entropy function [15]. Using a $p-$biased source, the source statistics can be best adjusted for matching the channel statistics and maximizing the average code rate.

The basic idea in the bit-stuffing algorithm is to write $p-$biased bits along successive diagonals of a rectangular lattice. The following steps illustrate the key encoding ideas of the bit-stuffing algorithm.

- Whenever a '1' occurs in the the $p-$biased source sequence, d zeros are inserted to the right of it and below it.
- In writing the $p-$biased sequence down the diagonals, any position already occupied by a previous stuffed zero is skipped.

Decoding is tuned to the encoding principle. The bits are read successively from the array. The zero bits that are stuffed are deleted and ignored. The remaining bits are from the $p-$ biased source. These bits are invertibly mapped back to the original data bits.

Halevy et al. [21] presented a detailed analysis on the coding rate of the bit-stuffing algorithm for symmetric (d, ∞) constraints and generalized this idea for hexagonal (d, ∞)

constraints.

In the next section, we derive bounds for the capacity of asymmetric (d, ∞) constraints based on the tiling algorithms presented in sections 4.1.1 and 4.1.2. We later compare our results against known bit-stuffing bounds.

4.2 Capacity Bounds for Asymmetric (d, ∞) *Constraints*

The tiling schemes presented in the previous section are useful for enumerating all valid 2-D arrays on a rectangular grid. Before we begin to derive the bounds [46], [43] we first analyze a few properties concerning the size of the set of all 1-D (d,∞) sequences of length m. This result will be used later in the derivations.

In Lemma 4.3, we obtain the maximum and minimum Hamming weights of a 1-D RLL sequence of length m.

Lemma 4.3. *The maximum and minimum Hamming weights of a* (d, k) *RLL sequence of length* m *are*

$$w_{max} = \lfloor \tfrac{m+d}{d+1} \rfloor, w_{min} = \lfloor \tfrac{m}{k+1} \rfloor.$$

Proof. The code has to be length m. To maximize the number of ones, we have to ensure that the runlength of zeros between any two consecutive ones is minimum, i.e., we maintain the runlength of zeros to be d. Consider the sequence $10^{a_1}10^{a_2}......0^{a_p}1$, where, each $a_i = d$ for $1 \leq i \leq w_{max} - 1$. We have

$$m - w_{max} - (w_{max} - 1)d \geq 0. \tag{16}$$

Seeking the smallest integer solution to (16), w_{max} is verified.

To obtain the minimum Hamming weight, we should maintain a runlength of k consecutive zeros between two ones. Consider the sequences of the type $0^{a_1}10^{a_2}1......0^{a_p}1$ and $10^{a_1}10^{a_2}......10^{a_p}$, where each $a_i = k$ for a string of zeros between two consecutive ones and in-between d and k at the leading and trailing ends. Using this observation, we have

$$m - w_{min} - w_{min}k \geq 0 \tag{17}$$

Seeking the nearest integer solution to (17), w_{min} is verified. It is trivial to see that $w_{min} = 0$ when $k = \infty$. $\square$

Aside: w_{max} is the total depth of a simple tree based algorithm for generating an exhaustive set of (d, ∞) sequences of length m.

In the following theorem, we derive an expression for the size of the set of all 1-D (d, ∞) RLL sequences of length m.

Theorem 4.1. *The size of the set of all (d, ∞) vectors of length m is*

$$|\mathcal{S}_m^{(d_2,\infty)}| = \sum_{w=0}^{\lfloor \frac{m+d}{d+1} \rfloor} T_w(m - w - (w-1)d); T_w(x) = \tfrac{(w+x)!}{x!w!}, x \geq 0.$$

Proof. For a (d,∞) sequence, $w_{min} = 0$ since $k = \infty$. Determining the size of the set of all vectors of length m with a Hamming weight w can be framed as the number of integer solutions (if they exist) to

$$\sum_{i=1}^{w-1} b_i = m - w - (w-1)d, \tag{18}$$

such that $b_i \geq 0$. The number of integer solutions to (18) is given by $T_w(m - w - (w-1)d)$. The total size $|S_m|$ is now the sum over all possible Hamming weights. Using Lemma 4.3, the theorem follows. $\square$

Using Theorem 4.1, we can obtain the Hamming weight distribution of the set of all (d, ∞) sequences of length m. Using Theorem 4.1, we have an alternative formula for computing the capacity of 1-D constraints in a combinatorial way.

Remark 4.1. *The capacity of 1-D RLL (d,∞) sequences can be numerically computed as*

$$C_{(d,\infty)} = \lim_{m \to \infty} \tfrac{\log_2(|\mathcal{S}_m^{(d_2,\infty)}|)}{m}$$

We note that Lemma 4.3 and Theorem 4.1 will be the key tools for obtaining the combinatorial lower bounds for the capacity of the 2-D constraints.

Before embarking on the proofs, we would like to point out some observations on the tiling algorithms. Let $Z_i^{(m)}$ denote the random variable representing the i^{th} column vector. Clearly, the index i is also the time index since we are sequentially writing the columns.

Proposition 4.1. *The probability* $P_i(Z_i^{(m)} = z_i^{(m)}|Z_{i-1}^{(m)} = z_{i-1}^{(m)}, Z_{i-2}^{(m)} = z_{i-2}^{(m)}, ..., Z_1^{(m)} = z_1^{(m)})$=$P(Z_i^{(m)} = z_i^{(m)}|Z_{i-1}^{(m)} = z_{i-1}^{(m)}, Z_{i-2}^{(m)} = z_{i-2}^{(m)}, ..., Z_{i-d_1}^{(m)} = z_{i-d_1}^{(m)})$ *and is independent of* i *for* $i \geq d_1 + 1$.

Proof. Consider the conditional probability of the random variable $Z_i^{(m)}$ conditioned on all the previous columns. As usual, $z_i^{(m)}$ is the value that the random variable $Z_i^{(m)}$ can take.

$$P_i(Z_i^{(m)} = z_i^{(m)}|Z_{i-1}^{(m)} = z_{i-1}^{(m)}, ..., Z_1^{(m)} = z_1^{(m)}) = \frac{P_i(Z_i^{(m)} = z_i^{(m)}, Z_{i-1}^{(m)} = z_{i-1}^{(m)}, ..., Z_1^{(m)} = z_1^{(m)})}{P_i(Z_{i-1}^{(m)} = z_{i-1}^{(m)}, ..., Z_1^{(m)} = z_1^{(m)})}. \tag{19}$$

From Fact 4.1, at any step i, it suffices to observe the random variables from $Z_{i-1}^{(m)}$ to $Z_{i-d_1}^{(m)}$. Hence, we can rewrite equation (19) as

$$P_i(Z_i^{(m)} = z_i^{(m)}|Z_{i-1}^{(m)} = z_{i-1}^{(m)}, ..., Z_1^{(m)} = z_1^{(m)}) = \frac{P_i(Z_i^{(m)} = z_i^{(m)}, Z_{i-1}^{(m)} = z_{i-1}^{(m)}, ..., Z_{i-d_1}^{(m)} = z_{i-d_1}^{(m)})}{P_i(Z_{i-1}^{(m)} = z_{i-1}^{(m)}, ..., Z_{i-d_1}^{(m)} = z_{i-d_1}^{(m)})}. \tag{20}$$

Given a valid block $z_{i-d_1}^{(m)} z_{i-d_1+1}^{(m)} ... z_{i-1}^{(m)}$, $Z_i^{(m)}$ is independent of i. Thus, we can write the conditional probability mass function (pmf) as

$$P(Z_{d_1+1}^{(m)}|Z_{d_1}^{(m)}, ..., Z_1^{(m)}) = P(Z_n^{(m)}|Z_{n-1}^{(m)}, ..., Z_{n-d_1}^{(m)}). \tag{21}$$

Equation (21) implies that the process of writing any column at step i conditioned on the previous d_1 columns is a homogenous d^{th} order shift-invariant Markov process. □

Using Proposition 4.1, we can derive an upper bound for the capacity.

Theorem 4.2. *The capacity of* $(d_1, \infty, d_2, \infty)$ *constrained channels is upper bounded by*

$$C_{(d_1,\infty,d_2,\infty)} \leq \lim_{m\to\infty} \tfrac{1}{m} H(Z_{d_1+1}^{(m)}|Z_{d_1}^{(m)}, ..., Z_1^{(m)}),$$

where $H(Z_{d_1+1}^{(m)}|Z_{d_1}^{(m)}, ..., Z_1^{(m)})$ *is the conditional entropy of the* $d_1 + 1^{th}$ *column conditioned on the previous* d_1 *columns.*

Proof. Let us consider tiling Algorithm-A. The maximum information rate is given by

$$R_A = \lim_{m,n\to\infty} \frac{\sum_{i=1}^{n} H(Z_i^{(m)}|Z_{i-1}^{(m)}, ..., Z_1^{(m)})}{mn}. \tag{22}$$

From equation (21) in Proposition 4.1, for $i \geq d_1 + 1$, we infer that

$$H(Z_i^{(m)}|Z_{i-1}^{(m)} = z_{i-1}^{(m)}, ..., Z_1^{(m)} = z_1^{(m)}) = H(Z_i^{(m)}|Z_{i-1}^{(m)} = z_{i-1}^{(m)}, ..., Z_{i-d_1}^{(m)} = z_{i-d_1}^{(m)}) = \log_2(|\mathcal{U}|), \tag{23}$$

where $\mathcal{U}$ is the set of all valid column vectors that satisfy the orthogonality conditions in Lemma 4.1 such that each vector in that set can be concatenated with a previous valid block. However, the unresolved step is the joint pmf of a block of $m \times d_1$ column vectors.

At this point, it must be noted that the joint pmf of a set of random variables representing a block of d_1 column vectors is time varying. In other words, for $i \neq j$

$$P_i(Z_i^{(m)} Z_{i-1}^{(m)} ... Z_{i-d_1}^{(m)}) \neq P_j(Z_j^{(m)} Z_{j-1}^{(m)} ... Z_{j-d_1}^{(m)}). \tag{24}$$

Consider the least upper bound of the sequence of conditional entropies in the limiting case.

$$H(Z_{d_1+1}^{(m)}|Z_{d_1}^{(m)}, ..., Z_1^{(m)}) = \lim_{n\to\infty} \sup_{P_n(Z_n^{(m)} Z_{n-1}^{(m)} ... Z_{n-d_1}^{(m)})} H(Z_n^{(m)}|Z_{n-1}^{(m)}, ..., Z_{n-d_1}^{(m)}) \tag{25}$$

Using (23) and (25), we can bound (22) as

$$R_A \leq \lim_{m\to\infty} \frac{1}{m} \lim_{n\to\infty} \frac{H(Z_1^{(m)}) + ... + H(Z_{d_1}^{(m)}|Z_{d_1-1}^{(m)}, ..., Z_1^{(m)}) + (n - d_1) H(Z_{d_1+1}^{(m)}|Z_{d_1}^{(m)}, ..., Z_1^{(m)})}{n}. \tag{26}$$

Evaluating $R_A^{(m)}$ for a large value m, we get $C \leq \frac{1}{m} H(Z_{d_1+1}^{(m)}|Z_{d_1}^{(m)}, ..., Z_1^{(m)})$. Using this method, we can numerically obtain an estimate for the capacity upper bound of the constraints. □

Corollary 4.1. *The following bound is trivially true:*

$$\tfrac{m}{m+d_2} R_A^{(m)} \leq C \leq R_A^{(m)}, \text{ where } R_A^{(m)} = \tfrac{1}{m} H(Z_{d_1+1}^{(m)}|Z_{d_1}^{(m)}, ..., Z_1^{(m)}).$$

Proof. Let $R_A^{(m)}$ be the maximum information rate on an $(m \times \infty)$ grid resulting from Algorithm-A. By stuffing d_2 rows of all zero vectors along the bottom rows, we can always tile the entire plane. The information rate resulting from the stuffing of d_2 rows of all zero vectors can be obtained as $\frac{m}{m+d_2} R_A^{(m)}$. This result serves as a simple lower bound, proving the corollary.

□

Our next step is to get a closed-form formula for the joint pmf and the conditional entropy in terms of the Hamming weight distribution. From the tiling algorithms, for every '1' that occurs in a particular column, d_1 zeros are stuffed horizontally to the right to satisfy the row constraint. This creates vacant spaces along the successive columns.

The sequences that can be written along the vacant spaces of the columns can be categorized into the following two cases.

Case 1: In this case, we allow (d_2, ∞) constrained sequences to be written along the vacant spaces of the columns. The idea is illustrated in Figure 14.

Case 2: In this case, we place sequences $\notin (d_2, \infty)$ along the vacant spaces. But, the resulting array satisfies the overall (d_2, ∞) column constraints. This idea is illustrated in Figure 15.

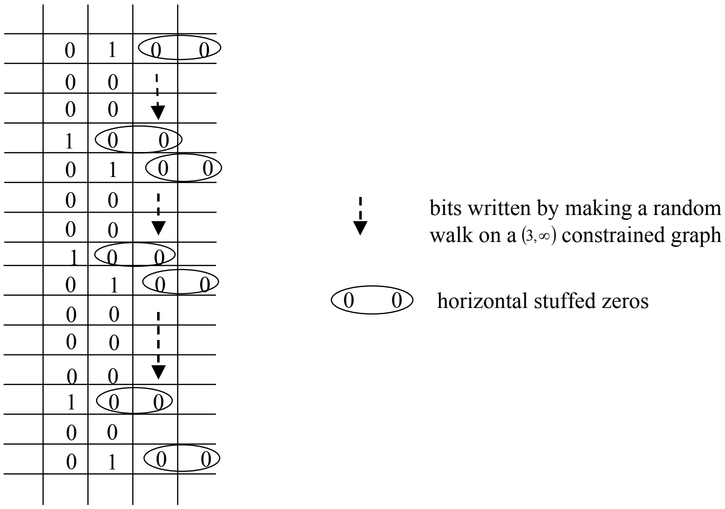

bits written by making a random walk on a $(3,\infty)$ constrained graph
horizontal stuffed zeros

Figure 14: Illustration of the configuration for case 1 for writing $(2, \infty, 3, \infty)$ RLL arrays. Schematic for writing the $(3, \infty)$ sequences along the vacant spaces.

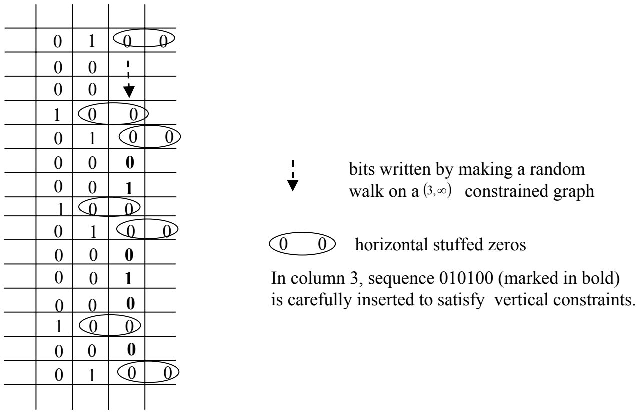

0 1 0 0
0 0
0 0
1 0 0
0 1 0 0
0 0 0
0 0 1
1 0 0
0 1 0 0
0 0 0
0 0 1
0 0 0
1 0 0
0 0 0
0 1 0 0
bits written by making a random walk on a (3,∞) constrained graph
0 0 horizontal stuffed zeros
In column 3, sequence 010100 (marked in bold) is carefully inserted to satisfy vertical constraints.

Figure 15: Illustration of the configuration for Case 2 for writing $(2, \infty, 3, \infty)$ RLL arrays. Non $(3, \infty)$ sequences are inserted carefully between the horizontal stuffed zeros such that the overall column constraints are satisfied in the second column. In the third column, the sequence 010100 is inserted between the stuffed zeros such that the overall $(3, \infty)$ column constraints are satisfied. The random walk on $\mathcal{G}_{(3,\infty)}$ is then resumed over the remaining vacant spaces in the column.

We note that the above two cases covers all the possibilities for generating valid arrays. Consider the conditional band entropy of the $d_1 + 1^{th}$ column conditioned on the previous d_1 columns. Let $W_1, W_2, ..., W_{d_1}$ denote the random variables representing the Hamming weights of a set of sequences placed along the columns from one to d_1, respectively. Assuming a uniform joint pmf over all the blocks of memory d_1, the information rate resulting from Case 1 can be obtained in the following combinatorial way.

Lemma 4.4. *The information rate obtained by enumerating all the configurations belonging to Case 1 is lower bounded as*

$$\lim_{m\to\infty} \frac{1}{m} \sum_{w_1=0}^{\lfloor \frac{m+d_2}{d_2+1} \rfloor} \sum_{w_2=0}^{\lfloor \frac{m-w_1+d_2}{d_2+1} \rfloor} \cdots \sum_{w_{d_1}=0}^{\lfloor \frac{m-\sum_{j=1}^{d_1-1} w_j+d_2}{d_2+1} \rfloor} P_{W_1 W_2 \ldots W_{d_1}} \log_2(\psi),$$

where,

- $P_{W_1 W_2 \ldots W_{d_1}}$ *is the joint probability mass function of the first* d_1 *column constrained sequences with the Hamming weights* w_1,w_2,...,w_{d_1} *respectively.*

- $\psi = \sum_{w_{d_1+1}=0}^{\lfloor \frac{m-\sum_{j=1}^{d_1} w_j + d_2}{d_2+1} \rfloor} T_{w_{d_1+1}} \left(m - \sum_{j=1}^{d_1+1} w_j - (w_{d_1+1} - 1)d_2 \right).$

- $T_w(m)$ *denotes the number of all valid* (d_2,∞) *1-D sequences of length* m *and Hamming weight* w.

Proof. Consider the placement of all valid column vectors of Hamming weight w_1 in the first column. When a '1' occurs in the first column, the tiling algorithm introduces stuffed zeros along the next d_1 columns of that row. The total free space in the second column is effectively $m - w_1$. Continuing this process iteratively for the next d_1 columns over the Hamming weights $w_2, w_3, ..., w_{d_1}$, the effective free space length in the $d_1 + 1^{th}$ column is $m - \sum_{j=1}^{d_1} w_j$.

Let $S^{(d_2,\infty)}_{(d_1+1)}$ denote the set of all (d_2,∞) column vectors that can be placed in the $d_1 + 1^{th}$. Denote $S^{(d_2,\infty)}_{w,i}$ to represent the set of all valid (d_2,∞) column vectors of Hamming weight w in the i^{th} column. The number of sequences that can be placed in the i^{th} column is given by

$$|S^{(d_2,\infty)}_{(i)}| = \left| \bigcup_{w=0}^{\lfloor \frac{m-\sum_{j=1}^{i-1} w_j + d_2}{d_2+1} \rfloor} S^{(d_2,\infty)}_{w,i} \right|. \tag{27}$$

Since the placement of a vector of Hamming weight w in the i^{th} column depends on the set of vectors with Hamming weights $w_1, w_2, ..., w_{i-1}$ from the previous columns, the joint probability mass function can be obtained as

$$P_{W_1 W_2,\ldots,W_{d_1}} = \prod_{j=1}^{d_1} P(W_j = w_j | W_1 = w_1, ..., W_{j-1} = w_{j-1}). \tag{28}$$

The conditional pmf in (28) can be obtained as

$$P(W_j = w_j | W_1 = w_1, W_2 = w_2, ..., W_{j-1} = w_{j-1}) = \frac{T_{w_j}(m - \sum_{k=1}^{j} w_k - (w_j - 1)d_2)}{|S^{(j)}|}. \quad (29)$$

Using (28) and (27), we can lower bound the entropy rate of the first d_1 columns as

$$\lim_{m\to\infty} \frac{1}{m} \sum_{w_1=0}^{\lfloor \frac{m+d_2}{d_2+1} \rfloor} \sum_{w_2=0}^{\lfloor \frac{m-w_1+d_2}{d_2+1} \rfloor} \cdots \sum_{w_{d_1}=0}^{\lfloor \frac{m-\sum_{j=1}^{d_1-1} w_j+d_2}{d_2+1} \rfloor} P(w_1, w_2, ..., w_{d_1}) \log_2(\psi). \quad (30)$$

□

Let $\alpha = \{0, 0^{d_2}1\}$ be an alphabet set. Let $P(0^{d_2}1) = p$ be the probability of emitting the sequence $0^{d_2}1$ from the set α. Lemma 4.4 is equivalent to writing independent and identically distributed (i.i.d) sequences chosen from the alphabet set α along the vacant spaces of the columns. Based on this fact, the following expression can be derived.

Lemma 4.5. *The information rate computed from Lemma 4.4 is equivalent to*

$$R^{(1)}_{(d_1,\infty,d_2,\infty)} \geq \sup_{p\in[0,1]} \tfrac{h(p)}{(1+d_1p)(1+d_2p)}.$$

Proof. Let $\mathcal{W}_1$ and $\mathcal{W}_2$ be the random variables corresponding to the emission of the alphabets $0^{d_2}1$ and 0 respectively. The entropy rate of this source is

$$H_n = \sup_{p\in[0,1]} \frac{h(p)}{E(\mathcal{W}_1 + \mathcal{W}_2)} = \sup_{p\in[0,1]} \frac{h(p)}{1 + d_2p} \quad (31)$$

Let f_s be the random variable representing the available free space length. By stuffing d_1 zeros horizontally, we can compute the available free space by the recursion

$$E(f_s) = m - d_1pE(f_s). \quad (32)$$

The rate corresponding to Case 1 is now lower bounded as

$$R_{(d_1,\infty,d_2,\infty)} \geq \frac{E(f_s)}{m} H_n. \quad (33)$$

Using (32) and (31) in (33), the lemma follows.

□

Though the Hamming weight structure is insightful, it is still difficult to enumerate all the cases pertaining to Case 2 especially for large values of d_1.

We will use the structure of tiling Algorithm-B to derive a constructive lower bound for the 2-D capacity. The basic idea is to compute the exact entropy rate for configurations belonging to Case 1 by assuming an 'identical composition' of valid column constrained sequences across all the columns. In other words, the entropy rate obtained by writing column constrained sequences in each column is the same and depends on the horizontal bit stuffing resulting from the previous d_1 columns.

Consider a random walk on the graph $\mathcal{G}_{(d,\infty)}$. Let x_t be a bit emitted at time t. Let s_t denote the state of the graph at time t. Let the probability of emitting a zero in the last state be $P(x_t = 0|s_t = d) = p$. The probability of emitting a zero from all the other states $i = 0, 1, ..., d-1$ is $P(x_t = 0|s_t = i) = 1$. The probability state transition matrix is given by $A = [a_{ij}] = P(s_{t+1} = j|s_t = i)$. Let $\pi = [\pi_0, \pi_1, ..., \pi_d]$ be the row vector containing steady state probabilities of all the states.

Theorem 4.3. *The capacity of* $(d_1, \infty, d_2, \infty)$ *constrained channels is lower bounded by*

$$C_{(d_1,\infty,d_2,\infty)} \geq \sup_{0\leq p\leq 1} \tfrac{h(p)}{1+(d_1+d_2)(1-p)},$$

where $h(p) = -p\log_2(p) - (1-p)\log_2(1-p)$.

Proof. We are working with a column constrained graph. Solving the eigenvalue relation $\pi\mathbf{A} = \pi$, the steady state probability π_i for states $i = 0, 1, .., d_2 - 1$ is the same and equals,

$$\pi_i = \frac{1-p}{1+d_2-d_2p} \tag{34}$$

For the last state, the steady state probability π_{d_2} is given by

$$\pi_{d_2} = \frac{1}{1+d_2-d_2p}. \tag{35}$$

The entropy rate for the 1-D Markov chain is obtained as

$$H_m = \pi_{d_2}h(p). \tag{36}$$

Horizontal bit-stuffing creates vacant spaces in the suceeding columns, restricting the amount of the available free space where the coded bits can be written. Let f_s be the length of the free space where bits can be written. The probability of emitting a one ($P(1)$) from the graph $\mathcal{G}_{(d_2,\infty)}$ is given by

$$P(1) = \pi_{d_2}(1-p). \tag{37}$$

The expected length of f_s can be computed as

$$E(f_s) = m - d_1 P(1) E(f_s). \tag{38}$$

The capacity of $(d_1, \infty, d_2, \infty)$ constrained channels can be lower bounded as

$$C_{(d_1,\infty,d_2,\infty)} \geq \lim_{m\to\infty} \sup_{0\leq p\leq 1} \frac{E(f_s)}{m} H_m. \tag{39}$$

Using (37) in (38) and (35) in (36), we can simplify (40) as

$$C_{(d_1,\infty,d_2,\infty)} \geq \sup_{0\leq p\leq 1} \frac{h(p)}{1+(d_1+d_2)(1-p)} \tag{40}$$

So far, we considered the case where the (d_2, ∞) constraints are along the columns. But, the capacity is invariant to swapping the row and column constraints. In other words,

$$C_{(d_1,\infty,d_2,\infty)} = C_{(d_2,\infty,d_1,\infty)} \tag{41}$$

Deriving the above analysis for the case when the column constraints are (d_1, ∞), we arrive at the same result as in equation (40). This proves the theorem.

□

4.3 Coding Schemes for Asymmetric (d, ∞) *Constraints*

In the previous sections, we discussed algorithms for constructing arrays satisfying the constraints and derived bounds for the maximum asymptotic information rate. In this section, we will develop an algorithm for mapping the information bits to coded arrays. We propose an algorithm based on the well-known state splitting technique [2] for constructing

2-D coded arrays. Our coding technique is sequential with a nested fixed rate column constrained code [46]. This is different from the variable rate bit-stuffing algorithm [41].

A: Constructing the Encoder

To obtain a mapping of raw bits to a 2-D coded array, we apply the ideas presented in tiling Algorithm-B along with the state splitting algorithm. The procedure is highlighted in the following steps.

1. Fix integers p and q that relatively are prime such that $\frac{p}{q} \leq C_{d_2,\infty}$, where $C_{d_2,\infty}$ is the 1-D capacity of the column constraints.
2. Obtain the q^{th} power of the (d_2,∞) constrained graph, i.e., $\mathcal{G}^{(q)}$.
3. Perform the basic v-consistent splitting of $\mathcal{G}^{(q)}$ and obtain the final encoder graph $\mathcal{G}'$ according to the state splitting algorithm [2].
4. Initialize: column $i = 1$.
5. Map the raw binary sequences to coded sequences in the i^{th} column by writing along the vacant spaces.
6. For every '1' occurring in i^{th} column of the coded array, stuff d_1 zeros to the right along the next columns.
7. Go to the next column and iterate over Step 5 over all the remaining columns.
8. Terminate the procedure after encoding all the columns.

B: Decoding Process

The decoding process follows the encoding principle. The following steps illustrate the procedure.

1. Initialize $i = n$, i.e., last column of the coded array.

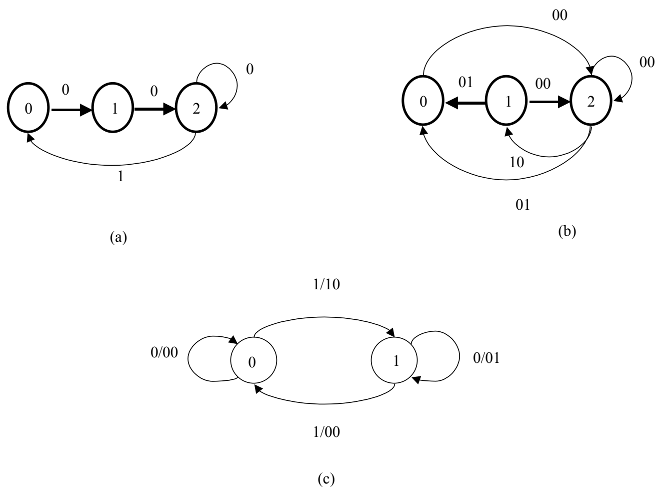


Figure 16: (a) $(2,\infty)$ column constrained graph $\mathcal{G}_{(2,\infty)}$. (b) second power Graph $\mathcal{G}^2$. (c) final encoder construction $\mathcal{G}'$ for $(2,\infty)$ constraints with rate 1/2.

2. Look back from columns $i - d_1$ to $i - 1$ and identify all the ones occurring in these columns.

3. Remove the stuffed zeros occurring in the next d_1 columns.

4. The resulting sequence is a code from the encoder graph $\mathcal{G}'$.

5. Decode the original bits p from the coded q binary blocks.

6. $i \leftarrow i - 1$. Iterate from Step 2 until all the bits are recovered.

C: Sequential Codes for $(1,\infty,2,\infty)$ *RLL Constraints*

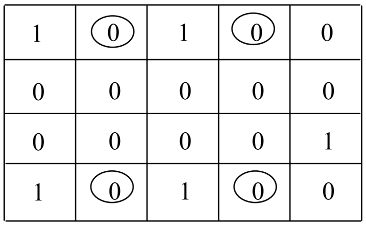
We will illustrate the encoding process with the design of a sequential code for the $(1,\infty,2,\infty)$ RLL constraint. The 1-D capacity of $(2,\infty)$ constraints is 0.5538. For the purpose of demonstrating an invertible mapping, we fix the rate of the column code as 0.5.

As a first step, consider the constrained graph $\mathcal{G}_{(2,\infty)}$, shown in Figure 16 (a). The graph has 3 states.

By considering all paths of length 2 originating from each state of $\mathcal{G}_{(2,\infty)}$ and terminating in all the possible states, we obtain the graph $\mathcal{G}^2$, as shown in Figure 16 (b).

After splitting and merging the states of $\mathcal{G}^2$ using the state splitting algorithm [30], we arrive at the final encoder state transition diagram, as shown in Figure 16 (c).

Consider the encoding of raw data on an $(m \times n)$ grid. For the sake of illustration, we will assume a 4×5 grid array. Let the input sequence be $\{10110101\}$. Let the starting state be '0' in $\mathcal{G}'$. The output sequence can be obtained as $\{10, 01, 00, 10, 01, 00, 00, 10\}$ by stepping through the graph $\mathcal{G}'$ starting from the state 0. We start by writing the coded sequence in the first column. Whenever a '1' appears in the coded sequence, a zero is written horizontally in the next column to the right. The coding pattern continues and we obtain a coded array, as shown in Figure 17. It is easy to note that the decoding procedure is tuned to the encoding principle. The coded array is scanned from the last column, the stuffed zeros are identified and deleted. This results in a 1-D coded sequence that was created from $\mathcal{G}'$. The raw bits can be easily extracted knowing the initial state.



1	⓪	1	⓪	0
0	0	0	0	0
0	0	0	0	1
1	⓪	1	⓪	0

Figure 17: Schematic of a $(1, \infty, 2, \infty)$ coded array on a 4×5 grid. The marked circles indicate stuffed bits.

At this point, we would like to comment on the choice of the decoder. We can use either an initial state dependent decoder or a sliding block decoder. The choice of a sliding block decoder circumvents the problems associated with the initial state dependencies and catastrophic error propagation. The construction of such decoders is well-known [30] and will not be addressed here. We considered a simple state dependent decoder for the purpose of illustrating a one-to-one mapping.

4.4 Numerical Results for the Capacity of Asymmetric (d, ∞) *Constraints*

In this section, we present the numerical results based on the capacity analysis presented in section 3. In the first example we will evaluate the capacity of 1-D (d, ∞) RLL constraints using Theorem 4.1 and compare it with the capacity computed using the adjacency approach[40].

We note from Table 3 that the combinatorial formula evaluated for finite 'm' is an upper bound. The numerical values computed using Theorem 4.1 for $m = 165$ agree well within less than 1% of the actual capacity. The numerical values approach the capacity of the constraint when $m \to \infty$.

Table 3: Capacity for 1-D (d, ∞) constraints

	Constraint	Combinatorial Formula-Thm 4.1	Adjacency Approach
1	$(1,\infty)$	0.69562	0.69424
2	$(2,\infty)$	0.55384	0.55146
3	$(3,\infty)$	0.46814	0.46495
4	$(4,\infty)$	0.40954	0.40568

In the second example, we consider a class of symmetric (d, ∞, d, ∞) constraints. In Table 4, we compare our upper bound and lower bounds computed from Theorems 4.2 and Theorem 4.3 with Theorem 8 [27] and the improved bit-stuffing lower bound [21] respectively. Since it is computationally intensive to obtain an exhaustive set of orthogonal column vectors for evaluating Theorem 4.2 for large values of m, we evaluate the bounds for $m = 12$.

From Table 4, we infer that our capacity upper bounds are better than the analytical results presented in [27]. Our lower bound agrees fairly well with the improved bit-stuffing lower bound [21]. The numerical results for our lower bounds and the bit-stuffing bound [41] are exactly the same, despite being derived from a different formula. This fact can be interpreted as follows. The entropy rate of the Markov chain for (d, ∞) constraints is mainly due to the uncertainty in the last state of $\mathcal{G}_{(d,\infty)}$ and is equivalent to writing biased bits from a distribution transformer[41].

Table 4: Capacity estimates for (d, ∞, d, ∞) constraints

d	Thm 4.2-Upper bound	Upper bound [27]	Thm 4.3-Lower Bound	Lower Bound [21]
1	0.5932	1.000	0.5515	0.5878
2	0.4294	0.7501	0.4056	0.4267
3	0.3530	0.6206	0.3281	0.3402
4	0.3078	0.5366	0.2787	0.2858

We now present a few capacity results for other asymmetric constraints for various values of d_1 and d_2. Tables 5 and 6 show the numerical computation of the bounds for a class of (1,∞,d,∞) and (2,∞,d,∞) constraints respectively. Since we are computing the conditional entropy upper bound for a finite value of m, i.e., $m = 12$, the upper bound computation will be an estimate for the capacity. However, the lower bound computation is a strict analytical bound.

Table 5: Capacity estimates for (1,∞,d,∞) constraints

d	$R_{(1,\infty,d,\infty)}$(Upper bound)	$R_{(1,\infty,d,\infty)}$(Lower Bound)
2	0.5167	0.4649
3	0.4613	0.4056
4	0.4149	0.3620

It is not surprising to extrapolate from Tables 5 and 6 that $R \to 0$ when $d_2 \to \infty$, keeping d_1 fixed.

Table 6: Capacity estimates for $(2, \infty, d, \infty)$ constraints

d	$R_{(2,\infty,d,\infty)}$(Upper bound)	$R_{(2,\infty,d,\infty)}$(Lower Bound)
3	0.3805	0.3620
4	0.3761	0.3281
5	0.3508	0.3011

4.5 Asymmetric $(0,k)$ *Constraints*

In this section, we outline a scheme for writing 2-D asymmetric k-constrained RLL arrays on an $m \times n$ rectangular grid based on the ideas presented in the previous sections. The encoding procedure is sequential and is done column by column. In the first step, $(0,k_2)$ column constrained sequences are written in the first k_1 columns. In the next step, the row locations where a string of successive zeros spanning the previous k_1 columns are identified. For each such row location, a '1' is placed in the k_1+1^{st} column adjacent to a string of successive k_1 zeros so that the row constraint is satisfied. The sequences satisfying the overall column constraints are then written in the vacant positions of the k_1+1^{st} column. This procedure is iteratively repeated for all the succeeding columns. The steady state memory depth of the column by column encoding process is k_1. In other words, while writing constrained sequences in the i^{th} column, we need to track a string of k_1 consecutive zeros that occurred in any row of the previous k_1 columns. These ideas are summarized in a simple algorithm below.

4.5.1 Tiling Algorithm

Consider the constrained graph $\mathcal{G}_{(0,k)}$, as shown in Figure 18.

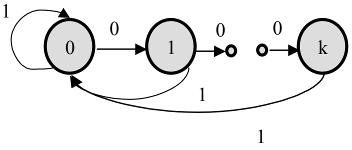


Figure 18: Constrained graph $\mathcal{G}_{(0,k)}$.

Outline of Algorithm

1. *Initialize:* $j = 1, k = k_2$.

2. Starting from the first column, do a random walk of length m on $\mathcal{G}_{(0,k_2)}$, shown in Figure 18.

3. Repeat step 2 for the first k_1 columns.

4. Initialize the column index $j = k_1 + 1$.

5. Locate the row indices $\{r_i\}_{i=1}^{p}$ where a string of k_1 consecutive zeros occurred in the columns from $j - k_1$ to $j - 1$.

6. For each row index r_i, stuff a '1' in the j^{th} column.

7. Write valid sequences along the vacant spaces of the j^{th} column by making a random walk on $\mathcal{G}_{(0,k_2)}$ such that the overall column constraints are satisfied.

8. $j \leftarrow j + 1$. Loop over Step 5 until all the n columns are filled.

Figure 19 shows how the constrained arrays can be written using the above tiling algorithm. We note that the random walk process is not a maxentropic random walk on $\mathcal{G}_{(0,k_2)}$ since a '1' is stuffed to satisfy the row constraints. The column by column encoding can be symmetrically extended to the row by row case. As mentioned in Step 7 of the algorithm, there are many locations where the vacant spaces between stuffed ones is less than k_2. After stuffing the ones to satisfy the row constraints, we should make a combination of the following moves:

- Do a random walk on the graph $\mathcal{G}_{(0,k_2)}$ along the vacant spaces between the stuffed ones.

- Identify the positions where the runlength of a vacant space between two successive ones is k_2. Insert a string of all zeros within this vacant space of length k_2. Over the remaining vacant space, do a random walk on $\mathcal{G}_{(0,k_2)}$.

An exhaustive and unique combination of the above moves will result in the enumeration of all 2-D valid arrays. Some of the configurations resulting from the move in the second step are depicted in Figure 20.

4.6 Capacity Bounds for Asymmetric $(0, k)$ *Constraints*

In this section, we analyze the tiling algorithm presented in the previous section for deriving a lower bound for the capacity of 2-D asymmetric $(0, k)$ constraints. Consider a random

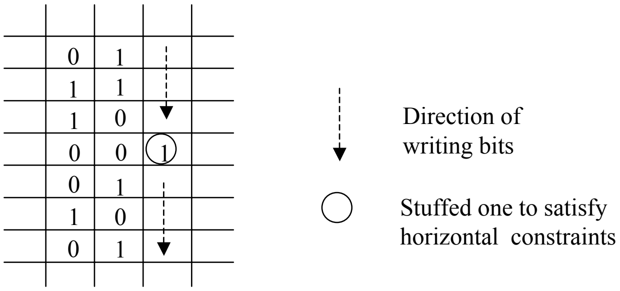

0
1
1
1
1
0
0
0
1
0
1
1
0
1
0
1
Direction of writing bits
Stuffed one to satisfy horizontal constraints

Figure 19: Schematic of writing $(0, k)$ arrays on a rectangle.

walk on the graph $\mathcal{G}_{(0,k_2)}$, shown in Figure 18. Let the transition probability of emitting a zero from any state $i = 0, 1, ..., k-1$ be $P(0|s = i) = p$. Thus, the probability of emitting a one is $P(1|s = i) = 1-p$. For the last state, i.e., state k, bit one is emitted with a probability one. This assumption on the structure of the state transition matrix will help us to derive analytical bounds as a function of the parameter p. We can relax this assumption and obtain slightly improved bounds by assuming different probability transitions. In that case, the analysis outlined below can be directly applied with an additional overhead of tracking k different transition probabilities for optimization. From the graph $\mathcal{G}_{(0,k_2)}$, it is clear that the probability transition matrix $\mathbf{A} = [a_{i,j}]$ is given by

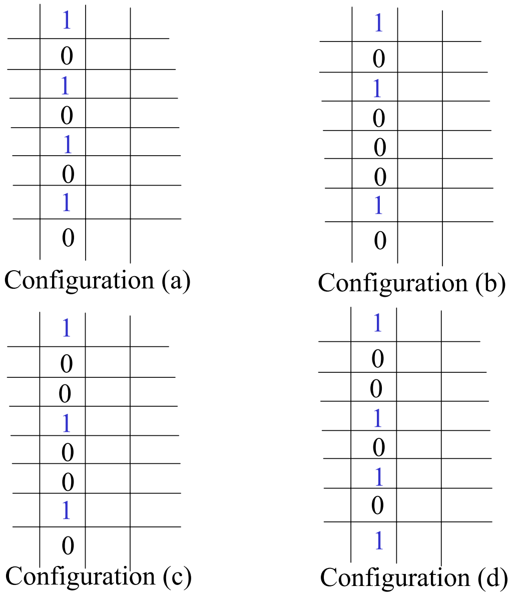
$$\mathbf{A} = \begin{bmatrix} 1-p & p & 0 & \cdots & 0 \\ 1-p & 0 & p & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & 0 & 0 & \cdots & 0 \end{bmatrix}. \tag{42}$$

Let $\pi = [\pi_0, \pi_1, ..., \pi_{k_2}]$ denote the steady state occupancy probabilities of the states $0, 1, ..., k_2$ respectively. From the eigenvalue relation, we have

$$\pi \mathbf{A} = \pi. \tag{43}$$

Using (42) in (43), the steady state occupancy probabilities π_i can be obtained as

$$\pi_i = \frac{p^i}{\sum\limits_{j=0}^{k_2} p^j}. \tag{44}$$



Configuration (a)

Configuration (b)

Configuration (c)

Configuration (d)

Figure 20: Example: $(0,k_1,0,3)$ Illustrating Step 7 while encoding 2-D $(0,k_1,0,3)$ RLL arrays - some configurations where a string of zeros are intentionally stuffed to generate additional 2-D sequences.

Using the ideas from the tiling algorithm, we derive a lower bound for the capacity of the 2-D constraints [47] as follows:

Theorem 4.4. *The capacity of $(0,k_1,0,k_2)$ constrained channels is lower bounded by*

$$C_{(0,k_1,0,k_2)} \geq \max\{\sup_{p\in[0,1]} \tfrac{(1-p^{k_2})(1-p^{k_2+1})^{k_1-1}h(p)}{(1-p^{k_2+1})^{k_1}+p^{k_1}(1-p^{k_2})^{k_1}}, \sup_{p\in[0,1]} \tfrac{(1-p^{k_1})(1-p^{k_1+1})^{k_2-1}h(p)}{(1-p^{k_1+1})^{k_2}+p^{k_2}(1-p^{k_1})^{k_2}}\}.$$

Proof. Consider the first case, i.e., encoding the data column by column. Recall from Step 7 of the tiling algorithm that a '1' is forcibly written when there is a string of k_1 consecutive zeros in the previous k_1 columns. Denoting the 2-D array by x, let E_1 be the event of forcing to write a '1' in the location $x_{i,j}$. Clearly, $E_1 : x_{i,j-1} = 0 \bigcap x_{i,j-2} = 0 \bigcap ... \bigcap x_{i,j-k_1} = 0$. Since each of $\{x_{i,k} = 0\}_{k=j-1}^{j-k_1}$ are independent events, the overall probability of stuffing a '1' is given by

$$P(E_1) = P_1(0)^{k_1}, \tag{45}$$

where $P_1(0)$ is the probability that a '0' was written by a random walk on the column constrained graph. But we have

$$P_1(0) \quad = \quad p\sum_{k=0}^{k_2-1}\pi_k$$

$$= \quad (1-\pi_{k_2})p. \tag{46}$$

After stuffing a 1 in the row locations of the j^{th} column corresponding to the event E_1, vacant spaces created in this column. Let f_s be the random variable denoting the available free space where bits can be written. The expected length of this free space is given by the recursion

$$E(f_s) = m - P(E_1)E(f_s). \tag{47}$$

The capacity of the column constrained scheme can be lower bounded as

$$C^{(1)}_{0,k_1,0,k_2} \geq \lim_{m\to\infty} \frac{E(f_s)}{m} R_1, \tag{48}$$

where R_1 is the entropy rate of column constrained Markov process. But we can compute R_1 as

$$\begin{aligned} R_1 &= \sum_{k=0}^{k_2-1} \pi_k h(p) \\ &= (1-\pi_{k_2})h(p). \end{aligned} \tag{49}$$

Using (47) and (49) in (48) and simplifying, we get

$$C^{(1)}_{0,k_1,0,k_2} \geq \frac{(1-\pi_{k_2})h(p)}{1+[p(1-\pi_{k_2})]^{k_1}}. \tag{50}$$

Using (44) in (50), and maximizing (50) over all the choices of p,

$$C^{(1)}_{0,k_1,0,k_2} \geq \sup_{p\in[0,1]} \frac{(1-p^{k_2})(1-p^{k_2+1})^{k_1-1}h(p)}{(1-p^{k_2+1})^{k_1}+p^{k_1}(1-p^{k_2})^{k_1}}. \tag{51}$$

In many cases when $k_2 > k_1$, it is possible to improve the bounds by doing a row by row encoding of data. This is because the assumed structure of the transition probabilities in the constrained graph benefits the likelihood of stuffing lesser ones to satisfy the row constraints. Repeating the above analysis for the row by row encoding of data through a row constrained graph $\mathcal{G}_{(0,k_1)}$, we get

$$C^{(2)}_{0,k_1,0,k_2} \geq \sup_{p\in[0,1]} \frac{(1-p^{k_1})(1-p^{k_1+1})^{k_2-1}h(p)}{(1-p^{k_1+1})^{k_2}+p^{k_2}(1-p^{k_1})^{k_2}}. \tag{52}$$

It is trivial to note that the 2-D capacity of the constraints is the same when the row and the column constraints are swapped. Picking a better of the two lower bounds from (51) and (52), the theorem follows.

□

We will now establish an upper bound for these constraints by making the following observations.

Lemma 4.6. *The number of sequences satisfying the joint constraints is lesser than or equal to the number of sequences satisfying either the row or column constraints separately.*

$$
\begin{aligned}
N_{m,n}^{(0,k_1,0,k_2)} &\leq N_{m,n}^{(0,\infty,0,k_2)}. \\
N_{m,n}^{(0,k_1,0,k_2)} &\leq N_{m,n}^{(0,k_1,0,\infty)}.
\end{aligned}
$$

Proof. Consider the process of writing the sequences column by column according to the tiling algorithm. When the k_2 constraint does not exist, column constrained sequences can be written independently without violating the constraints. Since there is no need for stuffing a '1' to satisfy the row constraints, the maxentropic rate is bounded by the 1-D capacity of the column constraints. Similarly, the result holds when row by row encoding is considered. □

Theorem 4.5. *The capacity of $(0, k_1, 0, k_2)$ constrained channels is upper bounded by*

$$C_{(0,k_1,0,k_2)} \leq C_{0,\min(k_1,k_2)},$$

where $C_{0,\min(k_1,k_2)}$ is the minimum of the 1-D capacities for the row and column constraints.

Proof. From Lemma 4.6 we have,

$$N_{m,n}^{(0,k_1,0,k_2)} \leq N_{m,n}^{(0,\infty,0,k_2)}. \tag{53}$$

$$N_{m,n}^{(0,k_1,0,k_2)} \leq N_{m,n}^{(0,k_1,0,\infty)}. \tag{54}$$

Taking logarithms on both sides of (53) and (54) in the limiting case,

$$\lim_{m,n\to\infty} \frac{\log_2(N_{m,n}^{(0,k_1,0,k_2)})}{mn} \leq \lim_{m,n\to\infty} \frac{\log_2(N_{m,n}^{(0,\infty,0,k_2)})}{mn} \tag{55}$$

$$\lim_{m,n\to\infty} \frac{\log_2(N_{m,n}^{(0,k_1,0,k_2)})}{mn} \leq \lim_{m,n\to\infty} \frac{\log_2(N_{m,n}^{(0,k_1,0,\infty)})}{mn}. \tag{56}$$

Equations (55) and (56) actually denote the combinatorial entropy of the 2-D constraints. The existence of these limits can be proved by sub-additivity arguments [27]. When either one of the joint constraints do not exist, the resulting combinatorial entropy is just the entropy of the other 1-D constraint.

$$\begin{aligned}
\lim_{m,n\to\infty} \frac{\log_2(N_{m,n}^{(0,\infty,0,k_2)})}{mn} &= C_{(0,k_2)} \qquad (57)\\
\lim_{m,n\to\infty} \frac{\log_2(N_{m,n}^{(0,k_1,0,\infty)})}{mn} &= C_{(0,k_1)}. \qquad (58)
\end{aligned}$$

Using (57) and (58) in (55) and (56) respectively, we get

$$\lim_{m,n\to\infty} \frac{\log_2(N_{m,n}^{(0,k_1,0,k_2)})}{mn} = C_{(0,k_1,0,k_2)} \leq \min\{C_{(0,k_1)}, C_{(0,k_2)}\}. \qquad (59)$$

Since the 1-D capacity of a $(0, k)$ constraint is a non-decreasing function of k, we conclude that $\min\{C_{(0,k_1)}, C_{(0,k_2)}\} = C_{0,\min(k_1,k_2)}$. This completes the proof.

□

4.7 Coding Schemes for Asymmetric $(0, k)$ Constraints

We will sketch the encoding and decoding process for constructing sequential codes with rates close to the derived 2-D lower bounds.

A: Constructing the Encoder

To obtain an invertible mapping of raw bits to a 2-D coded array, we apply the same coding ideas that we presented in section 4.3 for $(d_1, \infty, d_2, \infty)$ constraints with some minor modifications. We will outline the steps for clarity.

1. Fix integers p and q that are relatively prime such that $\frac{p}{q} \leq C_{0,k_2}$, where C_{0,k_2} is the capacity of the 1-D column constraint.

2. Obtain the q^{th} power of $\mathcal{G}_{(0,k_2)}$, i.e., $\mathcal{G}^{(q)}$.

3. Perform the basic v-consistent splitting of $\mathcal{G}^{(q)}$ and obtain the final encoder graph $\mathcal{G}'$ according to the state splitting algorithm [30].

4. Initialize: column $j = k_1 + 1$.

5. Map the raw binary sequences to coded sequences along the j^{th} column by writing along the vacant spaces of that column.

6. Locate the row indices $\{r_i\}_{i=1}^{p}$ where a string of k_1 consecutive zeros occurred in the columns from $j - k_1$ to $j - 1$.

7. For each row index r_j, stuff a '1' in the j^{th} column.

8. Go to Step 5 or identify locations where the length of the void space between successive stuffed ones is less than k_2. Randomly stuff a string of all zeros in the length of the void spaces and iterate Step 5 over all the columns.

9. Terminate the procedure after encoding all the columns.

We note that the bit-stuffing due to horizontal constraints will result in an overall rate approaching the derived lower bounds for the 2-D constraints.

B: Decoding Process

The decoding process follows the encoding principle. The following steps illustrate the procedure.

1. Initialize: $j = n$, i.e., last column of the coded array.

2. Look back from the columns $j - k_1$ to $j - 1$ and identify the locations where an all zero pattern occurred.

3. Remove the stuffed ones and the string of all zeros of length $\leq k_2$ (if any) along the consecutive void spaces in i^{th} column.

4. The resulting sequence is a code from the encoder graph $\mathcal{G}'$.

5. Decode the original p bits from the coded q binary blocks.

6. $j \leftarrow j - 1$. Iterate from Step 2 until all the bits are recovered.

4.8 Numerical Results for the Capacity of Asymmetric $(0, k)$ Constraints

In Tables 7 and 8, we show the numerical computation of the bounds for some choices of the 2-D constrained parameters. We compare our lower and upper bounds with Theorems 3 and 7 in [27] respectively and the bit stuffing lower bounds [41]. From the tables, we note that our bounds are much better than the bounds in [27], marginally better than bit stuffing bounds [41], and extend to the general class of asymmetric constraints.

Table 7: Capacity bounds for symmetric (0,k) constraints

k	C_{LB}-Thm 4.4	C_{LB}-Thm 3 [27]	C_{LB} [41]	C_{UB}-Thm 4.5	C_{UB}-Thm.7[27]
1	0.5515	0.4122	0.5515	0.6942	0.7925
2	0.7768	0.4122	0.7769	0.8791	0.9358
3	0.8826	0.7061	0.8788	0.9468	0.9767
4	0.9368	0.7061	0.9320	0.9752	0.9908

Table 8: Capacity bounds for (0,k_1,0,k_2) constraints

(k_1, k_2)	C_{LB}-Thm 4.4	C_{UB}-Thm 4.5
(1,2)	0.6484	0.6942
(1,8)	0.6942	0.6942
(2,3)	0.8334	0.8791
(2,4)	0.8571	0.8791
(3,5)	0.9288	0.9468

Before we conclude this section, we would like to comment on the bounds and the code construction. Recall from the tiling algorithms that vacant spaces are created after stuffing a '1' to satisfy the row(column) constraints. When the length of the vacant space between any two successive stuffed ones is less than or equal to $k_2(k_1)$, depending on the column by column(row by row) encoding scheme, we could randomly choose a set of void locations and stuff them with a string of zeroes. We could then do a random walk on the graph $\mathcal{G}_{(0,k_2)}(\mathcal{G}_{(0,k_1)})$ as depicted in Figure 20. By enumerating all such configurations and computing the additional combinatorial entropy, we can improve the lower bounds to approach the capacity of the 2-D constraints. A thorough analysis of this case is a daunting task. We present an improved enumeration bound for the capacity of the $(0, 1, 0, 1)$ RLL constraint in Appendix A.

4.9 Summary

In this chapter, we presented tiling algorithms for constructing asymmetric (d, ∞) and $(0, k)$ constrained arrays. We derived bounds for the capacity of these constraints and highlighted the Hamming weight structure of the arrays. Using the tiling algorithms, we presented code constructions with rates close to the derived lower bounds. Some of the bounds on the symmetric case compare well with the existing bounds. In other cases, our bounds are the first reported bounds for asymmetric constraints. An open problem at this point in time is to generalize the bounds presented in this chapter to an arbitrary 2-D RLL constraint with finite d and k constraints.

CHAPTER V

CAPACITY BOUNDS FOR MULTI-LEVEL RUNLENGTH-LIMITED CONSTRAINED ARRAYS

Localized holographic recording [34] is characterized by high signal-to-noise ratios sufficient for supporting multi-level/M-ary channel codes. In an M-ary coding scheme, data is encoded in M levels, i.e., $\{0, 1, 2, ..., M-1\}$. We can get a raw coding gain of $\log_2(M)$ using multi-level codes. Motivated by the application of such codes for ultra-high density localized holography, we examine the 2-D capacity of M-ary RLL constrained channels. A 2-D M-ary constrained modulation scheme can be imagined as a combination of pulse amplitude, pulse width, and pulse position modulation [5] in two dimensions.

In this chapter, we derive the lower bounds for the capacity of asymmetric $(M, 0, k)$ and (M, d, ∞) RLL arrays and deduce coding algorithms by extending the constructions in chapter 4. This chapter is organized as follows. In section 5.1, we derive bounds for the capacity of asymmetric $(M, 0, k)$ constraints and present some numerical results in section 5.2. We derive the capacity bounds for asymmetric (M, d, ∞) constraints in section 5.3 and discuss the numerical results in section 5.4. We summarize our results in section 5.5.

First, we begin with a few definitions related to 2-D M-ary RLL arrays.

Definition 5.1. *A 2-D array satisfies (M, d_1, k_1, d_2, k_2) RLL constraints on a rectangular lattice if there are at least d_1 zeros and at most k_1 zeros between any two non-zero symbols horizontally; and at least d_2 zeros and at most k_2 zeros between any two non-zero symbols vertically.*

Definition 5.2. *The capacity of a noiseless 2-D M-ary RLL constrained channel is defined as* $C_{(M,d_1,k_1,d_2,k_2)} = \lim_{m,n\to\infty} \frac{\log_2(N_{m,n}^{(M,d_1,k_1,d_2,k_2)})}{mn}$.

5.1 Capacity Bounds for Asymmetric $(M, 0, k)$ *Constraints*

In this section, we derive a lower bound for the capacity of asymmetric 2-D M-ary k-constrained arrays [43] by extending our ideas on the binary constraints. Figure 21 shows the schematic of a 2-D $(4, 0, 2, 0, 1)$ RLL constraint on a 4×5 grid.

1	0	3	1	0
2	1	0	0	2
0	0	3	2	1
1	2	0	0	2

Figure 21: Schematic of a $(4, 0, 2, 0, 1)$ RLL array on a 4×5 grid. Each symbol is from a 4-ary alphabet.

The bounds for the capacity of the constrained channels can be derived using the basic tiling algorithm presented in [49] and can be summarized as follows.

1. Column constrained sequences are sequentially written along the first k columns by doing a random walk on an M-ary k-constrained graph, as shown in Figure 22.

2. For every string of k adjacent zeros that are juxtaposed along the k consecutive columns of a particular row, a non-zero symbol is placed in the $k + 1^{st}$ column in that row. This ensures that the row constraint is satisfied.

3. Along the vacant spaces of the $k + 1^{st}$ column, constrained sequences are placed such that the overall column constraints are satisfied.

The above three steps are sequentially repeated for all the columns.

Consider an M-ary 1-D k-constrained graph as shown in Figure 22. Let s_t denote the state of the graph at time t. The source emits M-ary alphabets $x_t \in \{0, 1, ..., M - 1\}$, such that, between any two non-zero symbols there are at most k zeros. The probability state transition matrix for the graph in Figure 22 is given by, $A = [a_{ij}] = P(s_{t+1} = j | s_t = i)$. Let the probability of emitting the symbol 0 from each state be p. In other words, $P(x_t = 0 | s_t = i) = p$ for all the states $s = 0, 1, ..., k - 1$. Forcing a uniform distribution over all the other

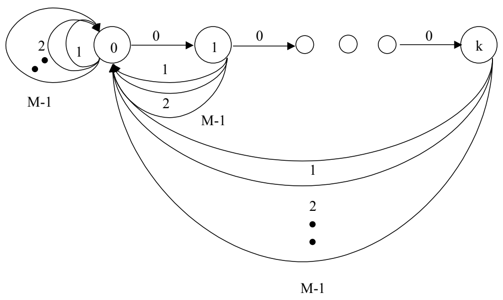


Figure 22: Schematic of a $(M, 0, k)$ 1-D constrained graph.

alphabets, let the probability of emitting a non-zero symbol b be $P(x_t = b | s_t = i) = \frac{1-p}{M-1}$ for all the states $s = 0, 1, ..., k-1$ and uniformly $\frac{1}{M-1}$ for the last state, i.e., $s = k$. We restrict the transition probabilities in this way so that a computable closed-form expression can be tractably derived as a function of p and M. In general, non-uniform transition symbol probabilities can be assumed for obtaining slightly improved bounds. Let $\pi = [\pi_0 \pi_1 ... \pi_k]$ denote the row vector representing the steady state occupancy probabilities.

The following theorem provides a lower bound for the capacity of asymmetric $(M, 0, k)$ 2-D RLL constraints.

Theorem 5.1. *The capacity of $(M, 0, k_1, 0, k_2)$ constrained channels is lower bounded by*

$$C_{(M,0,k_1,0,k_2)} \geq \max\{ \sup_{p \in [0,1]} f_1(p) R_1(p), \sup_{p \in [0,1]} f_2(p) R_2(p) \},$$

where $f_1(p) = \frac{(p^{k_2+1}-1)^{k_1}}{(p^{k_2+1}-1)^{k_1} + (p^{k_2+1}-p)^{k_1}}$,
$R_1(p) = [-p \log_2(p) - (1-p) \log_2(\frac{1-p}{M-1})](\frac{p^{k_2}-1}{p^{k_2+1}-1}) + (\frac{p^{k_2+1}-p^{k_2}}{p^{k_2+1}-1}) \log_2(M-1)$,
$f_2(p) = \frac{(p^{k_1+1}-1)^{k_2}}{(p^{k_1+1}-1)^{k_2} + (p^{k_1+1}-p)^{k_2}}$,
$R_2(p) = [-p \log_2(p) - (1-p) \log_2(\frac{1-p}{M-1})](\frac{p^{k_1}-1}{p^{k_1+1}-1}) + (\frac{p^{k_1+1}-p^{k_1}}{p^{k_1+1}-1}) \log_2(M-1)$.

Proof. We will prove this result by using the structure of the probability state transition matrix described in the previous paragraph. The proof is similar to the analysis in Theorem 4.4.

From the structure of $\mathbf{A}$, the steady state occupancy probabilities π_i for each state i

can be obtained as

$$\pi_i = \frac{p^i}{\sum\limits_{j=0}^{k_2} p^j}. \tag{60}$$

Let us consider the case when the encoding is done using a column constrained graph $\mathcal{G}_{(M,0,k_2)}$. A non-zero symbol is forcibly written whenever a string of k_1 consecutive zeros occurs in the previous k_1 columns. Denoting the 2-D array by x, let E_1 be the event of forcing a non-zero symbol in the location $x_{i,j}$. Clearly, $E_1 : x_{i,j-1} = 0 \bigcap x_{i,j-2} = 0 \bigcap ... \bigcap x_{i,j-k_1} = 0$.

Since each of $\{x_{i,k} = 0\}_{k=j-1}^{j-k_1}$ are independent events, the overall event probability of stuffing a non-zero symbol is given by

$$P(E_1) = P_1(0)^{k_1}, \tag{61}$$

where $P_1(0)$ is the probability that a '0' was written by a random walk on the column constrained graph. But we have

$$\begin{aligned} P_1(0) &= p\sum_{k=0}^{k_2-1} \pi_k \\ &= (1-\pi_{k_2})p. \end{aligned} \tag{62}$$

After stuffing a 1 in the row locations of the j^{th} column corresponding to the event E_1, vacant spaces are created along the column. Let f_s be the random variable denoting the available free space where bits can be written. The expected length of this free space is given by the recursion

$$E(f_s) = m - P(E_1)E(f_s). \tag{63}$$

The capacity of the column constrained scheme can be bounded as

$$C^{(1)}_{M,0,k_1,0,k_2} \geq \lim_{m\to\infty} \frac{E(f_s)}{m} R_1, \tag{64}$$

where R_1 is the entropy rate of column constrained Markov process. We can compute R_1 as

$$R_1 = \sum_{k=0}^{k_2-1} \pi_k H(p, \frac{1-p}{M-1}, ..., \frac{1-p}{M-1}) + \pi_{k_2} H(\frac{1-p}{M-1}, \frac{1-p}{M-1}, ..., \frac{1-p}{M-1}), \tag{65}$$

where $H(.)$ is the usual entropy function.

Substituting (65) in (64) and simplifying using (60), (63), and (62), the overall 2-D rate can be maximized over all the possible choices of p as

$$C^{(1)}_{M,0,k_1,0,k_2} \geq \sup_{p\in[0,1]} f_1(p)R_1(p). \tag{66}$$

Capacity is invariant to a change in the order of the row and column constraints. Repeating the analysis when the column constraint is $(M,0,k_1)$ and the row constraint is $(M,0,k_2)$, we get

$$C^{(2)}_{M,0,k_1,0,k_2} \geq \sup_{p\in[0,1]} f_2(p)R_2(p). \tag{67}$$

Choosing the maximum of the two bounds in (66) and (67), the theorem follows.

□

By letting $k_1 = k_2 = k$ in Theorem 5.1, we can obtain a lower bound for the capacity of symmetric $(M,0,k)$ constrained channels using the following corollary.

Corollary 5.1. *The capacity of $(M,0,k)$ constrained channels is lower bounded by*

$$C_{(M,0,k)} \geq \sup_{p\in[0,1]} \tfrac{(p^{k+1}-1)^k}{(p^{k+1}-1)^k+(p^{k+1}-p)^k} R(p),$$

where $R(p) = [-p\log_2(p) - (1-p)\log_2(\frac{1-p}{M-1})](\frac{p^k-1}{p^{k+1}-1}) + (\frac{p^{k+1}-p^k}{p^{k+1}-1})\log_2(M-1)$.

The following theorem provides an upper bound for the 2-D capacity of M-ary asymmetric $(0,k)$ RLL constraints. The derivation is straightforward and we avoid a repetitive proof here.

Theorem 5.2. *The capacity of $(M,0,k_1,0,k_2)$ constrained channels is upper bounded by*

$$C_{(M,0,k_1,0,k_2)} \leq C_{(M,0,\min(k_1,k_2))},$$

where $C_{(M,0,\min(k_1,k_2))}$ is the minimum of the 1-D capacities for M-ary $(0,k)$ row and column constraints.

We note that the coding algorithms presented in the previous section for the binary can be straightforwardly extended to the M-ary case and we avoid repetitive discussions.

5.2 Numerical Results for Asymmetric $(M, 0, k)$ Constraints

In Tables 9 and 10, we show the numerical computation of the bounds for some choices of the 2-D constrained parameters.

Table 9 shows the computation of the bounds for the symmetric case. The bounds are tight for increasing values of k. We believe that the lower bound is closer to capacity than the upper bound.

Table 9: Capacity bounds for $(M, 0, k)$ constraints

(M, k)	C_{LB}-Thm 5.1	C_{UB}-Thm 5.2
(2,2)	0.7768	0.8791
(4,2)	1.9113	1.9824
(2,3)	0.8826	0.9468
(4,3)	1.9727	1.9957

Table 10: Capacity bounds for $(M, 0, k_1, 0, k_2)$ constraints

(M, k_1, k_2)	C_{LB}-Thm 5.1	C_{UB}-Thm 5.2
(2,1,3)	0.6805	0.6942
(3,2,4)	1.5349	1.5458
(4,3,5)	1.9937	1.9957
(5,4,7)	2.3215	2.3216

We note that the upper and lower bounds are tight for increasing values of k_1 and k_2 and approach $\log_2(M)$ for large values of k_1 and k_2.

5.3 Capacity Bounds for Asymmetric (M, d, ∞) Constraints

In this section, we derive bounds for the capacity of (M, d, ∞) constraints. By definition, a $(M, d_1, \infty, d_2, \infty)$ RLL constraint has at least d_1 zeros and at least d_2 zeros between any two non-zero symbols along rows and columns respectively. Figure 23 shows the schematic of a 2-D $(M, 1, \infty, 2, \infty)$ array on a 4×5 grid. We can construct M-ary coded sequences by

2	0	0	0	2
0	2	0	3	0
0	0	3	0	0
1	0	0	0	2

Figure 23: Schematic of a $(4, 1, \infty, 2, \infty)$ RLL array on a 4×5 grid.

a minor modification of the tiling Algorithm-B described in section 4.1. Instead of stuffing d_1 zeros for the occurrence of a one along any column, we stuff d_1 zeros across the columns for every non-zero symbol that occurs in any particular column.

The random walk mechanism on a M-ary constrained graph is described in Figure 24. We assume that the probability of emitting a non-zero symbol from the last state is uniformly $\frac{1-p}{M-1}$. The probability of emitting a zero is p for state d, and one for all the other states. By proceeding with the analysis as outlined in Theorem 5.1, we can prove the following result.

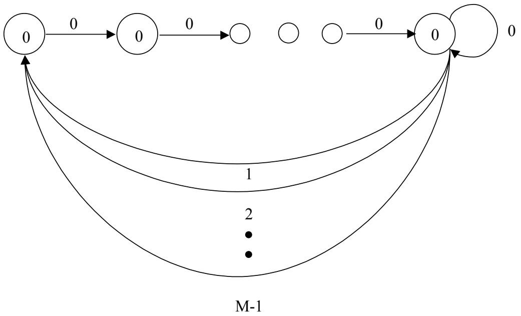


Figure 24: Schematic of the (M, d, ∞) constrained graph.

Theorem 5.3. *The capacity of* $(M, d_1, \infty, d_2, \infty)$ *constrained channels is lower bounded by*

$$C_{(M,d_1,\infty,d_2,\infty)} \geq \sup_{p\in[0,1]} \frac{-p\log_2(p)-(1-p)\log_2(\frac{1-p}{M-1})}{1+(d_1+d_2)(1-p)}.$$

By substituting $d_1 = d_2 = d$ in Theorem 5.3, we have the following corollary.

Corollary 5.2. *The capacity of* (M, d, ∞) *constrained channels is lower bounded by*

$$C_{(M,d,\infty)} \geq \sup_{p\in[0,1]} \frac{-p\log_2(p)-(1-p)\log_2(\frac{1-p}{M-1})}{1+2d(1-p)}.$$

5.4 Numerical Results for Asymmetric (M, d, ∞) Constraints

In Tables 11 and 12, we show the numerical computation of the bounds for some choices of the 2-D constrained parameters.

Table 11: Capacity bounds for (M, d, ∞) constraints

(M, d)	C_{LB}- Thm 5.3
(4,2)	0.6203
(8,2)	0.8062
(4,3)	0.4848
(8,3)	0.6183

Table 12: Capacity lower bounds for $(M, d_1, \infty, d_2, \infty)$ constraints

(M, d_1, d_2)	C_{LB}-Thm 5.3
(2,1,3)	0.4056
(3,2,4)	0.3281
(4,3,5)	0.4025
(5,4,7)	0.3509

For large values of d_1 and d_2, the 2-D capacity shrinks to zero.

5.5 Summary

In this chapter, we examined the capacity of 2-D M-ary RLL constraints motivated by applications in localized holography. By extending our tiling algorithms and bounding techniques for the binary case, we generalized the bounding techniques for computing the capacity of $(M, 0, k_1, 0, k_2)$ and $(M, d_1, \infty, d_2, \infty)$ constraints. The coding algorithms for the 2-D M-ary RLL case can be derived by a straightforward extension of our algorithms for the binary case.

CHAPTER VI

M-ARY, BINARY, AND SPACE-VOLUME MULTIPLEXING TRADE-OFFS FOR HOLOGRAPHIC CHANNELS

We are interested in the theoretical limits for the amount of error-free information that can be physically stored and retrieved from a holographic disk drive. The achievable storage density is a function of the number of recorded pages per unit volume, the number of pixels per page, and the capacity of the holographic channel. We pointed out in chapter 2 that the channel capacity ultimately determines the amount of information that can be stored within the medium. If the channel capacity is zero, irrespective of system enhancements, zero storage density is realized. Thus, the capacity of holographic channels is an important result to understand the physical limits for data storage. We need to compute information-theoretic limits for predicting the amount of data storage and for designing multi-level codes that can achieve these limits. Recently, with the introduction of advanced coding techniques such as low density parity check codes applied to multi-level coding [20], [42], [54], [55], it is possible to come very close to the theoretical limits given in this chapter using practical algorithms.

In this chapter, we derive a lower bound for the channel capacity using the transmission model developed by Heanue et al [22]. Using this bound, we examine the trade-off between the storage density and the number of recorded pages for angle multiplexing and localized holographic recording [51], [50]. We optimize the number of recorded pages and the desired level of a modulation code for maximizing the storage density. This chapter is organized as follows. In section 6.1, we present the transmission model for holographic channels. In section 6.2, we highlight the need for M-ary modulation codes for holography and determine the i.i.d capacity of holographic channels. In section 6.3, we present an analysis for maximizing the volumetric storage density by examining the density versus multiplexing trade-off. In section 6.4, we review the bit-error rate versus signal-to-noise ratio performance of M-ary codes for holographic channels and summarize the results in section 6.5.

6.1 Transmission Model for Holographic Channels

In this section, we review the transmission model [22] developed by Heanue, Bashaw, and Hesselink. This model will be used in our subsequent analysis. The information storage in holographic channels can be modeled as data transmission over a noisy communications channel. The magnitude of the received signal r can be represented by the vector addition of the signal amplitude A and a random noise phasor. The noise is predominantly due to optical scattering and is characterized by a circularly symmetric Gaussian probability distribution. The detector is a square law device that records the intensity $y = |r|^2$ of the received signal. Assuming that the magnitude and phase of the received signal are independent, the probability density function (pdf) of the magnitude of the received vector r is given by [22]

$$p_R(r) = \frac{r}{\sigma^2} \exp\left(-\frac{r^2 + A^2}{2\sigma^2}\right) I_o\left(\frac{rA}{\sigma^2}\right), \tag{68}$$

where I_0 is the zero-order modified Bessel function of the first kind, and σ^2 is the noise variance. We note that (68) is a Rician pdf commonly seen in wireless communication systems for describing the output statistics of an incoherent receiver.

At the CCD, the detected signal is intensity whose pdf is given by

$$p_Y(y) = \frac{1}{2\sigma^2} \exp\left(-\frac{y + A^2}{2\sigma^2}\right) I_o\left(\frac{\sqrt{y}A^2}{\sigma^2}\right). \tag{69}$$

Let x_H and x_L denote the transmitted intensities for the 'On' and 'Off' pixels respectively. Define the signal-to-noise ratio as $S = \frac{x_H}{2\sigma^2}$ and the contrast factor as $c = \frac{x_H}{x_L}$. Let $\tilde{y} = \frac{y}{x_H}$ be the normalized detected intensity. Using these definitions in (69), the pdf of the

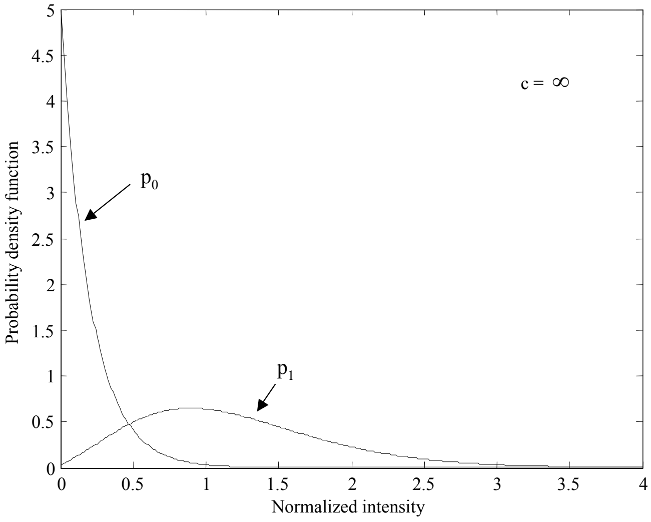
[2] We note that this definition of SNR is actually the peak SNR, and is a commonly used definition in the optics community. However, in communication theory the standard definition for the SNR is average SNR and given by $S = \frac{E(y)}{\sigma^2}$. For consistency, we adhere to the peak SNR definition [22] in our analysis.

normalized detected intensities for the ‘On’ and the ‘Off’ pixels are respectively given by [22]

$$p_1(\tilde{y}) = S \exp\left[-S(\tilde{y}+1)\right] I_0\left(2S\sqrt{\tilde{y}}\right), \tag{70}$$

$$p_0(\tilde{y}) = S \exp\left[-S\left(\tilde{y}+\frac{1}{c}\right)\right] I_0\left(2S\sqrt{\frac{\tilde{y}}{c}}\right). \tag{71}$$

Figure 25 shows the pdf of the ‘On’ and ‘Off’ pixels evaluated for different contrast ratios at 5dB SNR. As we can observe from the plots in Figure 25, an infinite contrast ratio is preferred since the probability of error is minimized for detection. However, practical systems always have a finite contrast ratio, which makes the detector design complicated. The transmission model that we described in this section is for the binary case. In the next section, we address the M-ary encoding of pixels and analyze the channel capacity.



(a)

6.2 M-ary Encoding of Pixels

In the M-ary encoding scheme, each SLM pixel is encoded as a gray-level with a certain intensity level chosen from a set of M intensity levels. Typically, M is chosen as a power of two so that an integer number of bits can be assigned to each level. At the detector, the

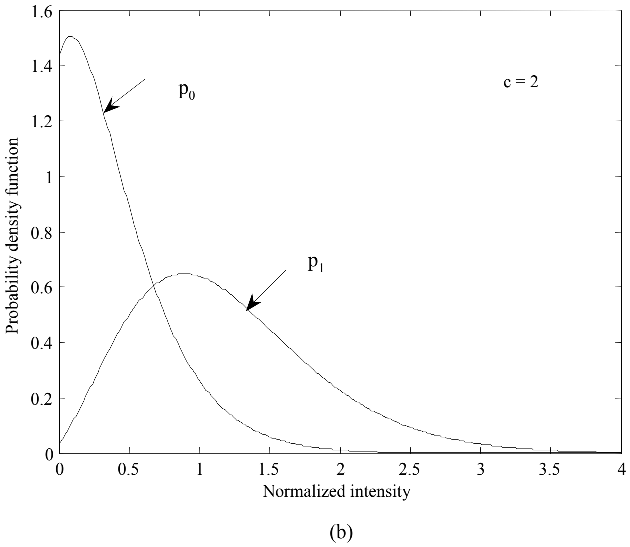


Figure 25: Probability density function of the received intensity (a) SNR = 5dB, c = ∞ (b) SNR = 5dB, c = 2

received intensity should be properly distinguished from neighboring pixel intensities. The detection process can be done using a simple thresholding scheme.

One of the main points of interest is the spacing of the intensity levels to minimize the bit-error rate. There are two simple possibilities:

- Equal spacing in amplitude.
- Equal spacing in intensity.

When the SLM amplitude levels are equally spaced, the intensity levels are quadratically spaced. The transmitted intensity level x_m for the m^{th} level is given by

$$x_m = \left(\sqrt{x_L} + m\frac{\sqrt{x_H} - \sqrt{x_L}}{M-1}\right)^2. \tag{72}$$

When the intensity levels are equally spaced, the intensity level x_m for the m^{th} level is given by

$$x_m = x_L + m\frac{x_H - x_L}{M-1}. \tag{73}$$

The variance in the detected intensity depends on the transmitted signal. From (69), the standard deviation of the detected intensity can be computed as [22]

$$\sigma_y^2 = 4\sigma^2(\sigma^2 + A^2). \tag{74}$$

From equation (74), we infer that the variance in the intensity of the detected signal depends on the intensity of the transmitted signal. This suggests that a non-uniform spacing of the intensities can reduce bit-error rates since the overlapping probability regions of the detected signals are minimal. Simulation results in [22] suggest a uniform spacing in the amplitude levels so that the bit-error rates are minimized. For our subsequent analysis, we assume a uniform spacing of amplitudes. However, an optimal strategy for placing the intensities needs to be worked out analytically.

6.2.1 Lower Bound for the Capacity of the Holographic Channel

With a uniform spacing of SLM amplitudes, the pdf of the normalized intensity for the m^{th} level is given by

$$p_m(\tilde{y}) = S \exp\left[-S\left(\tilde{y} + \frac{x_m}{x_H}\right)\right] I_0\left(2S\sqrt{\frac{\tilde{y}x_m}{x_H}}\right). \tag{75}$$

Using (75), we can compute a lower bound for the holographic channel capacity C. The following theorem provides a lower bound for the holographic channel capacity as a function of the signal-to-noise ratio.

Theorem 6.1. *The capacity C of a holographic channel is lower bounded by*

$$C \geq \sup_M \tfrac{1}{M}\left[\int_0^\infty \sum_{m=0}^{M-1} p_m(\tilde{y}) \log_2(p_m(\tilde{y}))d\tilde{y} - \sum_{m=0}^{M-1}\int_0^\infty p_m(\tilde{y}) \log_2\left(\frac{\sum_{m=0}^{M-1} p_m(\tilde{y})}{M}\right) d\tilde{y}\right].$$

Proof. To determine the holographic channel capacity, we need to determine the a priori probability distribution of the input x that maximizes the mutual information $I(x, y)$ between the input and output y [15]. In other words,

$$C = \sup_{p(x)} I(x; y). \tag{76}$$

To get a computable lower bound, we pick a particular family of probability distributions and compute the mutual information. By choosing a uniform probability distribution at the input, the a priori probability of each SLM intensity level is assumed to be $\frac{1}{M}$ for all the M levels. The mutual information computed for a uniform distribution is called the i.i.d capacity of the channel ($C_{i.i.d}$) and will always be a lower bound for the true capacity. For the transmission model in (75), the i.i.d capacity can be computed as

$$C_{i.i.d}(M) = I(x, \tilde{y}) = h(\tilde{y}) - h(\tilde{y}|x), \tag{77}$$

where $h(\tilde{y})$ and $h(\tilde{y}|x)$ are the differential and conditional differential entropies respectively.

Equation (75) is the conditional pdf for a certain intensity level. The overall pdf of the detected intensity is given by

$$p(\tilde{y}) = \sum_{m=0}^{M-1} \frac{1}{M} p_m(\tilde{y}). \tag{78}$$

Using (78) and (75), we compute the differential entropy terms in (77) as

$$h(\tilde{y}) = -\int_0^\infty p(\tilde{y}) \log_2(p(\tilde{y}))d\tilde{y} \tag{79}$$

$$h(\tilde{y}|x) = -\frac{1}{M} \sum_{m=0}^{M-1} \int_0^\infty p_m(\tilde{y}) \log_2(p_m(\tilde{y}))d\tilde{y}. \tag{80}$$

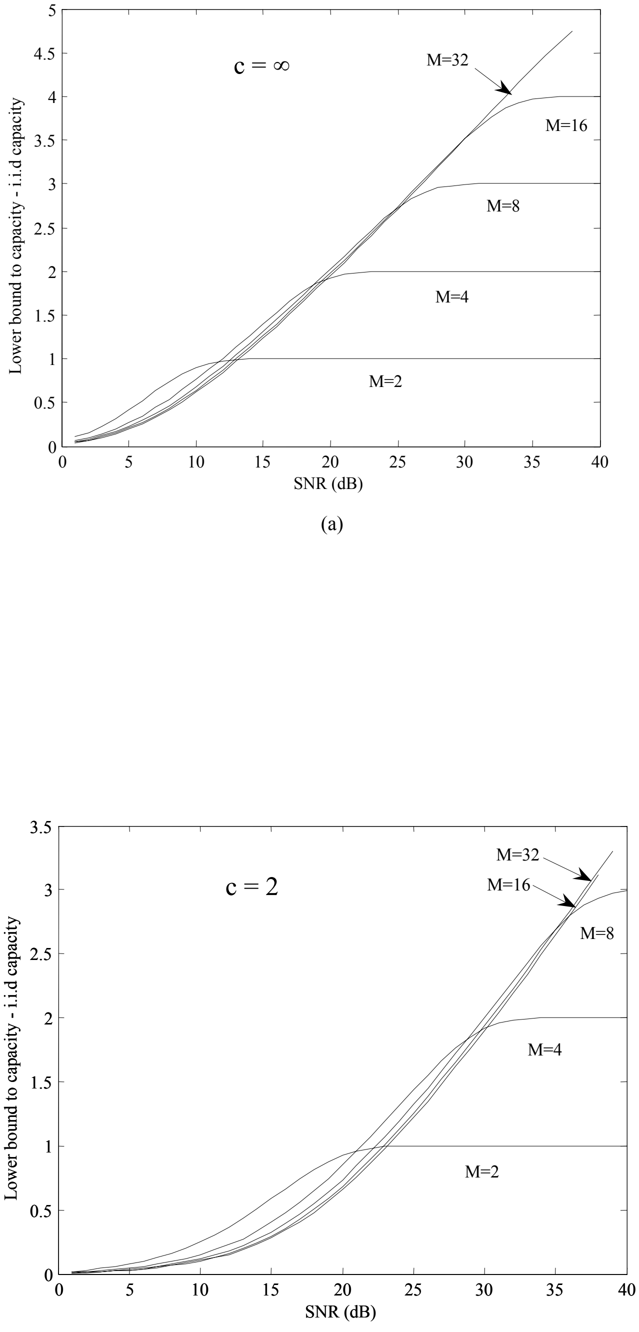
Substituting (79) and (80) in (77), the theorem follows.

□

We note that Theorem 6.1 can be computed using numerical integration.

In Figure 26 (a), we plot the i.i.d capacity versus SNR for different values of M at an infinite contrast ratio. As we can observe from Figure 26 (a), for a given number of levels M, the i.i.d capacity converges to $\log_2(M)$ at high SNRs. This observation can be explained as follows. When the noise floor approaches zero, the output is a close replica of the input. The mutual information reduces to computing the self-entropy of the source. Assuming a uniform prior distribution, the self-entropy is $\log_2(M)$ bits.

Figure 26 (b) shows the computation of i.i.d capacity for a small contrast ratio $c = 2$. For a given SNR and an M-ary level, a low contrast ratio decreases the achievable information rate.



(a)

(b)

Figure 26: Plot of the i.i.d capacity curves for several M-ary levels as a function of SNR (a) $c = \infty$ (b) $c = 2$

We note that $C_{i.i.d}$ is not a monotonically increasing function of the modulation level M. This fact can be observed in Figures 26 (a) and (b). In other words, for a given SNR, it may be such that a lower value of M can give a higher i.i.d capacity. This is because we are fixing the input distribution to be uniform. Only when the channel description is exactly known for real continuous inputs, the prior distribution of the input can be chosen to maximize the mutual information according to equation (76).

Theorem 6.1 is useful for two reasons. First, we can guarantee a certain achievable information rate using an M-ary modulation code. Second, we can use a combination of 2-D modulation and error correcting codes to achieve $C_{i.i.d}$ for any given SNR. Of course, designing rate-efficient 2-D modulation/error correcting codes can be challenging in practice.

6.2.2 Overall Storage Density

Computing the overall storage density D_s is the main focus of practical interest. The achievable rate $R(S)$ (bits/channel use) is a function of the available SNR (S) that the material can provide. The number of recorded pages P per unit volume is a function of the diffraction efficiency which is related to the SNR of the system.

Suppose each data page has B pixels per page coded at an average rate R, the overall storage density D_s in bits per unit volume is given by

$$D_s = P(S)BR(S). \tag{81}$$

The number of SLM pixels B per page is fixed. Using Theorem 6.1, we can compute $R(S)$. The next step is to compute $P(S)$ and optimize D_s for a given SNR. In the next section, we study the trade-off between the storage density and multiplexing and suggest the optimal number of pages that should be recorded.

6.3 Density versus Multiplexing Trade-off

In this section, we examine angle multiplexing [3] and localized recording [34] from a signal-to-noise ratio point of view. We pointed out in section 2.2.1 that the diffraction efficiency is related to the recording mechanism.

- In angle multiplexed holography, several holograms share the entire volume of the holographic medium, as illustrated in Figure 27. The diffraction efficiency of each hologram is inversely proportional to the square of the number of recorded holograms [14].

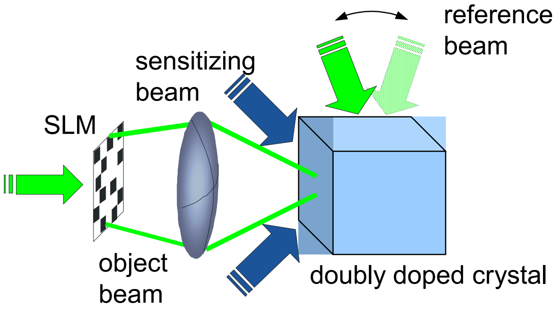


Figure 27: Angular multiplexing holography: holograms overlap over the entire volume of the doubly doped crystal.

- In localized holography, each hologram is recorded within a thin slice of the medium, as shown in Figure 28. The diffraction efficiency of a hologram in localized recording is inversely proportional to the number of recorded holograms.

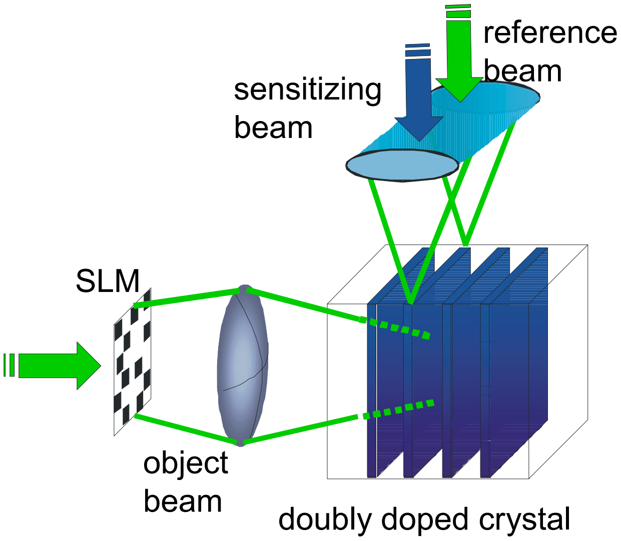


Figure 28: Localized holography: holograms are recorded within the slices along the doubly doped crystal.

Localized holography offers the unique advantage of selective recording and erasure of holograms which is not present in angle multiplexed volume holography. From a storage standpoint, by using localized holography, we can record a few hundred holograms compared to a thousand holograms in the angle multiplexed case. However, localized holography provides improved SNR than angle multiplexing. By designing multi-level codes for localized holography, we can achieve higher coding gains, thereby, maximizing the overall storage density.

Thus, given the SNR budget for a material/medium, we are interested in maximizing the density D_s by optimally choosing the number of recorded pages and the number of levels of a multi-level modulation code. To make meaningful comparisons, we fix $M/\#$ to be the same for both recording mechanisms. Let this constant be κ.

Let us first consider localized holography. Fixing the number of slices/holograms, let this number be M_l. The resulting SNR for localized holography S_{lo} is given by

$$S_{lo} = \frac{\kappa}{2P_l\sigma^2}. \tag{82}$$

Assuming that the channel statistics do not change with the recording mechanism, the information rate can be computed by reading the maximum value of i.i.d capacity for S_{lo} from Figure 26. The overall density for localized holography $D_s^{(l)}$ is computed as

$$D_s^{(l)} = \frac{\kappa}{2\sigma^2 S_{lo}} BR(S_{lo}). \tag{83}$$

Since P_l is fixed, $D_s^{(l)}$ can be exactly computed.

We now look into the angle multiplexing case. Let P_a be the number of pages that can be multiplexed within the volume of the medium. The storage density $D_s^{(a)}$ is given by

$$D_s^{(a)} = P_a BR\left(\frac{\kappa^2}{2P_a^2\sigma^2}\right). \tag{84}$$

To achieve the best storage density, we need to optimize (84) with respect to the number of pages and the number of levels. The optimum density (D^*) is given by

$$D^* = \max_{P_a, M} D_s^{(a)}. \tag{85}$$

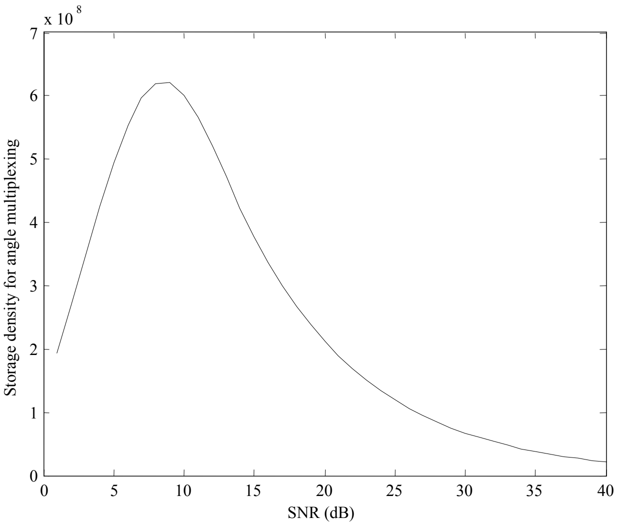


Figure 29: Achievable storage density as a function of the SNR for M=2 levels.

Example: We will explain this trade-off with an example. Let $\kappa = 3$ and $\sigma^2 = 10^{-6}$. Choosing $M_l = 400$ with $B = 10^6$ pixels/page, the SNR can be computed as $S_{lo} = 38.75$dB. The best rate for 38.75dB can be read from Figure 26 (a) as 4.5 bits/channel use. Thus, $D_s^{(l)} = 1.8$Gb per unit volume for localized holography.

Following the optimization in (85), for angle multiplexing, a storage density of at least 0.62Gb per unit volume can be achieved using binary recording over 753 pages. Figure 29 shows the optimization results for the binary case.

Thus, given two different recording schemes with the same material and constraints, M-ary localized recording seemed better in this example. The theoretical analysis can be worked out for different practical choices of the system parameters.

6.4 Probability of Error for Threshold Detection

In the previous sections, we developed a framework for analyzing the capacity and the overall storage density of holographic channels. These results suggest using a joint modulation/error correcting code, for reliable storage and retrieval of digital data. To recover the information bits, we need efficient detection algorithms. The construction of practical maximum likelihood (ML) detectors is an open problem in 2-D signal processing. However, a simple algorithm can be realized using threshold detection. In threshold detection, an

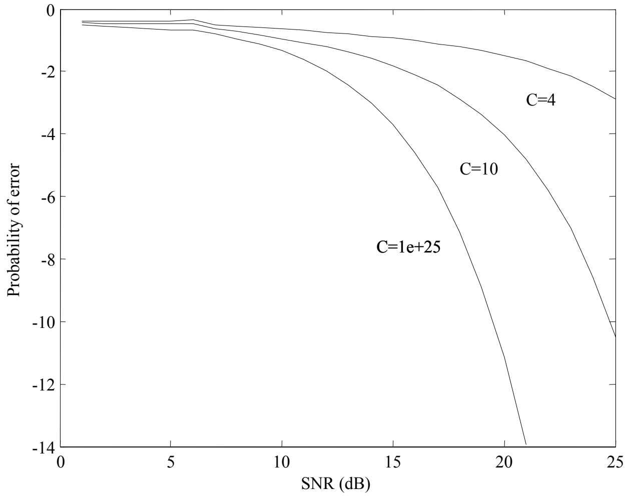


Figure 30: Probability of error for threshold detection for M=2 levels and different constrast ratios.

optimum threshold is chosen for comparing the detected signal statistics for decoding the symbols. Heanue, Bashaw, and Hesselink [22] analyzed the probability of error for M-ary threshold detection. The probability of error as a function of the SNR can be computed as

$$P_{error} = \sum_{m=0}^{M-1} \frac{1}{M} \left[\int_0^{\tilde{y}_{T,m}} p_m(\tilde{y}) d\tilde{y} + \int_{\tilde{y}_{T,m+1}}^{\infty} p_m(\tilde{y}) d\tilde{y} \right], \tag{86}$$

where $y_{T,m}$ is the threshold between the levels $m-1$ and m.

Figure 30 shows the probability of error as a function of the SNR for different contrast ratios. The optimal thresholds were chosen according to equations (9) and (10) in [22]. We infer that maintaining a practically large contrast ratio (infinite contrast is hard to realize) using better optical systems can provide improved bit error rates.

6.5 Summary

We presented an analysis for the storage density versus multiplexing trade-off of holographic channels. This result proves that recording a lot of holograms does not necessarily maximize the storage density. There is an optimal choice for the number of recorded pages and the number of levels of a multi-level code that maximizes the volumetric density. We analyzed our results for localized holography and angle multiplexed holography. These results are analytical and useful for understanding the benefits and limitations of various recording schemes. We also reviewed results for the probability of error of a threshold detector. We conclude that maintaining a high contrast ratio is beneficial for reducing the bit error rate and increasing the capacity. An exact description of the channel in terms of the conditional probability distribution for real continuous inputs is needed. Such a model will be helpful for precisely characterizing the holographic channel capacity.

CHAPTER VII

TWO-DIMENSIONAL TRANSLATION AND ROTATIONAL PIXEL MISREGISTRATION: SIGNAL RECOVERY AND PERFORMANCE LIMITS

Signal processing is an integral part of any data storage system. A practical holographic system has several optical components with inherent limitations in the fabrication and design. We need signal processing algorithms to model the data, to compensate interpixel interference, and to recover the data from noisy detected samples.

The detection and imaging process in most systems is not perfect. The inherent effects of band-limiting aperture, diffraction, misfocus, magnification, optical aberrations, and material shrinkage [14] lead to interpixel interference. We need signal processing algorithms to remove the residual energy from unintended pixels and recover the transmitted data.

The detection process in a holographic system is non-linear. The signal intensity integrated over the effective spatial aperture of the detector pixel is received at the CCD. The integrating effect coupled with interpixel cross talk and random noise makes the signal recovery problem rather difficult. Several authors have considered 2-D linear models for signal recovery based on different optimality criteria. There are signal processing algorithms [13], [23], [28], [53], for reducing interpixel interference, for correcting pixel blurs, and for recovering the signal from a linear combination of known pixel patterns. However, there are relatively very few algorithms [11], [23], [32], for correcting pixel shifts. Burr developed signal reconstruction algorithms [11] for compensating fractional lateral shifts in two dimensions. Rotational misalignments lead to non-uniform fractional shifts that are not constant over pixels. In other words, pixels that are farther away from the center suffer more severe distortion than those at the center.

In this chapter, we extend the idea of handling fractional shifts for combined translation and rotational distortion [48], [44]. Our algorithm is applicable for optically misaligned systems with square apertures and for holographic systems with low fill factors in the transmitter and detector arrays.

The chapter is organized as follows. In section 7.1, we formulate a channel model for translational and rotational misalignments and derive a bound for the detector efficiency. In section 7.2, we derive maximum likelihood estimators for determining the unknown misalignment parameters and analyze the asymptotic efficiency of these estimators through Cramer-Rao bounds. In section 7.3, we develop a compensation algorithm for recovering information bits from the detector output in the presence of rotation and translation misalignments and additive white Gaussian noise. We also present an analysis for the error performance of the algorithm. In section 7.4, we present simulation results relating to the decoding performance. Discussions are summarized in section 7.5.

7.1 Channel Model for Translation and Rotational Misalignment

A coherent holographic setup forms the basis of our channel model. Figure 3 shows the basic holographic setup. The system comprises of two identical lenses separated by the sum of their focal lengths. A square aperture of dimensions D is placed in the common focal plane of the two lenses. This square aperture is a crystalline material where holograms are recorded. The transmitter is an equispaced pixelated SLM array. The detector array is typically a charge-coupled device (CCD) and is assumed to be identical to the SLM. The spatial sampling rate is determined by the spacing of the pixels in the SLM. We assume that the SLM spacing is identical to the aperture width. The pixel-spread function is the convolution of the space-invariant impulse response (due to the aperture) with the pixel shape. The space-invariant impulse response is determined by the continuous space Fourier transform of the aperture shape. With a square aperture, the impulse response is an integrated 2-D separable sinc function given by

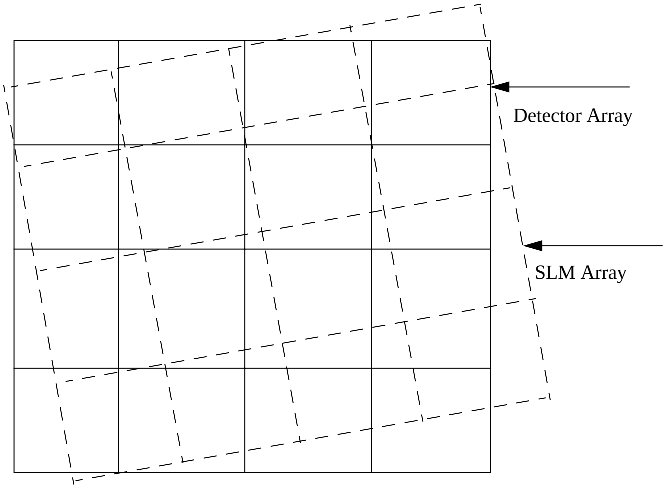


Figure 31: Rotation and translational misalignment of the SLM array with respect to the detector ($\sigma_x = 0.5, \sigma_y = 0.5, \alpha = 3^o$)

$$h(x,y) = c \int_{-\frac{1}{2}g_{SLM}}^{\frac{1}{2}g_{SLM}} \frac{\sin(\pi(x-x'))}{\pi(x-x')} dx' \int_{-\frac{1}{2}g_{SLM}}^{\frac{1}{2}g_{SLM}} \frac{\sin(\pi(y-y'))}{\pi(y-y')} dy', \tag{87}$$

where g_{SLM} is the SLM fill factor where the transmitted intensity is modulated. The variables x, x', y, and y' are in the units of the pixel dimensions and the normalizing constant c is chosen so that $\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} h^2(x,y)dxdy = 1$. The lower and upper limits in the integral of (87) are due to the planar field intensity from the SLM in the region where the CCD is placed. We note that $h(x,y) = 1$ when evaluated at the center of the CCD pixel and is oscillatory decaying along both the axes. We assume that the pixel pitch dimensions are normalized to unity.

Figure 31 shows the schematic of a misaligned SLM array with respect to the detector about the optical axis. It must be noted that the effects of translation and rotation do not commute with each other. A 2-D translation followed by a rotation must be viewed as the transformation of translation via the rotational transform. In this chapter, we assume that there is a rotation first and then a translation. This assumption makes sense since rotation is an inherent misalignment and cannot be perfectly corrected even though the optical centers of the two arrays are exactly aligned. The translation could be due to mechanical misadjustments. The order of translation and rotation do not in any way affect the method of analysis but the details of the channel model for the detected signal will be different. The analysis for translation first followed by rotation can be trivially worked out, like the rotation first followed by the translation case.

7.1.1 Detected signal and Coordinate Transformations

The angle of rotation α is assumed to be positive in the anti-clockwise direction and the 2-D lateral translations (σ_x, σ_y) are such that $|\sigma_x|, |\sigma_x| \leq \frac{1}{2}$. The coordinates of any point on the SLM with respect to the detector can be obtained as $R(x,y)^T + (\sigma_x, \sigma_y)^T$, where R is the rotational transform given by

$$R = \begin{pmatrix} \cos(\alpha) & -\sin(\alpha) \\ \sin(\alpha) & \cos(\alpha) \end{pmatrix}. \tag{88}$$

The received signal at the detector pixel $d(m,n)$ is given by

$$d(m,n) = \int_{-\frac{1}{2}g_{CCD}}^{\frac{1}{2}g_{CCD}} \int_{-\frac{1}{2}g_{CCD}}^{\frac{1}{2}g_{CCD}} \left(\sum_{m_i,n_i} g_{m_i,n_i}(x,y) \sqrt{s(m_i,n_i)} \right)^2 dxdy + w(m,n), \tag{89}$$

where g_{CCD} is the CCD fill factor where most of the intensity at the detector pixel is received. The term $s(m_i, n_i)$ denotes the binary signal from the SLM pixel with discrete index (m_i, n_i) that overlaps with the detector pixel with a 2-D discrete index (m,n). The term $w(m,n)$ denotes the noise at the output of the detector. For small angles α, the indices (m_i, n_i) of the SLM pixels contributing to the cross talk terms in the detected signal $d(m,n)$ are due to $(\lceil (m-\sigma_x)\cos(\alpha) + (n-\sigma_y)\sin(\alpha) \rceil, \lceil -(m-\sigma_x)\sin(\alpha) + (n-\sigma_y)\cos(\alpha) \rceil)$, and its three neighbors on the left, the bottom, and the left-diagonal corners. Referring to Figure 31, let us fix the origin as the center of the detector grid array. The signal at the detector pixel with top right corner coordinates $(1,2)$ is indexed as $d(1,2)$ and has energy mainly contributed by the SLM pixels $s(2,2), s(2,1), s(1,2)$, and $s(1,1)$. The kernel $g_{m_i,n_i}(x,y)$ is a rotated version of the function $h(x,y)$ and is given by

$$g_{m_i,n_i}(x,y) = \frac{1}{|R|} h(x-a) h(y-b), \tag{90}$$

where $(a,b) = (m_i - \frac{1}{2}, n_i - \frac{1}{2})R^T - (\sigma_x, \sigma_y)$. Also, $|R| = 1$ since R is orthonormal.

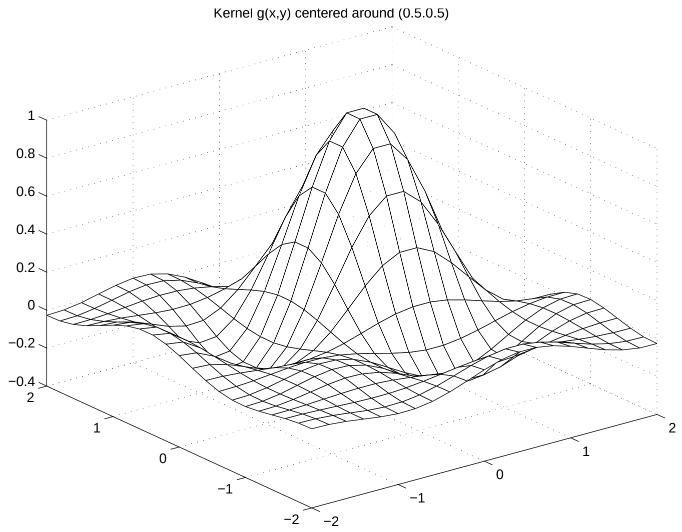


Figure 32: Schematic of the rotated and translated kernel $g(x, y)$.

Figure 32 shows the sketch of the kernel $g(x, y)$ centered at the point $(\frac{1}{2}, \frac{1}{2})^T$. We can imagine a tiling of such kernels at the center of each SLM pixel. These kernels low pass filter the transmitted signal causing a pixel blur. The effect of rotation results in a non-uniform interpixel interference at the intended detector pixel. The goal of the problem is to recover the transmitted bits from the detected samples $\{d(m, n)\}$.

7.1.2 Detector Efficiency

Without loss of generality, we assume that the detector grid array and the SLM array are of size $2m \times 2m$ and each square is of unit area. Without any coding, every element of this uniformly spaced grid array is an equally likely binary symbol. Without any noise and misalignment, $4m^2$ information bits can be stored and retrieved. Let A_s and A_d denote the areas of the transmitted and detected arrays respectively. As a result of translational and rotational misalignments, not all the SLM pixels land exactly aligned to the CCD array. We are interested in calculating the resulting inefficiency because of the misalignments. The number of SLM bits rendered useless is given by the fraction of the area that does not overlap between the transmitted and detected arrays. Thus, the portion of the channel not containing $A_s \bigcap A_d$ is lost due to the misalignments. Hence, the effective area is the 2-D overlapping region between the transmitter and the detector arrays. Ideally, when the detector has infinite region of support, all the information bits can be recovered. However, when the grid arrays are finite, the number of transmitted bits lost (due to effective channel seen by the detector) is given by

$$T_{bits\_lost} = 4m^2 \left(\frac{A_s - A_s \bigcap A_d}{A_s} \right). \tag{91}$$

From simple coordinate geometry, we can compute the overlapping areas of the CCD and SLM arrays and obtain an upper bound on the number of transmitted bits that can be recovered losslessly. The result is stated in Fact 7.1, and the derivation is relegated to Appendix B for the sake of continuity.

Fact 7.1. *For $2m \times 2m$ equally likely binary symbols of the transmitted signal, at most $4m^2(1 - T_{loss})$ bits can be recovered from the rotational and translational misalignments. The parameter T_{loss} is given by*
$T_{loss} = \frac{1}{2} f(\alpha) g(\alpha) + \frac{1}{4}[\epsilon_1\epsilon_2 + \epsilon_3\epsilon_4]$,

where

$$\begin{aligned} f(\alpha) &= 1 + \cot(\alpha) - \sin(\alpha) - \cot(\alpha)\cos(\alpha), \\ g(\alpha) &= 1 + \tan(\alpha) - \cos(\alpha) - \tan(\alpha)\sin(\alpha), \\ \epsilon_1 &= \frac{\pm\sigma_x \mp \sigma_y \cot(\alpha)}{m}, \\ \epsilon_2 &= \frac{\pm\sigma_x \tan(\alpha) \mp \sigma_y}{m}, \\ \epsilon_3 &= \frac{\pm\sigma_x \cot(\alpha) \pm \sigma_y}{m}, \\ \epsilon_4 &= \frac{\pm\sigma_x \pm \sigma_y \tan(\alpha)}{m}. \end{aligned} \tag{92}$$

By choosing a large grid array, i.e., as $m \to \infty$, T_{loss} is a function of α. Asymptotically, rotational misalignment affects the system performance. We observe that the loss function T_{loss} is periodic with $\frac{\pi}{2}$ and the maximum loss is around 0.18 bits per pixel occurring at an angle $\pm\frac{n\pi}{4}$. This absolute loss is unavoidable. When the angle of rotation is 45^o, many detector pixels are rendered useless, implying inefficient use of expensive optical components.

Figure 33 shows a plot of the loss function sketched for $0 \leq \alpha \leq \frac{\pi}{4}$. We note that this function has a maxima at $\frac{\pi}{4}$. This result is rather intuitive to guess. The number of bits lost due to the misalignments is rather insignificant for small angles.

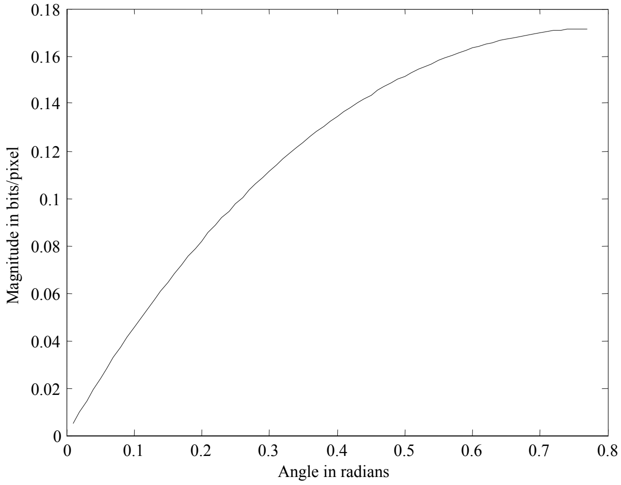


Figure 33: Loss function in bits/pixel as a function of the angle of rotation.

In the presence of additive noise, the transmitted pixels can suffer errors. In such cases, the fundamental information rate of the 2-D channel needs to be determined. Let C_{2D} be the capacity of the 2-D ISI channel. The number of bits that can be recovered losslessly in the presence of misalignments and noise can be upper bounded as $4m^2(1 - T_{Loss})C_{2D}$. Computing C_{2D} is an open problem. In this chapter, we focus on the signal recovery aspects for compensating 2-D misalignment.

7.2 Signal Recovery: Parameter Estimation

We now consider the problem of recovering the input pixels from the detected signals. We assume small angles of rotational misalignment [14], i.e., less than or equal to 3 degrees. This is a pragmatic assumption. The translation misalignments are a fraction of the pixel. Typically, $|\sigma_x|, |\sigma_y| \leq \frac{1}{2}$. Large angles of rotation can always be compensated for by carefully aligning the parts until a point where it is difficult to fine tune the angle alignment. Integer translations can always be easily compensated by coarse alignments and re-indexing the detected signal coordinates. Without any loss of generality, we assume that the detector array is a square grid of size $2m \times 2m$. We define a coordinate system of the detector array as follows. The coordinate of the center of the detector grid array is designated as the origin $(0, 0)$. The index for a detector pixel is identified by the coordinate of its right top corner. The coordinates of the SLM array are referenced with respect to the detector's coordinate system. This convention will be followed throughout the rest of this chapter. There are two issues that need to be addressed here. The first step is to get an estimate of the unknown misalignment parameters buried in noise. The next step is to detect and recover the bits.

Parameter estimation is an important precursor step before applying any compensation technique. Robinson and Milanfar [38] have analyzed the effect of parameter estimation on the performance of gradient-based image registration techniques. Their analysis is motivated by the motion estimation problem commonly encountered in video processing applications. In the present work, we are interested in the bit error rate after misalignment compensation. Hence, the error measure is formulated in a 2-D communications framework.

We need to estimate the misalignment parameters before feeding this information into the decoding engine. We assume that the unknown parameters are non-random. By sending a known preamble of a pixel pattern and measuring the detected signals at a pre-designated location, we can statistically estimate the parameters. The measured parameters can be used subsequently in the reconstruction algorithm. Without any loss of generality, we assume that σ_x and σ_y are positive, as shown in Figure 31. By presetting the transmitted SLM array to an all zero pattern except at the locations $s(0,0)$, and $s(2,2)$, and by measuring the detected signals $d(0,0), d(0,1), d(1,0)$, and $d(2,3)$, we can estimate all the parameters. We note that we are judiciously choosing the location $(0,0)$ in the SLM array since any other location will result in non-causal measurements. At this point, we note that the sign of the translation parameters can be easily obtained by analyzing the dominant energy from the neighborhood pixels of the detector array. The following analysis can be trivially extended for different combinations of the negative and positive pairs of translation offsets.

We now formulate the equations for the detected signals $d(0,0), d(0,1), d(1,0)$, and $d(2,3)$ by presetting the transmitted bits as $s(0,0) = 1$ and $s(2,2) = 1$ and zero everywhere else. We choose the SLM pixel $s(2,2)$ since the effect of the blurring function corresponding to $s(0,0)$ is very close to zero in the neighborhood of the detector pixels affected by $s(2,2)$. We assume low SLM and CCD fill factors so that we can get rid of the integrals in (89) and evaluate the blurring functions at the center of the detector pixel. By setting $s(0,0) = 1$

and evaluating the integral in (89) at the center of the detector pixel, we have

$$\begin{aligned} y_1 = d(0,0) &= h^2\left(-\frac{1}{2}-a\right)h^2\left(-\frac{1}{2}-b\right)+\mathcal{N}(0,\sigma_v^2), && (93)\\ y_2 = d(1,0) &= h^2\left(\frac{1}{2}-a\right)h^2\left(-\frac{1}{2}-b\right)+\mathcal{N}(0,\sigma_v^2), && (94)\\ y_3 = d(0,1) &= h^2\left(-\frac{1}{2}-a\right)h^2\left(\frac{1}{2}-b\right)+\mathcal{N}(0,\sigma_v^2), && (95)\end{aligned}$$

where $h(x) = \frac{\sin(\pi x)}{\pi x}$. The detector noise is assumed to be zero mean additive white Gaussian noise with a variance σ_v^2. The blurring parameters a, b in the above set of equations can be obtained using the inverse transformation as

$$(a,b) = R^T\left(-\frac{1}{2},-\frac{1}{2}\right)-(\sigma_x,\sigma_y). \quad (96)$$

Similarly, we obtain the detected signal $d(2,3)$ as

$$y_4 = d(2,3) = h^2\left(\frac{3}{2}+\sigma_x-\frac{3}{2}(\sin(\alpha)+\cos(\alpha))\right)h^2\left(\frac{5}{2}+\sigma_y+\frac{3}{2}(\sin(\alpha)-\cos(\alpha))\right)+\mathcal{N}(0,\sigma_v^2). \quad (97)$$

It is clear that we cannot simultaneously solve for all the parameters by measuring y_1, y_2, and y_3, since one of them is redundant. We need a fourth measurement, like the one in equation (97). Let us define the auxiliary parameters μ_1 and μ_2 as

$$\begin{aligned} \mu_1 &= \cos(\alpha)+\sin(\alpha)+2\sigma_x && (98)\\ \mu_2 &= \cos(\alpha)-\sin(\alpha)+2\sigma_y. && (99)\end{aligned}$$

Using (98) and (99) in (93), (94), and (95) we have

$$\begin{aligned} y_1 &= h^2\left(\frac{1}{2}(-1+\mu_1)\right)h^2\left(\frac{1}{2}(-1+\mu_2)\right)+\mathcal{N}(0,\sigma_v^2) = f_1+\mathcal{N}(0,\sigma_v^2), && (100)\\ y_2 &= h^2\left(\frac{1}{2}(1+\mu_1)\right)h^2\left(\frac{1}{2}(-1+\mu_2)\right)+\mathcal{N}(0,\sigma_v^2) = f_2+\mathcal{N}(0,\sigma_v^2), && (101)\\ y_3 &= h^2\left(\frac{1}{2}(-1+\mu_1)\right)h^2\left(\frac{1}{2}(1+\mu_2)\right)+\mathcal{N}(0,\sigma_v^2) = f_3+\mathcal{N}(0,\sigma_v^2). && (102)\end{aligned}$$

Using the above set of equations, we will obtain the ML estimates of μ_1 and μ_2. By setting $s(0,0) = 1$, and making N independent measurements on y_1, the probability density function (pdf) of y_1 is obtained as

$$p_{Y_1}(y_1|\mu_1,\mu_2) = \prod_{i=1}^{N}\frac{1}{\sqrt{2\pi\sigma_v^2}}\exp\left(-\frac{(y_1(i)-f_1)^2}{2\sigma_v^2}\right). \quad (103)$$

By setting the partial derivates of the log likelihood function of (103) to zero, we have

$$\begin{aligned} \frac{\partial \ln p_{Y_1}(y_1|\mu_1,\mu_2)}{\partial \mu_1} &= 0 \Rightarrow f_1 = \frac{1}{N}\sum_{i=1}^{N} y_1(i) \\ \frac{\partial \ln p_{Y_1}(y_1|\mu_1,\mu_2)}{\partial \mu_2} &= 0 \Rightarrow f_1 = \frac{1}{N}\sum_{i=1}^{N} y_1(i). \qquad (104) \end{aligned}$$

Similarly, by obtaining the joint pdf of N measurements of y_2 and y_3 and doing an ML estimation, we obtain f_2 and f_3 as

$$\begin{aligned} f_2 &= \frac{1}{N}\sum_{i=1}^{N} y_2(i) \qquad (105) \\ f_3 &= \frac{1}{N}\sum_{i=1}^{N} y_3(i). \qquad (106) \end{aligned}$$

The estimates $\hat{\mu_1}$ and $\hat{\mu_2}$ can be obtained by solving the following equations

$$\begin{aligned} \frac{h^2\left(\frac{1}{2}(\hat{\mu_1}-1)\right)}{h^2\left(\frac{1}{2}(\hat{\mu_1}+1)\right)} &= \frac{\sum_{i=1}^{N} y_1(i)}{\sum_{i=1}^{N} y_2(i)} \qquad (107) \\ \frac{h^2\left(\frac{1}{2}(\hat{\mu_2}-1)\right)}{h^2\left(\frac{1}{2}(\hat{\mu_2}+1)\right)} &= \frac{\sum_{i=1}^{N} y_1(i)}{\sum_{i=1}^{N} y_3(i)}. \qquad (108) \end{aligned}$$

We note that the estimates $\hat{\mu_1}$ and $\hat{\mu_2}$ are asymptotically unbiased and efficient. We will now use the ML estimates for $\hat{\mu_1}$, $\hat{\mu_2}$ in y_4, and get an ML estimate for α. Expressing σ_x and σ_y in (98) and (99) in terms of $\hat{\mu_1}$, $\hat{\mu_2}$, and α and plugging it in (97), we have

$$\begin{aligned} \tilde{y_4} &= h^2\left(\frac{3}{2}+\frac{1}{2}\left(\hat{\mu_1}-4\cos(\alpha)-4\sin(\alpha)\right)\right)h^2\left(\frac{5}{2}+\frac{1}{2}\left(\hat{\mu_2}-4\cos(\alpha)+4\sin(\alpha)\right)\right)+\mathcal{N}(0,\sigma_v^2) \\ &= f_4+\mathcal{N}(0,\sigma_v^2). \qquad (109) \end{aligned}$$

Collecting N i.i.d samples of $\tilde{y_4}$, we obtain the joint pdf as

$$p_{\tilde{Y_4}}(\tilde{y_4}|\mu_1,\mu_2) = \prod_{i=1}^{N} \frac{1}{\sqrt{2\pi\sigma_v^2}} \exp\left(-\frac{(\tilde{y_4}(i)-f_4)^2}{2\sigma_v^2}\right). \qquad (110)$$

Again, doing an ML estimation for $\hat{\alpha}$, like (104), we numerically compute $\hat{\alpha}$ from the following equation.

$$h^2\left(\frac{3}{2}+\frac{1}{2}\left(\hat{\mu_1}-4\cos(\hat{\alpha})-4\sin(\hat{\alpha})\right)\right)h^2\left(\frac{5}{2}+\frac{1}{2}\left(\hat{\mu_2}-4\cos(\hat{\alpha})+4\sin(\hat{\alpha})\right)\right) = \frac{1}{N}\sum_{i=1}^{N}\tilde{y_4} \qquad (111)$$

We now derive the Cramer-Rao lower bound (CRLB) [37] for the error variance in estimating $\hat{\alpha}$. We note that $\hat{\alpha}$ is an unbiased estimate of α. The CRLB for unbiased estimators is given by

$$var(\hat{\alpha}|\alpha, \hat{\mu_1}, \hat{\mu_2}) \geq \frac{1}{-E\left(\frac{\partial^2 \ln p_{\tilde{Y_4}}(\tilde{y_4}|\hat{\mu_1}, \hat{\mu_2}, \alpha)}{\partial \alpha^2}\right)}. \tag{112}$$

Computing the denominator term we have,

$$\begin{aligned}
\frac{\partial \ln p_{\tilde{Y_4}}(\tilde{y_4}|\hat{\mu_1}, \hat{\mu_2}, \alpha)}{\partial \alpha} &= \frac{1}{\sigma_v^2}\frac{\partial f_4}{\partial \alpha}\left[\sum_{i=1}^{N}(\tilde{y_4}(i) - f_4)\right] & (113)\\
\frac{\partial^2 \ln p_{\tilde{Y_4}}(\tilde{y_4}|\hat{\mu_1}, \hat{\mu_2}, \alpha)}{\partial \alpha^2} &= \frac{-N}{\sigma_v^2}\left(\frac{\partial f_4}{\partial \alpha}\right)^2 + \frac{1}{\sigma_v^2}\frac{\partial^2 f_4}{\partial \alpha^2}\left[\sum_{i=1}^{N}(\tilde{y_4}(i) - f_4)\right] & (114)\\
E\left(\frac{\partial^2 \ln p_{\tilde{Y_4}}(\tilde{y_4}|\hat{\mu_1}, \hat{\mu_2}, \alpha)}{\partial \alpha^2}\right) &= \frac{-N}{\sigma_v^2}\left(\frac{\partial f_4}{\partial \alpha}\right)^2. & (115)
\end{aligned}$$

Computing $\frac{\partial f_4}{\partial \alpha}$ and using this result in (115), (112) can be simplified as

$$var(\hat{\alpha}|\alpha, \hat{\mu_1}, \hat{\mu_2}) \geq \frac{\sigma_v^2}{N}\frac{1}{(z_1(\alpha) + z_2(\alpha))^2}, \tag{116}$$

where

$$\begin{aligned}
z_1(\alpha) &= 2h^2(\eta_1(\alpha))\, h(\eta_2(\alpha))\frac{\eta_2'(\alpha)}{\eta_2(\alpha)}\left[\cos(\pi\eta_2(\alpha)) - h(\eta_2(\alpha))\right],\\
z_2(\alpha) &= 2h^2(\eta_2(\alpha))h(\eta_1(\alpha))\frac{\eta_1'(\alpha)}{\eta_1(\alpha)}\left[\cos(\pi\eta_1(\alpha)) - h(\eta_1(\alpha))\right],\\
\eta_1(\alpha) &= \frac{3}{2} + \frac{1}{2}\left(\hat{\mu_1} - 4\cos(\hat{\alpha}) - 4\sin(\hat{\alpha})\right),\\
\eta_2(\alpha) &= \frac{5}{2} + \frac{1}{2}\left(\hat{\mu_2} - 4\cos(\hat{\alpha}) + 4\sin(\hat{\alpha})\right),\\
\eta_1'(\alpha) &= 2\sin(\alpha) - 2\cos(\alpha),\\
\eta_2'(\alpha) &= 2\sin(\alpha) + 2\cos(\alpha). & (117)
\end{aligned}$$

For a given α and the estimates $\hat{\mu_1}$ and $\hat{\mu_2}$, the right hand side of (116) heads to zero as $N \to \infty$ with $z_1(\alpha) + z_2(\alpha) \neq 0$.

7.3 Signal Recovery: Scanning and Decoding

From the structure of equation (89), we observe that the system is fundamentally non-linear and anti-causal. The non-linearity is because of the cross terms in the squaring process. The non-causality arises because we cannot initiate a recursion without the knowledge of a few transmitted bits. To facilitate a recursion, we need to initialize the SLM pixels in the boundary layers to zero. To determine the number of such layers that are initialized to zero, we compute the coordinates of the SLM array that just exceeds the range of the detector array. Consider the column of pixels at the right most ends. The coordinate of the right top corner $(m, y)^T$ after rotational transform and translation is obtained as $(m\cos(\alpha) - y\sin(\alpha) + \sigma_x, m\sin(\alpha) + y\cos(\alpha) + \sigma_y)^T$. The condition where the ordinate y exceeds m is given by

$$m\sin(\alpha) + y\cos(\alpha) + \sigma_y > m. \tag{118}$$

From equation (118), we infer that we need to set the top $m - \left\lfloor \frac{m(1-\sin(\alpha))-\sigma_y}{\cos(\alpha)} \right\rfloor$ layer of SLM pixel bits to zero. Similarly, by symmetry, we set the bottom layer of $m - \left\lfloor \frac{m(1-\sin(\alpha))-\sigma_y}{\cos(\alpha)} \right\rfloor$ SLM pixels to zero. Proceeding along similar lines, the left and the rightmost $m - \left\lfloor \frac{m(1-\sin(\alpha))-\sigma_x}{\cos(\alpha)} \right\rfloor$ layer of SLM pixels are initialized to zero.

The recovery process is done in two blocks. The first block comprises of all the detector pixels towards the right-half plane of the detector array. The second block consists of all the pixels in the left-half plane. For the first block, the detector pixels are sequentially scanned starting from the topmost row until all the transmitted SLM bits are sequentially decoded from right to left along this row. The scanner moves to the next row and repeats the process of decoding all the row bits before starting from the next row. This process iterates until all the bits in the first block are decoded. This idea is illustrated in Figure 34. For the second block, the scanner starts from the bottommost row in the left-half plane, decodes all the bits from left to right along that row, moves to the next top row, and iterates the process until all the bits are recovered.

We can also decode by processing the array of detected signals in four different blocks corresponding to each of the four quadrants and then average the results. The averaging technique will be helpful in the presence of severe detector noise when decoding errors tend to propagate.

It is worthwhile to note that decoding from the edges is counter-intuitive to processing from the center since the bits at the center are more reliable to rotational artifacts. The reason for this strategy stems from the fact that any other signal processing recovery technique will be essentially anti-causal since the neighborhood of the intended pixel has dependencies on both the sides. Processing the information from the center will require manipulation of the encoding operation and a predetermined interleaving. To avoid these complications and overheads, we process the information from the edges. Using this strategy, we can reconstruct bits without any prior encoding or loss in performance. We now outline the steps for decoding the first block. The procedure is described below in the form of an algorithm. The decoding algorithm for the second block follows anti-symmetrically in exactly the same way as the first block.

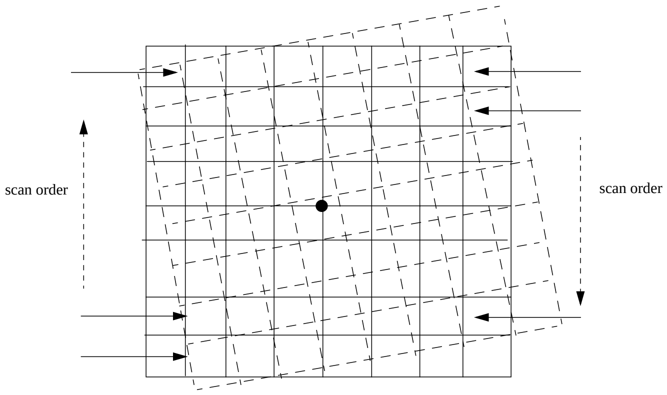


Figure 34: Scanning and decoding the CCD pixels.

<u>Outline of the Algorithm</u>

Introduce the following definitions:

s : Array representing the decoded bits.

d : Array holding the detected signal values.

Initialize:

- Obtain the estimates of the parameters $\hat{\alpha}, \hat{\sigma_x}$ and $\hat{\sigma_y}$.
- Set all the pixels of the array s along the rows from $\left\lfloor \frac{m(1-\sin(\hat{\alpha}))-\hat{\sigma_y}}{\cos(\hat{\alpha})} \right\rfloor$ to m as zero. Also, set all the pixels of the array s from columns $\left\lfloor \frac{m(1-\sin(\hat{\alpha}))-\hat{\sigma_x}}{\cos(\hat{\alpha})} \right\rfloor$ to m as zero.

Algorithm Steps:

1. Set the detector index to the top right corner $(r,c) = (m,m)$.

2. Obtain the SLM pixel indices that overlap with (r,c) as $(a,b), (a-1,b), (a,b-1)$, and $(a-1,b-1)$, where $(a,b) = \left(\lceil r\cos(\hat{\alpha}) + c\sin(\hat{\alpha}) - \hat{\sigma_x}\rceil, \lceil -r\sin(\hat{\alpha}) + c\cos(\hat{\alpha}) - \hat{\sigma_y}\rceil\right)$.

3. Evaluate the components of the kernel $g_{a-1,b}(x,y), g_{a,b}(x,y)$, and $g_{a,b-1}(x,y)$ at the center of the detector pixel $x = r - \frac{1}{2}, y = c - \frac{1}{2}$.

4. Obtain the component of the signal energy for the SLM pixel $(a-1,b-1)$ as $\sqrt{d(r,c)} - q$, where $q = g_{a,b}\sqrt{s(a,b)} - g_{a-1,b}\sqrt{s(a-1,b)} - g_{a,b-1}\sqrt{s(a,b-1)}$.

5. Compute $\gamma = \left(\frac{\sqrt{d(r,c)}-q}{g_{a-1,b-1}(r-\frac{1}{2},c-\frac{1}{2})}\right)^2$

6. If $\gamma \geq \tau_{th}$, decode $s(a-1,b-1) = 1$, else decode $s(a-1,b-1) = 0$.

7. $c \leftarrow c-1$. Loop back to Step 2 till $c = 1$.

8. $r \leftarrow r-1$. Loop back to Step 2 till $r = -m+2$.

We note that the decoding process is simple and the algorithm has no extra storage overheads. The time complexity of the algorithm is linear in the number of pixels decoded. We use threshold detection in Step 6 to circumvent the round off errors and for handling the detector noise. Since the decoded pixel value is either '0' or a '1' and is equally likely, the threshold τ_{th} is optimally set to $\frac{1}{2}$ in the high SNR regions.

7.3.1 Performance Limits and Error Analysis

From the recursive structure of the decoding algorithm, it is apparent that there will be error propagation effects especially when there is severe detector noise. However, these effects somewhat counterbalance the amount of residual energy available at the detector. In this section, we formulate equations for analyzing the error propagation dynamics of the algorithm.

Let t denote the sequential time index for decoding. The pixel decoded at time t is dependent on the decoded pixels at time instants $t-1, t-2t \bmod (m)$, and $t-2t \bmod (m)-1$. From step 5 of the algorithm, neglecting the fractional higher-order terms, we can obtain the decoded bit at time t as

$$\begin{aligned}\beta(t)\tilde{d}(t) &= \beta(t)d(t)+\beta(t-1)d(t-1)+\beta(t-t_1)d(t-t_1)+\beta(t-t_1-1)d(t-t_1-1)\\ &\quad -(\beta(t-1)\tilde{d}(t-1)+\beta(t-t_1)\tilde{d}(t-t_1)+\beta(t-t_1-1)\tilde{d}(t-t_1-1))\\ &\quad +N(0,\sigma_v^2) \qquad (119)\end{aligned}$$

where, $t_1 = 2t \bmod (m)$. The term $\beta(t)$ represents the time-varying component of the blurring function. Denoting the error by $e(t) = d(t) - \tilde{d}(t)$ and simplifying (119), we have

$$\beta(t)e(t)+\beta(t-1)e(t-1)+\beta(t-t_1)e(t-t_1)+\beta(t-t_1-1)e(t-t_1-1)+N(0,\sigma_v^2)=0 \quad (120)$$

Equation (120) implies that the error dynamics are non-linear and time-varying. This fact explains that significant bit error rates are possible when the noise variance is high. Since computing the probability of error analytically is intractable, we validate the results through simulations and explain the deviations based on the error dynamics. At high signal-to-noise ratios, the decoding errors are less. Hence, the probability of error is dependent mainly on the local noise threshold and less on the error propagation effects.

Before concluding this section, we would like to briefly comment on the choice of detectors and the asymptotic performance of detection algorithm. In order to make meaningful comparisons, we need to know the channel capacity in the first place. As pointed out before in section 2, the capacity of 2-D ISI channels is an open problem. There are a few detection strategies based on iterative techniques [31] for jointly decoding codes in the presence of 2-D channel ISI with a known time-invariant channel impulse response. These techniques cannot be formulated in a straightforward for time-varying ISI models like the rotational misregistration case. Constructing practical efficient finite-complexity detection algorithms for 2-D noisy ISI channels is an open research problem. Hence, in evaluating the performance of our algorithm, we compare our results against a baseline thresholding scheme to validate the necessity for a compensating technique. This will also serve as a benchmark for the detector performance.

7.4 Results and Discussion

In this section, we validate the performance of the estimators and the algorithm through simulations. We are comparing our algorithm with a baseline thresholding scheme without compensation since we are not aware of any other algorithm that handles fractional non-uniform misregistration.

Experiment: 1

In the first experiment, we simulate the performance of the estimators and compute the CRLB to validate the claims. $N = 10000$ samples of i.i.d measurements of y_1, y_2, y_3, and y_4 were collected in the presence of additive white Gaussian noise with $\sigma_v^2 = 0.001$. Table 13 shows the true and estimated values of α, σ_x and σ_y. It is clear that the estimators are asymptotically efficient and achieve the Cramer-Rao lower bounds. In practice, it might not be feasible to obtain 10000 measurements. Instead, the data can be read from a few hundred recorded holograms to get a rough estimate of the unknown parameter.

Table 13: Estimates of the parameters

$(\alpha, \sigma_x, \sigma_y)$	$(\hat{\alpha}, \hat{\sigma_x}, \hat{\sigma_y})$	`CRLB for` α
$(2, 0.25, 0.5)$	$(1.9824, 0.2505, 0.4999)$	1.513×10^{-5}
$(2.5, 0.3, 0.5)$	$(2.5382, 0.3002, 0.5003)$	1.669×10^{-5}

Experiment: 2

In the second experiment, we consider pure rotation. The parameters are $\alpha = 2, \sigma_x = 0$ and $\sigma_y = 0$. Several pages of 100×100 pixel arrays were modeled to mimic the detector output with 2 degrees of rotational misalignment. White Gaussian noise was added to the detector output. Numerically estimated values for $\hat{\alpha}$ were used in the decoding algorithm. The noise variance was varied to obtain different SNRs. Table 14 shows the average bit error rate (BER) versus signal-to-noise ratio after compensation. The above steps were repeated for different values of α. It is interesting to note that the decoding algorithm performs well with high SNR and is fairly robust with SNRs around 50dB. The error rate is certainly higher than that expected for equally likely binary symbols over an additive white Gaussian channel. This fact can be explained by the non-linear error propagation dynamics outlined in the last section. Reducing the crosstalk by compensation will increase the susceptance to noise. Interestingly, at very low SNRs the bit error rate seems to be rather not too high when compared to a binary communication over the AWGN channel. It appears that the residual energy from the neighbors somewhat counterbalances the noise effects in the low SNR region.

Table 14: SNR versus bit error rate

`SNR (dB)`	`BER` $(\alpha = 2)$	`BER` $(\alpha = 3)$
∞	0	0
60	0.0088	0.0085
50	0.0320	0.0330
40	0.0770	0.0760
30	0.2078	0.2078
20	0.2995	0.3024
1	0.4400	0.4824

Experiment: 3

In the third experiment, we consider rotation and translation. Several pages of 100 × 100 pixel arrays were modeled to mimic the detector output with the misalignment parameters outlined in Table 15. The setup for this experiment was done in exactly the same way as in the previous example. Table 15 shows the average BER versus SNR for the tuple $(\alpha, \sigma_x, \sigma_y)$. In this experiment, we compare the results with a simple baseline thresholding scheme to validate the need for misregistration compensation. The threshold for the baseline scheme without any misregistration compensation is chosen based on the mean of the detector intensity values.

It is clear from Table 15 that the misregistration compensation is needed for decoding bits resulting from interpixel crosstalk. A baseline scheme, such as a simple threshold detector will perform poorly since it does not have enough local statistics to decode the bits.

We would like to point out that high bit error rates can result due to sampling at the Nyquist frequency and can be overcome by oversampling at the SLM. But we limit our theoretical analysis to Nyquist rate sampling.

Table 15: SNR versus bit error rate

SNR (dB)	BER-Baseline $(2, \frac{1}{4}, \frac{1}{2})$	BER-Compensation $(2, \frac{1}{4}, \frac{1}{2})$	BER-Baseline $(2, \frac{1}{2}, \frac{1}{2})$	BER-Compensation $(2, \frac{1}{2}, \frac{1}{2})$
∞	0.3950	0	0.3950	0
60	0.3950	0.0120	0.4152	0.0142
50	0.4152	0.0316	0.4146	0.0338
40	0.4152	0.0804	0.4164	0.1020
30	0.4150	0.1986	0.4154	0.2166
20	0.4194	0.3176	0.4204	0.3230
1	0.4790	0.4394	0.4718	0.4408

7.5 Summary

The fractional pixel misregistration problem is frequently encountered in many imaging systems. Minor misalignment errors in rotation and translation lead to non-uniform fractional interpixel crosstalk requiring compensation. In this chapter, we formulated a channel model for handling translational and rotational misalignment for optical imaging systems like volume holographic memories and derived an upper bound on the number of recoverable bits. We derived maximum likelihood estimators for determining the unknown misalignment parameters and validated the efficiency of the estimators using Cramer-Rao bounds. We also developed a misregistration compensation algorithm and validated its performance through simulations and error propagation models. It is interesting to note that the algorithm performs well in the presence of detector noise. The effect of crosstalk somewhat counterbalances the effect of noise. When the crosstalk components are removed, the local statistic is prone to noise. This effect is unavoidable and is inherent in imaging systems with interpixel crosstalk. The recursive nature of the algorithm leads to decoding error propagation. This is primarily due to the serial structure of the algorithm. There is a natural trade-off between the gain in the SNR with crosstalk and the susceptibility to noise when crosstalk terms are compensated.

We note that bit error rates can be reduced by encoding information bits using powerful two-dimensional error correcting codes. There are a lot of fundamental problems associated with 2-D noisy interpixel interference channels. It would be interesting to analytically determine the 2-D channel capacity and develop powerful 2-D coding algorithms for maximizing the information rate over such channels. There are other open research issues that need to be addressed while designing 2-D detection algorithms. What is the best decoding algorithm that minimizes the average bit-error rate ? If so, are there bounds on the coding rate and the probability of error performance trade-off ? There are several other interesting problems in this area. Many of these problems cannot be solved by a simple extension of existing 1-D algorithms.

CHAPTER VIII

CONCLUSIONS AND FUTURE WORK

In this thesis, we dealt with constrained coding and signal processing aspects of holographic systems. Constrained coding in two-dimensions is an important open problem in mathematical physics with a wide range of applications from theoretical studies in lattice packings to finite automata theory. In the following section, we summarize the main contributions of our research.

8.1 Main Contributions

The following are the main results of this thesis.

- We derived bounds for the capacity of 2-D $(1, \infty, d, k)$ RLL constrained channels by extending existing ideas based on the adjacency construction. We proposed low complexity algorithms for writing valid arrays using an iterative approach.

- We derived bounds for the capacity of asymmetric $(d_1, \infty, d_2, \infty)$ and $(0, k_1, 0, k_2)$ binary RLL constrained channels. We deduced code constructions that achieve the derived capacity lower bounds.

- We generalized our ideas for computing the capacity bounds and extended the coding algorithms for asymmetric multi-level $(M, d_1, \infty, d_2, \infty)$ and $(M, 0, k_1, 0, k_2)$ 2-D RLL constrained channels. Our results are the first reported analytical bounds for these constraints. Our ideas can be extended further to a class of multi-dimensional multi-level RLL constraints.

- We derived a constructive bound for the capacity of holographic channels. We also analyzed the trade-off between the storage density and the multiplexing rate for holographic channels. This theoretical result strengthens the existing experimental framework for estimating the volumetric storage density in holographic memories.

- We developed a channel model for combined two-dimensional translational and rotational misregistration in holographic systems and proposed a signal recovery algorithm for interpixel interference cancellation. The theory can be extended to other optical imaging systems.

8.2 Future Work

There are a number of research challenges in the exciting world of 2-D constrained coding, error correction coding, and signal processing. Two-dimensional information-theoretic problems have wide spread applicability and impact in other interdisciplinary fields, such as computer science and physics. Some of these problems are considered quite hard to tackle. From a mathematical perspective, exactly computing the capacity and developing efficient coding algorithms for 2-D constrained channels is a contribution to symbolic dynamics. From a practical perspective, these results have applications in ultra-high capacity memories like volume holography and patterned media. We will sumarize some of the open problems in the field.

- Exactly computing the 2-D capacity of RLL constrained channels is an open problem. There are very few capacity bounds for 2-D finite (d, k) constraints. Most of the existing results are very loose bounds. A general theory needs to be developed for exactly computing the capacity of higher dimensional constrained channels.

- Computing the capacity for 2-D ISI channels is also an open problem. Algorithms based on iterative constructions are just capacity estimates. A clear analytical framework needs to be developed for this problem. The development of a theory for computing the capacity of 2-D ISI channels will advance our understanding for constructing codes and developing efficient detection algorithms for finite 2-D ISI channels.

- There are a number of other problems related to 2-D coding for spectral shaping. These problems are virtually unexplored. For example, there is no known technique for designing efficient 2-D higher-order spectral null constraints. Also, analyzing the 2-D power spectral density of modulation and error correcting codes is an interesting topic for further research.

- Developing new error correcting codes with modulation code properties is yet another challenging problem. No work is reported in this field at the moment.

- The development of signal processing techniques for holography is a self-contained topic in itself. Developing efficient maximum likelihood detectors and extending the framework of trellis type detection algorithms for two dimensions is a challenging problem. In our current work on pixel misregistration, we considered the case when the misalignment parameters are fixed but unknown. A general technique needs to be developed for efficiently handling time-varying misalignments due to material shrinkage effects. This can be more practically helpful for an engineer working on holographic memories.

To conclude, there are a number of rich theoretical problems in the field of multi-dimensional information theory that warrant further investigation. Many of these problems are unsolved and have deep consequences in other multi-disciplinary areas. It is certainly worthwhile to investigate these problems to further our understanding of higher-dimensional channels and systems.

APPENDIX A

IMPROVED ENUMERATION TYPE BOUNDS FOR 2-D $(0,1)$ AND $(1,\infty)$ RLL CONSTRAINTS

In this appendix, we present an analysis for obtaining slightly improved lower bounds for 2-D RLL constraints. We derive these bounds based on the tiling algorithms presented in Chapter 4.

Proposition A.1. *The capacity of* $(0,1,0,1)$ *constraints can be further lower bounded as*

$$C_{(0,1,0,1)} \geq \sup_{p\in[0,1]} \left[\tfrac{h(p)}{1+2p} + \sum_{s=2}^{\infty} \tfrac{(1-p)^2 p^{s+1}}{1+p+(2s+1)p^{s+1}(1-p)^2} \left(\tfrac{h(p)}{1+p}\right) \right].$$

Before we begin with the proof of the proposition, we would like to highlight our motive for working on this derivation. The bounds presented in Theorem 5.1 and Theorem 5.2 are tight within 0.5% for values of k greater than 5. Since our approach for computing capacity is constructive, we are interested in further tightening the lower bound to realize improved code rates.

We would like to recall the steps while deriving Theorem 5.1. We place a one for every k consecutive zeros occurring horizontally. Along the vacant spaces created in subsequent columns, we write valid sequences from a 1-D graph. Consider the $(0,1,0,1)$ constraint. While writing along the vacant positions within the 2^{nd} column using the constrained graph $\mathcal{G}_{(0,1)}$, two or more consecutive zeros can never occur. Suppose after horizontal bit stuffing of a one, we have a sequence of the type $x1x1x1x1...$, where x is a vacant space of unit length. (A vacant space is space where no bit has previously been written. It is the vacant space between two consecutive ones in this case.) We can introduce additional zeros in between the ones and write sequences from $\mathcal{G}_{(0,1)}$ along the remaining vacant spaces. We note that the new set of sequences generated by this process could never have been constructed by just writing sequences from a $(0,1)$ constrained graph along the vacant spaces. By enumerating these patterns, we gain entropy rate.

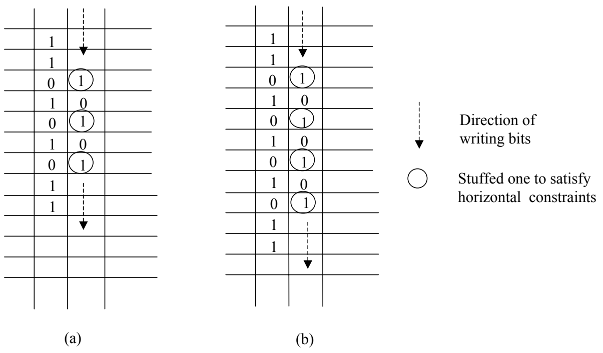

1 1 0 1 0 1 0 1 1
1 0 1 0 1
(a)
1 1 0 1 0 1 0 1 0 1 1
1 0 1 0 1 0 1
(b)
Direction of writing bits
Stuffed one to satisfy horizontal constraints

Figure 35: Additional valid configurations for generating $(M, 0, 1)$ arrays. (a) Sequence $\{110101011\}$ occurs in any column, a complementary sequence $xx10101xx$ (x can be a zero or a one) is placed in the next column (b) Sequence $\{11010101011\}$ occurs in any column, and a complementary sequence $xx1010101xx$ is placed in the next column.

Figure 35 illustrates the idea. In Figure 35 (a), the sequence $\{110101011\}$ occurs in the first column. A complementary sequence $xx10101xx$ can be placed in the next column and the bits can be written along the vacant spaces by doing a random walk on $\mathcal{G}_{(0,1)}$. Similarly, Figure 35 (b) illustrates the case for the sequence $\{11010101011\}$. In general, we consider sequences of the type $\{1101010.....11\}$ generated by $\mathcal{G}_{(0,1)}$, write complementary sequences in the next column, as shown in Figure 35, and then write sequences by doing a random walk on $\mathcal{G}_{(0,1)}$ along the vacant spaces. Using this idea, we will develop the proof of the proposition.

Proof. By stuffing a one for every zero that occurs in the previous column of the same row, the 2-D rate can be computed using Theorem 4.4 as

$$R_1 = \frac{h(p)}{1+2p}. \tag{121}$$

After doing a random walk on $\mathcal{G}_{(0,1)}$, we observe the occurrences of the subsequences of the type $E_2 : \{110101011\}$ as shown in Figure 35 (a). Using the adjacency matrix structure outlined in section 4.6, the probability of the subsequence $\{110101011\}$ can be obtained as

$$p_2 = [\pi_0(1-p) + \pi_1]p^3(1-p)^2. \tag{122}$$

We focus on a smaller sequence $\{01010\}$ nested within the subsequence $\{110101011\}$. For every zero occurring in the subsequence 01010, we stuff a one to the right, as shown in Figure 35. This transforms the pattern in the second column as $1x1x1$. We stuff 2 consecutive zeros which further transforms the pattern $1x1x1$ as 10101. We note that this particular configuration of writing 'two' consecutive zeros could never have occurred by a random walk on $\mathcal{G}_{(0,1)}$ on the available vacant spaces. After these steps, for the remaining vacant positions we do a random walk on $\mathcal{G}_{(0,1)}$. Since effectively 5 bits are lost for every occurrence of the sequence $\{110101011\}$, the expected free space $f_s^{(2)}$ for writing the bits along the vacant positions is given by the recursion

$$E(f_s^{(2)}) = m - 5p_2E(f_s^{(2)}). \tag{123}$$

Let R_m be the entropy rate of the 1-D $(0,1)$ constraint. We can compute R_m as

$$R_m = \pi_0 h(p), \tag{124}$$

where $h(p)$ is the binary entropy function and $\pi_0 = \frac{1}{p+1}$.

Using (122), (123), and (124) the additional 2-D entropy $\delta R^{(2)}$ gained due to the event E_2 is given by

$$\delta R^{(2)} = p_2 \frac{E(f_s^{(2)}}{m} R_m. \tag{125}$$

We continue this process for all the other subsequences of the type $\{11(01)^s1\}$. It is clear that a subsequence of the type $\{11(01)^s1\}$ is not a subsequence of $\{11(01)^{\tilde{s}}1\}$ for some value of $\tilde{s} \neq s$. Thus, the entropy obtained by considering such distinct subsequences is additive. Now, for every s zeros occurring in the the subsequence $\{11(01)^s1\}$, $2s+1$ bits are effectively lost. By proceeding in the same way as in (125), the increase in the 2-D entropy rate is given by

$$\delta R(p) = \sum_{s=2}^{\infty} \frac{(1-p)^2 p^{s+1}}{1+p+(2s+1)p^{s+1}(1-p)^2} \left(\frac{h(p)}{1+p} \right). \tag{126}$$

Using (121 and (126), and maximizing the overall 2-D entropy rate over all choices of p, we get

$$R = \sup_{p \in [0,1]} [R_1(p) + \delta R(p)]. \tag{127}$$

Equation (127) is a constructive lower bound for the 2-D capacity of the $(0, 1, 0, 1)$ constraints. This proves the proposition. □

Computing the bound in the Proposition A.1, we get an improved lower bound as 0.5632.

The 1-D constrained graphs $\mathcal{G}_{(0,1)}$ and $\mathcal{G}_{(1,\infty)}$ are isomorphic to each other. In other words, one graph can be realized from the other by complementing the bits along the edges. Thus, Proposition A.1 holds true for the $(1, \infty, 1, \infty)$ constraint as well.

We note that the analysis can be extended for the M-ary case.

APPENDIX B

COMPUTING DETECTOR EFFICIENCY

In this appendix, we derive an expression for the number of recoverable bits from misalignments. The basic idea is to find the overlapping areas between the SLM and CCD arrays using elementary coordinate geometry for computing the detector efficiency.

Let us fix the CCD coordinate system first and then obtain the SLM coordinates. We are assuming that the SLM is at a positive angle α with respect to the detector and is shifted by $(\pm\sigma_x, \pm\sigma_y)^T$ with respect to the optical center, as shown in Figure 36. Any point $(x, y)^T$ on the CCD plane is mapped to $R(x, y)^T + (\pm\sigma_x, \pm\sigma_y)^T$ on the SLM. Let the optical center of the CCD correspond to the coordinate $(0, 0)$. The four corners i.e., the right-top, the left-top, the left-bottom and the right-bottom corners of the CCD array correspond to the coordinates $(m, m)^T$, $(-m, m)^T$, $(-m, -m)^T$, and $(m, -m)^T$ respectively. Let us denote these four corners as a, b, c and d respectively. After transformation, these points will be mapped as:

$$\begin{aligned}
A &= (m\cos(\alpha) - m\sin(\alpha) \pm \sigma_x, m\sin(\alpha)m\cos(\alpha) \pm \sigma_y)^T \\
B &= (-m\cos(\alpha) - m\sin(\alpha) \pm \sigma_x, -m\sin(\alpha) + m\cos(\alpha) \pm \sigma_y)^T \\
C &= (-m\cos(\alpha) + m\sin(\alpha) \pm \sigma_x, -m\sin(\alpha) - m\cos(\alpha) \pm \sigma_y)^T \\
D &= (m\cos(\alpha) + m\sin(\alpha) \pm \sigma_x, m\sin(\alpha) - m\cos(\alpha) \pm \sigma_y)^T. \qquad (128)
\end{aligned}$$

The line segment AB intersects the line segments ab and bc at points p and q respectively. Similarly, the points of intersection r, s, t, u, v, and w can be obtained, as shown in Figure 36. The coordinates of the points p and q are obtained as follows. From elementary coordinate geometry, the equation of line AB is given by

$$y - m\sin(\alpha) - m\cos(\alpha) \mp \sigma_y = \tan(\alpha)(x - m\cos(\alpha) + m\sin(\alpha) \mp \sigma_x). \qquad (129)$$

Since the abscissa of q is $-m$, the ordinate can be obtained by plugging $x = -m$ in (129).

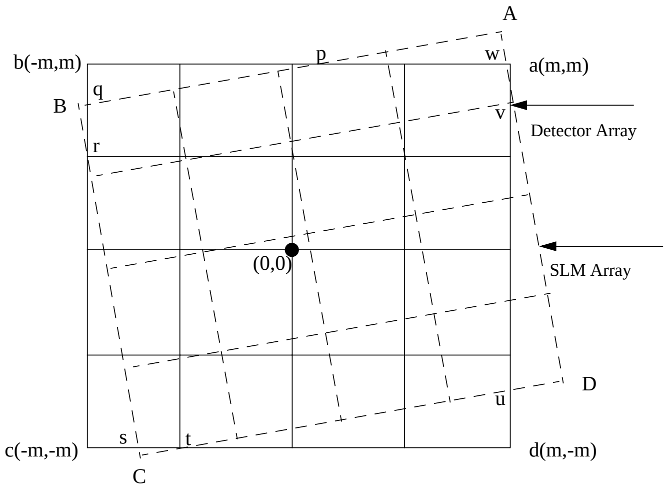


Figure 36: Coordinate system for SLM and CCD planes..

Similarly, the ordinate of p is m and the abscissa can be obtained by plugging $y = m$ in (129). We obtain the coordinates of p and q as

$$\begin{aligned} p &= (m(1+\cot(\alpha)-\cos(\alpha)\cot(\alpha)-\sin(\alpha)) \mp \cot(\alpha) \pm \sigma_x, m)^T \\ q &= (-m, m(\cos(\alpha)-\tan(\alpha)+\tan(\alpha)\sin(\alpha)) \pm \sigma_y \mp \sigma_x \tan(\alpha))^T. \end{aligned} \tag{130}$$

Using (130), we can obtain the area of the triangle $\triangle pbq$ as

$$\begin{aligned} \triangle_1 &= \frac{1}{2}m^2\left(1+\cot(\alpha)-\cos(\alpha)\cot(\alpha)-\sin(\alpha)+\frac{\pm\sigma_x \mp \sigma_y\cot(\alpha)}{m}\right) \\ &\quad \left(1-\cos(\alpha)+\tan(\alpha)-\sin(\alpha)\tan(\alpha)+\frac{\mp\sigma_y \pm \sigma_x\tan(\alpha)}{m}\right). \end{aligned} \tag{131}$$

Proceeding in the same way, we obtain the areas of the triangles $\triangle rcs$, $\triangle udt$, and $\triangle vaw$ as $\triangle_2, \triangle_3$, and $\triangle_4$ respectively given by

$$\begin{aligned} \triangle_2 &= \frac{1}{2}m^2\left(1+\cot(\alpha)-\cos(\alpha)\cot(\alpha)-\sin(\alpha)+\frac{\pm\sigma_x\cot(\alpha) \pm \sigma_y}{m}\right) \\ &\quad \left(1-\cos(\alpha)+\tan(\alpha)-\sin(\alpha)\tan(\alpha)+\frac{\pm\sigma_y\tan(\alpha) \pm \sigma_x}{m}\right) \\ \triangle_3 &= \frac{1}{2}m^2\left(1+\cot(\alpha)-\cos(\alpha)\cot(\alpha)-\sin(\alpha)+\frac{\mp\sigma_x \pm \sigma_y\cot(\alpha)}{m}\right) \\ &\quad \left(1-\cos(\alpha)+\tan(\alpha)-\sin(\alpha)\tan(\alpha)+\frac{\pm\sigma_y \mp \sigma_x\tan(\alpha)}{m}\right). \\ \triangle_4 &= \frac{1}{2}m^2\left(1+\cot(\alpha)-\cos(\alpha)\cot(\alpha)-\sin(\alpha)+\frac{\mp\sigma_x\cot(\alpha) \mp \sigma_y}{m}\right) \\ &\quad \left(1-\cos(\alpha)+\tan(\alpha)-\sin(\alpha)\tan(\alpha)+\frac{\mp\sigma_y\tan(\alpha) \mp \sigma_x}{m}\right) \end{aligned} \tag{132}$$

The fractional area $\frac{A_t - A_d \bigcap A_t}{A_t}$ where the channel containing the transmitted bits is effectively lost is given by

$$T_{loss} = \frac{\sum_{i=1}^{4} \triangle_i}{4m^2}. \tag{133}$$

Using (131) and (132) in (133), and simplifying we get,

$$T_{loss} = \frac{1}{2} f(\alpha) g(\alpha) + \frac{1}{4}[\epsilon_1 \epsilon_2 + \epsilon_3 \epsilon_4] \tag{134}$$

where,

$$\begin{aligned} f(\alpha) &= 1 + \cot(\alpha) - \sin(\alpha) - \cot(\alpha)\cos(\alpha) \\ g(\alpha) &= 1 + \tan(\alpha) - \cos(\alpha) - \tan(\alpha)\sin(\alpha) \\ \epsilon_1 &= \frac{\pm\sigma_x \mp \sigma_y \cot(\alpha)}{m} \\ \epsilon_2 &= \frac{\pm\sigma_x \tan(\alpha) \mp \sigma_y}{m} \\ \epsilon_3 &= \frac{\pm\sigma_x \cot(\alpha) \pm \sigma_y}{m} \\ \epsilon_4 &= \frac{\pm\sigma_x \pm \sigma_y \tan(\alpha)}{m} \end{aligned} \tag{135}$$

In the limit when $m \rightarrow \infty$, $\epsilon_1, \epsilon_2, \epsilon_3, \epsilon_4 \rightarrow 0$, the loss is chiefly governed by the term $\frac{1}{2} f(\alpha) g(\alpha)$.

REFERENCES

[1] Abdel-Ghaffar, K. A. S., McEliece, R. J., and van Tilborg, H. C. K., "Two-dimensional burst identification codes and their use in burst correction," *IEEE. Trans. Inform. Theory*, vol. 34, pp. 494–504, May 1988.

[2] Adler, R. L., Coppersmith, D., and Hassner, M., "Algorithms for sliding block codes - an application of symbolic dynamics to information theory," *IEEE. Trans. Inform. Theory*, vol. 29, pp. 5–22, Jan. 1983.

[3] Ashley, J., Bernal, M.-P., Burr, G. W., Coufal, H., Guenther, H., Hoffnagle, J. A., Jefferson, C. M., Marcus, B., Macfarlane, R. M., Shelby, R. M., and Sincerbox, G. T., "Holographic data storage," *IBM. J. Res. Develop.*, vol. 44, pp. 341–368, May 2000.

[4] Ashley, J. and Marcus, B., "Two-dimensional lowpass filtering codes for holographic storage," *IEEE. Trans. Commun.*, vol. 46, pp. 724–727, June 1998.

[5] Barry, J. R., Lee, E. A., and Messerschmitt, D. G., *Digital Communication.* Boston: Kluwer Academic Press, third ed., 2004.

[6] Baxter, R. J., "Dimers on a rectangular lattice," *J. Math. Phys.*, vol. 9, pp. 650–654, Nov. 1968.

[7] Bender, P. and Wolf, J. K., "A universal algorithm for generating optimal and nearly-optimal runlength-limited, charge constrained binary sequences," in *Proc. Intl. Symp. Inform. Theory.*, (San Antonio, TX), p. 6, IEEE, Jan. 1993.

[8] Blaum, M., Bruck, J., and Vardy, A., "Interleaving schemes for multi-dimensional cluster errors," *IEEE. Trans. Inform. Theory*, vol. 44, pp. 730–743, Mar. 1998.

[9] Blaum, M. and Farrell, P. G., "Array codes for cluster-error protection," *Electron. Lett.*, vol. 30, no. 21, 1994.

[10] Burr, G. W., Coufal, H., Hoffnagle, J. A., Jefferson, C. M., Jurich, M., Macfarlane, R. M., and Shelby, R. M., "High-density and high-capacity holographic data storage," *Asian. J. Phys*, vol. 10, pp. 1–28, Jan. 2001.

[11] Burr, G. W. and Weiss, T., "Compensation for pixel misregistration in volume holographic storage," *Opt. Lett.*, vol. 26, pp. 542–544, Apr. 2001.

[12] Calkin, N. J. and Wilf, H. S., "The number of independent sets in a grid graph," *SIAM J. Disc. Math.*, vol. 11, pp. 54–60, 1998.

[13] Chugg, K. M., Chen, X., and Neifeld, M. A., "Two-dimensional equalization in coherent and incoherent page-oriented optical memory," *J. Opt. Soc. Am. A*, vol. 16, pp. 549–562, Mar. 1999.

[14] Coufal, H. J., Psaltis, D., and Sincerbox, G., *Holographic Data Storage.* New York: Springer-Verlag, 2000.

[15] Cover, T. M. and Thomas, J. A., *Elements of Information Theory.* New York: John Wiley and Sons, Inc., first ed., 1991.

[16] Engel, K., "On the fibonacci number of an m x n lattice," *Fibonacci Quarterly*, vol. 28, pp. 72–78, 1990.

[17] Etzion, T. and Vardy, A., "Cascading methods for runlength-limited arrays," *IEEE. Trans. Inform. Theory*, vol. 43, pp. 319–324, Jan. 1997.

[18] Etzion, T. and Vardy, A., "Two-dimensional interleaving schemes with repetitions: constructions and bounds," *IEEE. Trans. Inform. Theory*, vol. 48, pp. 428–457, Feb. 2002.

[19] Forchhammer, S. and Justesen, J., "Entropy bounds for constrained two-dimensional random fields," *IEEE. Trans. Inform. Theory*, vol. 45, pp. 118–127, Jan. 1999.

[20] Gallager, R. G., *Low density parity check codes.* PhD dissertation, Massachussets Institute of Technology, Cambridge, 1963.

[21] Halevy, S., Chen, J., Roth, R. M., Siegel, P. H., and Wolf, J. K., "Improved bit-stuffing bounds on two-dimensional constraints," *IEEE. Trans. Inform. Theory*, vol. 50, pp. 824–838, May 2004.

[22] Heanue, J. F., Bashaw, M. C., and Hesselink, L., "Channel codes for digital holographic storage," *J. Opt. Soc. Am.*, vol. 12, pp. 2432–2439, Nov. 1995.

[23] Heanue, J. F., Gurkan, K., and Hesselink, L., "Signal detection for page-access optical memories with intersymbol interference," *Appl. Opt.*, vol. 35, pp. 2431–2438, May 1996.

[24] Imai, H., "Two-dimensional fire codes," *IEEE. Trans. Inform. Theory*, vol. 19, pp. 796–806, Nov. 1973.

[25] Immink, K. A. S., Siegel, P. H., and Wolf, J. K., "Codes for digital recorders," *IEEE. Trans. Inform. Theory*, vol. 44, pp. 2260–2299, Oct. 1998.

[26] Kamabe, H., "Two-dimensional codes for second order spectral null constraints," in *Proc. Intl. Symp. Inform. Theory*, (Sorrento), p. 308, IEEE, June 2000.

[27] Kato, A. and Zeger, K., "On the capacity of two-dimensional run-length constrained channels," *IEEE. Trans. Inform. Theory*, vol. 45, pp. 1527–1540, July 1999.

[28] Keskinoz, M. and Kumar, B. V. K. V., "Discrete magnitude-squared channel modeling and equalization and detection for volume holographic channels," *Appl. Opt.*, vol. 43, pp. 1368–1378, Feb. 2004.

[29] Lin, S. and Costello, D. J., *Error Control Coding: Fundamentals and Applications.* Englewood Cliffs, New Jersey: Prentice-Hall, Inc., second ed., 2004.

[30] Lind, D. and Marcus, B., *An Introduction to Symbolic Dynamics and Coding.* New York: Cambridge University Press, first ed., 1995.

[31] Marrow, M. and Wolf, J. K., "Iterative detection of 2-dimensional isi channels," in *Proc. Information Theory Workshop*, (Paris), pp. 131–134, IEEE, Mar. 2003.

[32] Menetrier, L. and Burr, G. W., "Density implications of shift compensation post processing in holographic storage systems," *Appl. Opt.*, vol. 42, pp. 845–860, Feb. 2003.

[33] Mok, F. H., Burr, G. W., and Psaltis, D., "System metric for holographic memory systems," *Opt. Lett*, vol. 21, pp. 896–898, June 1996.

[34] Moser, C. and Psaltis, D., "Holographic memory with localized recording," *Appl. Opt*, vol. 40, pp. 3909–3914, Aug. 2001.

[35] Nagy, Z. and Zeger, K., "Asymptotic capacity of two-dimensional channels with checkerboard constraints," *IEEE. Trans. Inform. Theory*, vol. 49, pp. 2115–2125, Sept. 2003.

[36] Ordentlich, E. and Roth, R. M., "Two-dimensional weight-constrained codes through enumeration bounds," *IEEE. Trans. Inform. Theory*, vol. 46, pp. 1292–1301, July 2000.

[37] Poor, H. V., *An Introduction to Signal Detection and Estimation.* New York: Springer-Verlag, second ed., 1994.

[38] Robinson, D. and Milanfar, P., "Fundamental performance limits in image registration," *IEEE. Trans. Image. Proc.*, vol. 13, pp. 1185–1199, Sept. 2004.

[39] Roth, R. M., Siegel, P. H., and Wolf, J. K., "Efficient coding schemes for the hard-square model," *IEEE. Trans. Inform. Theory*, vol. 47, pp. 1166–1176, Mar. 2001.

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

[41] Siegel, P. H. and Wolf, J. K., "Bit-stuffing bounds on the capacity of two-dimensional constrained arrays," in *Proc. Intl. Symp. Inform. Theory.*, (Cambridge, MA), p. 323, IEEE, Aug. 1998.

[42] Soriaga, J. B., Siegel, P. H., and Wolf, J. K., "On achievable rates of multistage decoding on two-dimensional isi channels," in *Proc. Intl. Symp. Inform. Theory*, (Adelaide), pp. 1348–1352, IEEE, Sept. 2005.

[43] Srinivasa, S. G. and McLaughlin, S. W., "Capacity bounds and coding algorithms for two-dimensional asymmetric (d, ∞) and $(0, k)$ runlength-limited channels," *In Review IEEE. Trans. Inform. Theory.*

[44] Srinivasa, S. G. and McLaughlin, S. W., "On translational and rotational misregistration: Signal reconstruction algorithms and performance limits," *In Review IEEE. Trans. Sig. Proc.*

[45] Srinivasa, S. G. and McLaughlin, S. W., "Algorithms for constructing a class of $(1, \infty, d, k)$ codes and estimates for capacity," in *Proc. Allerton Conference on Computers, Communication and Control*, (Monticello, IL), pp. 867–875, Univ. Illinois Press, Oct. 2003.

[46] Srinivasa, S. G. and McLaughlin, S. W., "Enumeration algorithms for constructing $(d_1, \infty, d_2, \infty)$ runlength-limited arrays: Capacity bounds and coding schemes," in *Proc. Information Theory Workshop*, (San Antonio, TX), pp. 141–146, IEEE, Oct. 2004.

[47] Srinivasa, S. G. and McLaughlin, S. W., "Capacity bounds and coding schemes for two-dimensional asymmetric k-constrained runlength-limited arrays," in *Proc. Allerton Conf. on Computers, Communication and Control*, (Monticello, IL), Univ. Illinois Press, Oct. 2005.

[48] Srinivasa, S. G. and McLaughlin, S. W., "Signal recovery due to rotational pixel misalignment," in *Proc. Intl. Conf. Acoust. Speech and Signal Proc.*, (Philadelphia, PA), pp. 121–124, IEEE, July 2005.

[49] Srinivasa, S. G. and McLaughlin, S. W., "Capacity lower bounds for two-dimensional m-ary (d, ∞) and $(0, k)$ runlength-limited arrays: Capacity bounds and coding schemes," in *Accepted in Intl. Symp. on Inform. Theory.*, (Seattle, WA), IEEE, July 2006.

[50] Srinivasa, S. G., Momtahan, O., Karbaschi, A., McLaughlin, S. W., Adibi, A., and Fekri, F., "Volumetric storage limits and space-volume multiplexing tradeoffs in holographic channels," *In Review Appl. Optics.*

[51] Srinivasa, S. G., Momtahan, O., Karbaschi, A., McLaughlin, S. W., Adibi, A., and Fekri, F., "M-ary, binary, and space-volume multiplexing trade-offs for holographic channels," in *Accepted in Globecom'06.*, (San Fransisco, CA), IEEE, Nov. 2006.

[52] Talyansky, R., Etzion, T., and Roth, R. M., "Efficient code constructions for certain two-dimensional constraints," *IEEE Trans Inform Theory*, vol. 45, pp. 794–799, Mar. 1999.

[53] Vadde, V. and Kumar, B. V. K. V., "Channel modeling and estimation for intrapage equalization in pixel-matched volume holographic data storage," *Appl. Opt.*, vol. 38, pp. 4374–4386, May 1999.

[54] Wachmann, U., Huber, J. B., and Schramm, P., "Comparison of coded modulation schemes for the awgn and the rayleigh fading channel," in *Proc. Intl. Symp. Inform. Theory*, (Cambridge), p. 5, IEEE, Aug. 1998.

[55] Wachsmann, U., Fisher, R. F. H., and Huber, J. B., "Multilevel codes: Theoretical concepts and practical design rules," *IEEE. Trans. Inform. Theory*, vol. 45, pp. 1361–1389, July 1999.

[56] Weeks, W. and Blahut, R. E., "The capacity and coding gain of certain checkerboard codes," *IEEE. Trans. Inform. Theory*, vol. 44, pp. 1193–1203, May 1998.

[57] Yang, Y., Adibi, A., and Psaltis, D., "Comparison of transmission and 90-degree holographic recording geometry," *Appl. Opt.*, vol. 42, pp. 3418–3427, June 2003.

VITA

Shayan Garani Srinivasa was born in Mysore and brought up in Bangalore, India. He received his Bachelor's degree in electronics and communication engineering from the University of Mysore, India in Aug 1997 and the Masters degree from the University of Florida, Gainesville in May 2001. Since Spring 2002, he has been working with Prof. McLaughlin on two-dimensional coding and signal processing algorithms for holographic memories.

Shayan held industry positions prior to joining his Ph.D. He worked with Infosys Technologies Limited from 1997-1999 as a software design engineer, developing communications software for Nortel DMS-100 switching circuits. He also worked with Broadcom Corporation as a senior design engineer during 2001-2002 and developed efficient video decoding algorithms for the PVR feature. During the summer of 2005, he worked as a research intern at Western Digital Inc. on post-ECC modeling of magnetic recording channels.

His academic research interests are in the broad areas of communication and signal processing, application of information and coding theory for data storage and transmission, neural networks, and recently in quantum information processing. Outside academics, he is a carnatic classical vocalist and actively performs in organized south-indian music concerts. His other hobbies include traveling, badminton, and Sanskrit literature.