CIRCULAR CODING IN HALFTONE IMAGES AND OTHER DIGITAL IMAGING PROBLEMS

by

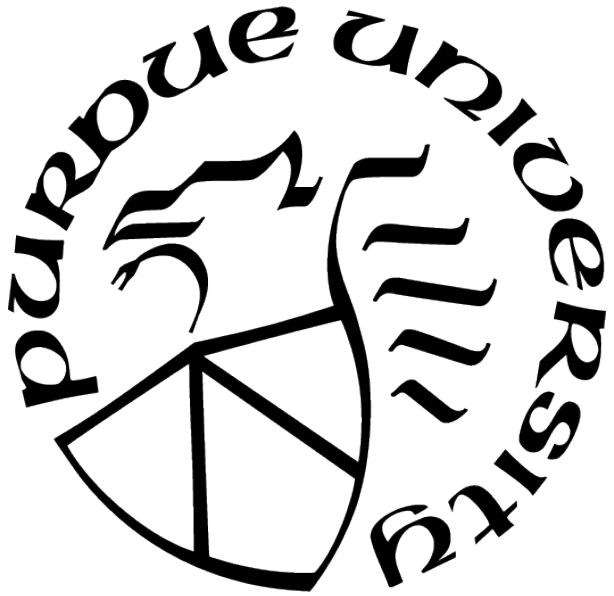
Yufang Sun

A Dissertation

Submitted to the Faculty of Purdue University

In Partial Fulfillment of the Requirements for the degree of

Doctor of Philosophy



School of Electrical and Computer Engineering

West Lafayette, Indiana

August 2021

THE PURDUE UNIVERSITY GRADUATE SCHOOL
STATEMENT OF COMMITTEE APPROVAL

Dr. Jan P. Allebach, Chair

School of Electrical Computer Engineering

Dr. Amy Reibman

School of Electrical Computer Engineering

Dr. Mark J.T. Smith

Dean of the Graduate School Senior Vice Provost for Academic Affairs and Department of Electrical and Computer Engineering University of Texas at Austin, TX

Dr. Robert Ulichney

School of Electrical and Computer Engineering

Approved by:

Dr. Dimitrios Peroulis

To my parents and family.

ACKNOWLEDGMENTS

Before starting my dissertation, I would like to acknowledge everyone important to me during my Ph.D. journey at Purdue University.

First and foremost, I would like to express my sincere gratitude to my advisor Prof. Jan P. Allebach for his continuous support. He always encourages me when I have difficulties. What we are exploring is quite new, so it is very common to encounter challenges. That is why we are here to research, and in hope that it can be beneficial for the upcoming researchers. With his strong support, I can take time to rethink the methodologies and find new clues from literature, with peace of mind. He provides me tremendous useful guidance, also recommends me extra resources, which help me to avoid detours.

I would also like to acknowledge my advisory committee members: Prof. Reibman Amy, Prof. Mark J.T. Smith, and Dr. Robert Ulichney. I would like to thank Prof. Reibman for her valuable guidance on the presentation, she helps me to explain the profound theories in simple languages and clarify my research claims with the global picture in mind, and also without losing the focus on technical details. I also would like to thank Dr. Ulichney for his detailed suggestion through my research and dissertation writing. He gives me timely correction when I make mistakes. Besides, I would like to thank Prof. Mark J.T. Smith providing me with new perspectives to level up my research work.

Thirdly, I would like to thank all the EISL members who have been working with me over the past few years. I enjoy the time we were attending the same classes, collaborating on projects and discussing research problems. The time flies away but the cherished memory stays.

Moreover, I would like to thank Hewlett-Packard Company for supporting my Ph.D. research. Also, I would like to thank Purdue University and Department of Electrical and Computer Engineering, and all the professors and staff members that helped me to finish my Ph.D. study.

TABLE OF CONTENTS

Page

Page

LIST OF TABLES

LIST OF FIGURES

LIST OF SYMBOLS

B	number of bits in the payload
$\boldsymbol{P}$	payload
$\boldsymbol{S}$	standard form of payload
C	number of circular shift to get the payload
c	number of bits needed to represent the circular shift
V	phase code row interleave period
D	row to row shift
$\boldsymbol{W}(W, H)$	crop window of data array
H	crop window height
W	crop window weight
$\boldsymbol{U}$	phase code representation of C
$\tilde{\boldsymbol{P}}$	candidate recovered payload
$\tilde{\boldsymbol{U}}$	candidate recovered phase
$T(k, l)$	halftone screening array
$\hat{\boldsymbol{P}}$	candidate recovered payload
$\hat{\boldsymbol{U}}$	candidate recovered phase
$T(k, l)$	halftone screening array
$\mathcal{B}$	total number of bit occurrence within a crop window
$\mathcal{B}^+$	number of majority occurrence of bits
$\mathcal{B}^-$	number of minority occurrence of bits
$\mathcal{M}$	sub-sampling mask
m, n	bit location index in data array
$\mathcal{P}(m, n)$	bit position index at location (m, n) in the data array
$\Delta m, \Delta n$	number of bit location shift
$f(\Delta m, \Delta n)$	bit position shift for location shift.
$\mathbf{W}$	crop window of data array
$q(h, j)$	column index for row h and bit position index j
μ	uncertainty calculated for each bit in the payload decoding

$\mathcal{C}$	confidence calculated for the payload decoding
$\mathfrak{P}(v^*)$	sub sets of the cropped data of all the actual payload
$\mathbb{P}(v^*)$	sub sets of the cropped data of mixture of payload and phase
$\mathcal{K}$	number of half of the bit occurrence
p	probability of transmission error
P	probability of some events
X_j	payload value in bit index j repeating positions
Z_j	phase value in bit index j repeating positions
Y_j	decoded payload value in bit index j repeating positions
$\mathbf{N}_{\tilde{\mathbf{P}}}$	number of experiments of correct detection of candidate payload
$\mathbf{N}_{\hat{\mathbf{P}}}$	number of experiments of correct detection of detected payload
$r_{\tilde{\mathbf{P}}}$	candidate payload decoding rate
$r_{\hat{\mathbf{P}}}$	payload decoding rate
$Cr(x)$	centroid in horizontal direction
$Cr(y)$	centroid in vertical direction
$s(i,j)$	sub-cell, where $0 \leq i,j \leq 3$
$h(k;i,j)$	unique sub-cell patterns, where $0 \leq i,j \leq 3$, $0 \leq k \leq 105$
$\hat{k}(s(i,j))$	sub-cell which has the smallest SSD value for all the sub-cells
$\mathcal{Q}$	number of bits to shift to standard payload form
$N_{n,j}$	number of positions for each row in the crop window $\mathbf{W}$
$\tilde{B}_{\text{row}}$	bit position repeat count of a row before rounding to the next larger integer
$\mathcal{B}_{\text{row}}$	bit position repeat count for a row
$\mathcal{B}_{\text{pay}}$	bit repeat count for payload rows
$\mathcal{B}_{\text{pha}}$	bit repeat count for phase rows
$\mathfrak{H}$	bit repeat count for the phase rows in a crop window of data array
$\mathfrak{H}_{\text{pay}}$	set of payload row indices in a crop window
$\mathfrak{H}_{\text{pha}}$	set of phase row indices in a crop window
R	total number of rows
M	number of phase rows

N	number of payload rows
$\mathcal{S}_1$	mathematical representation of the pure payload set without any error
$\mathcal{S}_2$	mathematical representation of the mixture of payload and phase set
$\tilde{\mathcal{S}}_1$	mathematical representation of the pure payload set with error
$\mathcal{S}_2$	mathematical representation of the mixture of payload and phase set with error
$\bar{Y}_1^{(j)}$	sample mean for the pure payload subset
$\bar{Y}_2^{(j)}$	sample mean for the mixture payload and phase subset
$\hat{Y}_k^{(j)}$	estimated bit value for each data set
$\mathbb{A}$	data set in the pure payload data set
$\mathbb{B}$	data set in the mixture of payload and phase
α	number of minority bits in data set $\mathbb{A}$
β	number of minority bits in data set $\mathbb{B}$.
ϵ	probability of bit error
$\mathcal{C}_1$	confidence value to select pure payload set
$\mathcal{C}_2$	confidence value to select mixture of payload and phase set
$\dot{\mathbf{U}}$	binary string that transferred from the decimal value C
$\hat{\dot{\mathbf{U}}}$	decoded binary string that transferred from the decimal value C
$\dot{M}$	actual bit repeat count for phase
$\hat{\dot{M}}$	decoded actual bit repeat count for phase

ABBREVIATIONS

AM	Amplitude Modulated
BPRC	Bit Position Repeat Count
BSC	Binary Symmetric Channel
CCWLS	Canonical Crop Window Location Set
FEC	Forward Error Correction
FM	Frequency Modulated
PQTS	Print Quality Troubleshooting Tool

ABSTRACT

Embedding information into a printed image is useful in many aspects, in which reliable channel encoding/decoding systems are crucial due to the information loss and error propagation during transmission. So how to improve the transmission accuracy and control the decoding error rate under a predictable level is always crucial to the channel design.

The current dissertation aims to discuss the design and performance of a two-dimensional coding method for printed materials – Circular Coding. It is a general two-dimensional coding method that allows data recovery with only a cropped portion of the code, and without the knowledge of the carrier image. While some traditional methods add redundancy bits to extend the length of the original massage length, this method embeds the message into image rows in a repeated and shifted manner with redundancy, then uses the majority votes of the redundant bits for recovery.

We introduce the encoding and decoding system and investigate the performance of the method for noisy and distorted images. For a given required decoding rate, we model the transmission error and compute the minimum requirement for the number of bit repeats. Also, we develop a closed form solution to find the the corresponding cropped-window size that will be used for the encoding and decoding system design.

Finally, we develop a closed-form formula to predict its decoding success rate in a noisy channel under various transmission noise levels, using probabilistic modeling. The theoretical result is validated with simulations. This result enables the optimal parameter selection in the encoder and decoder system design, and decoding rate prediction with different levels of transmission error.

We also briefly discuss two other projects: development of print quality troubleshooting tools and text line detection in scanned pages.

1. INTRODUCTION

Nowadays we use electronic media more than ever before, like the emails, ebooks, website, etc. However, according to researches, paper usage is still increasing these days rather than decreasing [1]–[3].

Printed documents serve as an interface between humans and the digital world [2]. One category of researches that has been done is to investigate the opportunities for abuse of trust through the generation of fallacious documents and illegal duplication of existing documents, including embedding of messages in these documents. For example, embedding information in printed documents can be used for a number of applications such as authentication of document content, proof of ownership, and identification of the printer that produced these documents. [4]–[9] On the other hand, some researches propose methods to improve the data embedding techniques, such as using intrinsic and extrinsic signatures to embed information inhalftoned images to be printed with a laser, electrophotographic printer [4], [10]–[15].

In this dissertation, we will focus on information embedding techniques for printed documents. One category of these data embedding techniques embeds data in a region that is solely dedicated to containing the message; but the visual appearance of the coded image is compromised. 1-D and 2-D barcodes [16], [17] and DataGlyphs [18], [19] are the predominant techniques in this class.

The other category of data embedding technologies carries the information in a manner that retains the original image in the content. Various methods for print information hiding have been proposed [2], [11], [20]–[28]. Among these methods, Bulan [22] used orientation modulation for data hiding in clustered-dot halftone prints. In this method, the message is represented by the different orientations of the dot clusters.

The circular coding with interleaving phase method was first proposed by Ulichney [20], [21] for channel encoding and decoding. It is a general two-dimensional coding method that allows recovery of data with only a cropped portion of the code, and without the knowledge of the carrier image. This method separates the message into two parts: the first part is the payload information which includes a sequence of all the values of the information bits; the second part is the phase code information which includes the starting point of the bit

sequence. It is enabled by circularly shifting the bit sequence according to embedded code in the phase line that is interleaved with the payload lines. This embedded the circular shift will be used to decode the payload. In other words, the circular coding method separates the message into two parts, and encodes both parts separately within a block of the image. Compared with some traditional methods that add redundancy bits to extend the length of the message, circular coding uses repeats of the message in the following rows, but in a circularly shifted fashion to add the redundancy. The decoding method then uses the majority votes of the redundant bits to recover the message.

Once the image is encoded and printed, and has been captured by some device, there will be different errors involved, such as the local distortion in printing, and the rotation and distortion in the image capturing device. These errors and the erosion of data require a robust channel coding method to ensure the decoding success.

From a communication systems point of view, the information can be hidden in halftone images and reliably transmitted after printing, and then extracted by a scanner [10], [11], [29]–[33].

As shown in Fig.1.1, an information embedding system usually contains the following components [34]: information source, source encoder, channel encoder, modulator, channel (storage media), demodulator, channel decoder, source decoder, and then destination. The source encoder transfers the information into a sequence of binary digits, or bits, called the information sequence. The channel encoder transfers the information sequence into a discrete encoded sequence called the code word. The code word then will be transferred in the coding channel (modulator + channel + demodulator) where noise is introduced. The output of the coding channel is called the received sequence. The channel decoder will transfer the received sequence back to the information sequence, called the estimated sequence.

The specific circular coding encoding and decoding communication system to be studied in this dissertation is shown in Fig. 1.2.

In Chapter 2, we introduce the error control coding system, and compare the similarity and differences between circular coding and other codes. We also introduce how the circular coding algorithm is applied to printed images for information embedding.

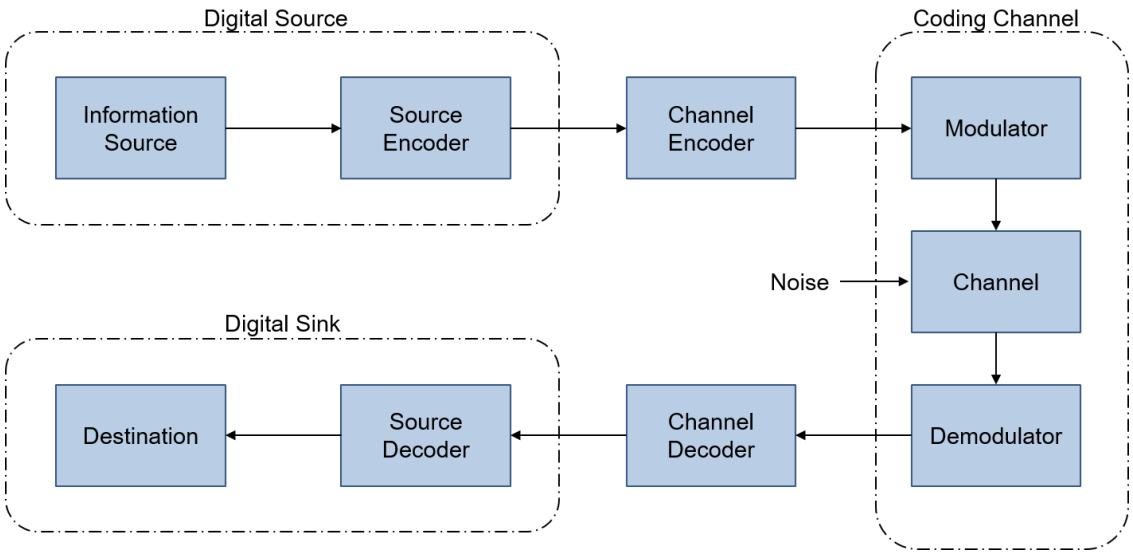


Figure 1.1. *Block diagram of a typical data transmission system.*

The design and development of the encoder and decoder of the circular coding algorithm is also discussed in Chapter 2. In addition, a simulation of the noise in the communication channel is discussed to analyze the decoding success rate.

In Chapter 3, we analyze the decoding performance in a noise free communication channel, including the minimum requirement of the crop window size of the data set, and develop a closed form solution of the bit repeat count for a given crop window of the data set.

In Chapter 4, we analyze the system performance by developing a closed-form expression to predict the decoding success rate in a noisy channel under various transmission noise levels, using probabilistic modeling. The theoretical result is validated with simulations. This result enables optimal parameter selection in the encoder and decoder system design, and decoding rate prediction with different levels of transmission error.

Some other projects are discussed in Chapters 5 and 6, including:

1. Print quality troubleshooting. We simulate the defects of the printers and provide online solutions for customer. Based on our work, print quality troubleshooting tools were released for six different products.

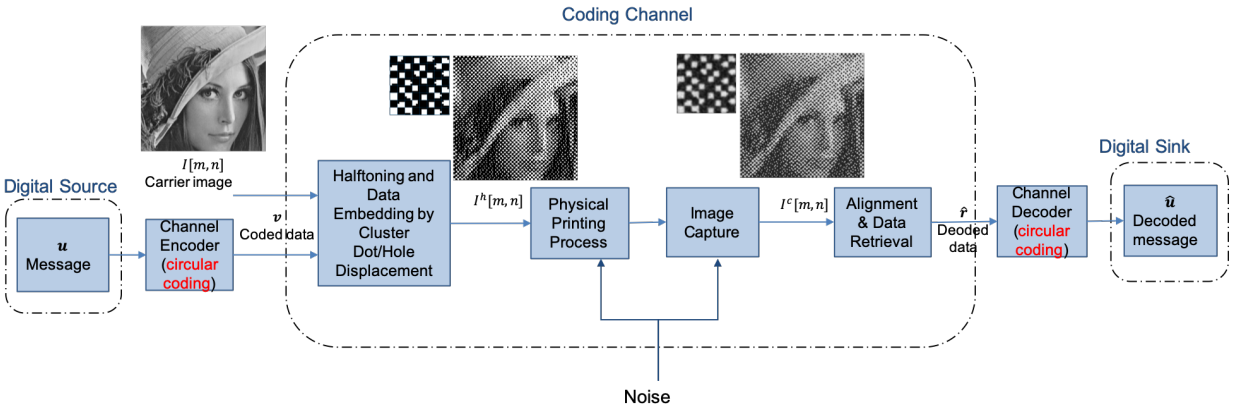


Figure 1.2. *Framework of the data transmission system. A message* $\mathbf{u}$ *is circularly coded and embedded in the carrier image* I*, and then transmitted in the coding channel, where noise may impact the result.*

2. Text line detection. For the document pre-processing purposes, we detect the text lines in a document by finding features of the symbols, and cluster the symbols into text lines.

2. DEVELOPMENT OF THE CIRCULAR CODING METHOD WITH INTERLEAVING PHASE PERIOD

2.1 Problem Statement

In 1948, Shannon [35] demonstrated that by proper encoding of the information, errors induced by a noisy channel can be reduced to any desired level without sacrificing the rate of information transmission. Since then, a lot of effort has been expended to find the efficient encoding and decoding methods for error control in a noisy channel [36].

The diagram shown in Fig. 1.1 represents a one-way system, which means that the transmission is in one direction only. Every message transmitted through the channel will be affected by some noise. Naturally,the aim of any communication channel is reliable delivery of information, that is to minimize the error in the transferred information. Error control for a one-way system must be accomplished using Forward Error Correction (FEC), so that the error correction will be automatically implemented by the receiver.

Generally speaking, FEC means the addition of redundancy to the transfer of information in a certain way, so that the errors after transmission can be detected and corrected. Simple examples of error correcting codes include:

1. Parity check. This is done simply by adding a single redundant bit as the sum modulo 2 of all data bits, and the repetition code (see below), which repeats every character multiple times. The simple parity check can detect, but not correct, single bit errors within a block.

2. Repetition code. This is done by repeating every data bit multiple times in order to ensure that it was sent correctly.

There are many other popular coding methods that have been developed to accomplish error correction. Here is a brief introduction to these codes:

- Hamming Code

  The Hamming Code [37] was invented by Richard Hamming in 1950. It is very efficient regarding the redundant bits used. The Hamming Code uses extra parity bits to allow

the identification and correction of errors. For example, Hamming's algorithm adds three additional check bits to every four data bits of the message. It can detect all single-bit and two-bit errors, and correct any single-bit error. To decode the message, each received word is assigned the nearest code word with respect to the Hamming distance, which corresponds to a minimization of the error probability.

- Linear Block Codes

  A linear block codes are a FEC code which encodes blocks of characters instead of single characters. Hamming code is a special case of linear block code.

- Reed–Solomon Codes

  The Reed-Solomon codes (RS codes) [38] were invented by Reed and Solomon in 1960. RS codes constitute a special case of linear block codes.

  In addition to the noise in the transmission, some information may be lost during the transmission. Some codes are designed to recover this information loss. They are in the category of erasure codes [39]–[41].

- Fountain Codes

  Fountain codes (also known as rateless erasure codes) [42]–[45] are a class of erasure codes. Potentially, a limitless sequence of encoding symbols can be generated from a given set of source symbols such that the original source symbols can ideally be recovered from any subset of the encoding symbols of size equal to or only slightly larger than the number of source symbols. The term fountain or rateless refers to the fact that these codes do not exhibit a fixed code rate.

- LT Codes

  Luby Transform (LT) codes [46], [47] are the first class of practical fountain codes that are near-optimal erasure correcting codes. They were invented by Luby in 1998 and published in 2002. The encoding process begins by dividing the uncoded message into many blocks of roughly equal length. Encoded packets are then produced with the help of a pseudorandom number generator [48].

- Raptor Codes

  Raptor codes **shokrollahi2007raptor**, [49], [50] are also in the class of fountain codes. They are formed by the concatenation of two codes. A fixed rate erasure code, usually with a fairly high rate, is applied as a 'pre-code' or 'outer code'. The inner code takes the result of the pre-coding operation and generates a sequence of encoding symbols. The inner code is a form of LT code. Each encoding symbol is the XOR of a pseudo-randomly chosen set of symbols from the pre-code output. The number of symbols that are XOR'ed together to form an output symbol is chosen pseudo-randomly for each output symbol according to a specific probability distribution. This distribution, as well as the mechanism for generating a pseudo-random numbers for sampling this distribution and for choosing the symbols to be XOR'ed, must be known to both the sender and receiver.

2.2 Overview of the Data Embedding Framework for Circular Coding

The data embedding framework for circular coding to be studied in this dissertation is shown in Fig. 1.2. The message is denoted as $\mathbf{u}$, and will be embedded into the continuous-tone image (carrying image) denoted as $I[m, n]$, which has first been halftoned. The data is embedded in the dot-cluster (highlight regions) or hole-clusters (shadow regions). The data embedded halftone image is denoted as $I^h[m, n]$. This image is then physically printed and captured by some device, such as a scanner, or a camera. The captured image is denoted as $I^c[m, n]$. The decoded message $\hat{\mathbf{u}}$ is then decoded from the captured image. To overcome the data erosion and errors in the print-capture channel, we employ a channel encoder with data redundancy.

The pipeline of the circular encoding/decoding framework can be summarized as the following procedures:

1. Encode the digital message $\mathbf{u}$ using the circular coding method to get the coded 2D data array $\mathbf{v}$;
2. Halftone the continuous-tone carrier image $I[m, n]$;

3. Shift dots within a selected subset of halftone cells corresponding to the metadata to be embedded into the image, denoted as $I^h[m, n]$;
4. Print the encoded image;
5. Capture the printed image, denoted as $I^c[m, n]$;
6. Decode the data array, denoted as $\hat{\mathbf{r}}$;
7. The recovered data array $\hat{\mathbf{r}}$ is then decoded to get back the message, denoted as $\hat{\mathbf{u}}$.

2.3 Channel Encoder

Digital image halftoning quantizes a gray-scale image to one bit per pixel. It may be classified as Amplitude Modulated (AM), Frequency Modulated (FM), or an AM-FM hybrid. Block-error diffusion [51] is one such method of producing FM halftones for printing and display. It has been used to generate hardcopy bar codes. Circular Coding [20] [21] is a general two-dimensional coding method that allows recovery of data with only a cropped portion of the code, and without the knowledge of the carrier image. It is used with AM halftoning. This is how it encodes the data: It repeats a payload with a fixed number of bits, while interleaving phase rows that embed the information of the circular shift for payload recovery, and shifts a fixed number of bits from row to row. The recovery system is given the number of bits of the payload, the interleaving phase period, and the row to row shift. It evaluates each candidate phase row and ranks its confidence based on the variance of the payload bits.

The goal of the encoding is to represent a payload using a 2-dimensional array of binary symbols. The data carrying unit is one halftone cell in the image that containing either a single dot cluster or a single hole cluster. It is usually a 4×4 array of pixels. The payload consists of B binary symbols. It is then repeated in the first row of the data array, until the end of the row. For each row below, every symbol is circularly shifted by a given bit value D. However, at every V-th row, the payload is replaced by a phase row. The phase row has the same length as the payload. But it is used to represent the following information: As we circularly shift the payload, and transfer each version of the payload to a decimal value,

there will be some versions that have the smallest decimal value. We define the circular shifted version that has the smallest decimal value as the standard version of the payload P, denoted as S. Then the payload P can be represented as the standard version S, and the circular shifting bits C. The circular shifting bits C are encoded in a phase line using some method. To avoid any confusion in the further decoding, we will select a payload that has a unique standard version S.

There are three steps for image encoding:

1. Create the data array for payload P with length B, row to row shift D, and interleaving phase period V.

2. Halftone the carrying image and embed the data array into it.

2.3.1 Create the data array for payload P

Given a payload with length B, row to row shift D, and interleaving phase period V, here are the steps for generating the data array:

1. Find the standard form S, which is one of the circularly shifted versions of the binary payload P that has the smallest decimal value for the binary string.

2. Find out the minimum number of bit shifts from the standard form S to the payload P. Denote it as C.

3. The maximum number of bits c that will be needed to represent the decimal value C is denoted as c, and is give by Eq. 2.1.

$$c = \lceil \log_2 B \rceil \tag{2.1}$$

   Encode the number of shifts C into another string of bits of length B (the same as the payload length), using some repeating strategy. One example of how to encode C is to double the bits and repeat this string until we fill in all the B bits. This new string that embeds the shift number of bits C is called the phase U.

4. Repeat the standard form of the payload S until the end of each row.

5. At each row below, circularly shift the payload S by a certain number of bits D.

6. Do the same row to row shift to the phase U.

7. Replace every V-th row of the circularly shifted standard form of the payload S with the accordingly circularly shifted phase rows.

8. Then we get the circularly shifted data array with interleaving period V.

The flowchart to create the data array is shown in Fig. 2.1.

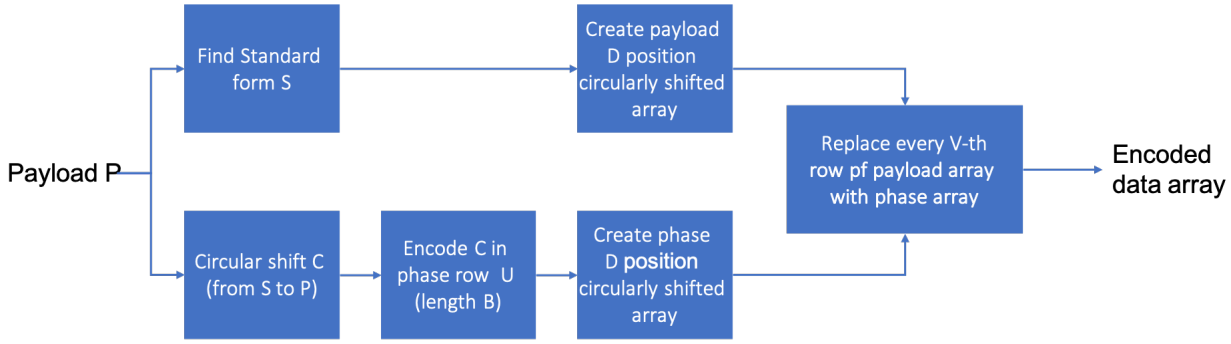


Figure 2.1. *Flow chart for creating the encoded data array.*

Note that the following parameters will be known to the encoder: B, V, D.

2.3.2 A toy example of creating the encoded data array

Here is an example to illustrate the encoding. The payload we would like to encode is $P = [1110000]$, So the payload length $B = 7$. The row to row shift is $D = 2$, interleaving period is $V = 3$.

(a) Find the standard form $S = [0000111]$.

(b) It will take 4 shifts to go from the standard form S to the payload P, or $C = 4$.

(c) Thus, $c = \lceil \log_2 7 \rceil = 3$. Double the bits and repeat the string to fill in the 7 bits. Then we have the phase code $U = [0011110]$.

(d) Repeat the payload in the successive rows.

(e) Replace each V-th row of payload with phase.

(f) Circularly shift each row by D bits from the previous row.

This toy example is illustrated in Fig. 2.2.

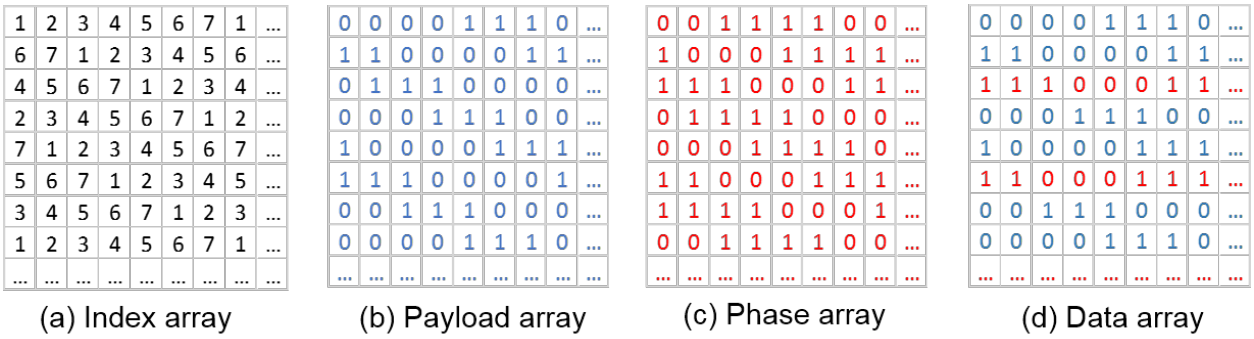


Figure 2.2. *Example of encoding the data array.* $P = [1110000]$, $B = 7$, $D = 2$, $V = 3$.

2.4 Channel Decoder

Generally speaking, the decoder knows the parameters that include the payload length B, the row-to-row shift D, and the interleaving phase period V. Also a cropped portion of the data array will be the input to the decoder. But the decoder does not know in which row the phase row first appears in the cropped data array. The decoder tries every possible case where that the first phase row could be, removes the assumed phase rows, and calculates the confidence that the remaining rows are pure payload rows. If the assumption of which rows are phase rows is correct (In other words, the assumption of payload rows is correct.), and if there is no error in the data array, then every bit will be repeated in its predefined position. So this will yield perfect consistency. Even if there are some bit errors, the consistency is high. On the other hand, if the assumption of phase rows is incorrect, then the remaining payload rows will contain both payload rows and phase rows. For each bit and its repeating positions, it contains the value of the payload and phase, which will have a lower consistency. The higher the consistency of the repeating bits, the higher probability that these are the

payload rows. So we can separate the payload and phase rows by selecting the one with highest consistency.

Then, the decoder takes the majority bit value of each repeating bit position of the payload rows, to find the shifted version of the payload, denoted as P. Similarly, by checking the majority bit value of each repeating bit position of the phase rows, we can find a shifted version of the phase, denoted as U.

For every payload, since the standard version is unique, there is a unique circular shift C that shifts from the standard version S to the original payload P . We can find the standard version from P, and figure out the circular shift C that will take us from P to $\hat{S}$. It will be the same shift that shifts the phase U to $\hat{U}$. The circular shift $\hat{C}$ that takes us from the original payload to the standard form can be decoded from the phase $\hat{U}$, and then can be used to predict the decoded payload $\hat{P}$, referring to Fig. 2.3.

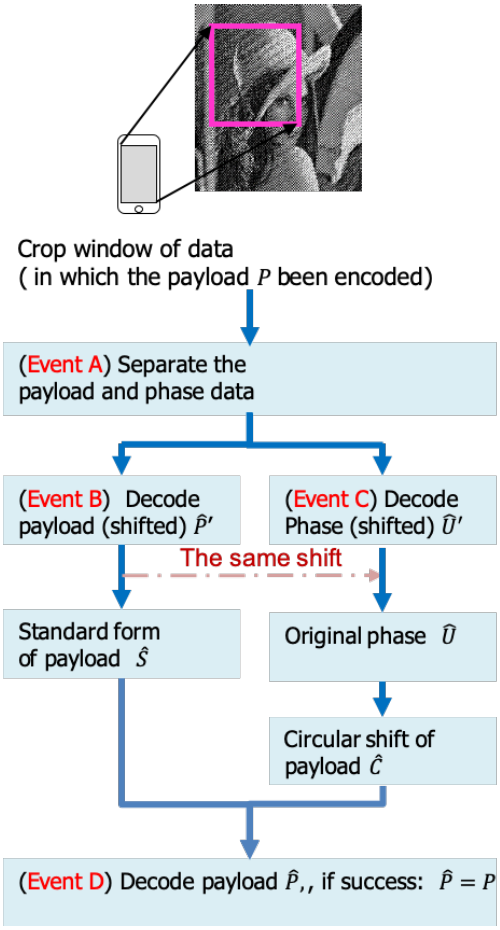


Figure 2.3. *Overview of the circular coding decoding process. The transmission error will impact the process of these four events: Event A: separating the payload and phase data; Event B: decoding the shifted sequence of the payload; Event C: decoding the shifted sequence of the phase; Event D: decoding the payload. All the other processes are deterministic.*

2.4.1 The majority and minority bits

Given a crop window of the the data array, for bit index j where the bit indices are assigned starting with row 1 column 1 within the crop window, as shown in Fig. 2.2 (a), we denote the total number of bit occurrences within this crop window as $\mathcal{B}(j)$. Note, as also shown in Fig. 2.2 (a), that each succeeding row of the bit index array is shifted by D positions relative the the previous row. Thus, each position in the bit index corresponds uniquely to one of the B bits in the payload. We call $\mathcal{B}(j)$ the bit repeat count for bit

position j. There is a number of $\mathcal{B}^{+}(j)$ bit occurrences that have the majority bit value, and a number $\mathcal{B}^{-}(j)$ of bit occurrences that have the minority bit value.

2.4.2 Decoding method

For each value of the bit position index j in the payload, we use the value of the majority occurrence of this bit index as the decoding value. This estimate has an uncertainty associated with it, that is the fraction of the minority value bits occurrence divided by the total bit position occurrence:

$$\mu(j) = \frac{\mathcal{B}^{-}(j)}{\mathcal{B}(j)} \tag{2.2}$$

Once we have estimated the bit values of the entire payload with length B, we will have an overall confidence value for the payload estimation, which is calculated as:

$$\mathcal{C} = 1 - \frac{2}{B} \sum_{j=0}^{B-1} \mu(j) \tag{2.3}$$

2.4.3 An example of decoding

The same example in Fig 2.2 that was used to illustrate encoding is continued here in Fig. 2.4 to illustrate how the decoding is works.

The first step is to restore the data array. By shifting each row back D bits circularly relative to the payload length B, we will get the restored data array, as shown in Fig. 2.4, the crop window width is 5, but the payload length B is 6. We can first fill in the one bit that is missing, and then circularly shift back D bits to restore the original aligned bit positions. Note that here the bit index is labeled as $1, 2, ..., 6$ to indicate this is the bit position relative to the crop window. So when we decode this payload, the payload bit index is a circularly shifted version that is related to the position of the crop window in the original data array.

Since the example in Fig. 2.4 does not have enough repeat bits for every bit index, we will use another example that has a larger crop window of 12×12 pixels. See the example in Fig. 2.5. Here, abstentions are halftone cells in which the dot-cluster or hole-cluster is not shifted. So abstentions carry no information.

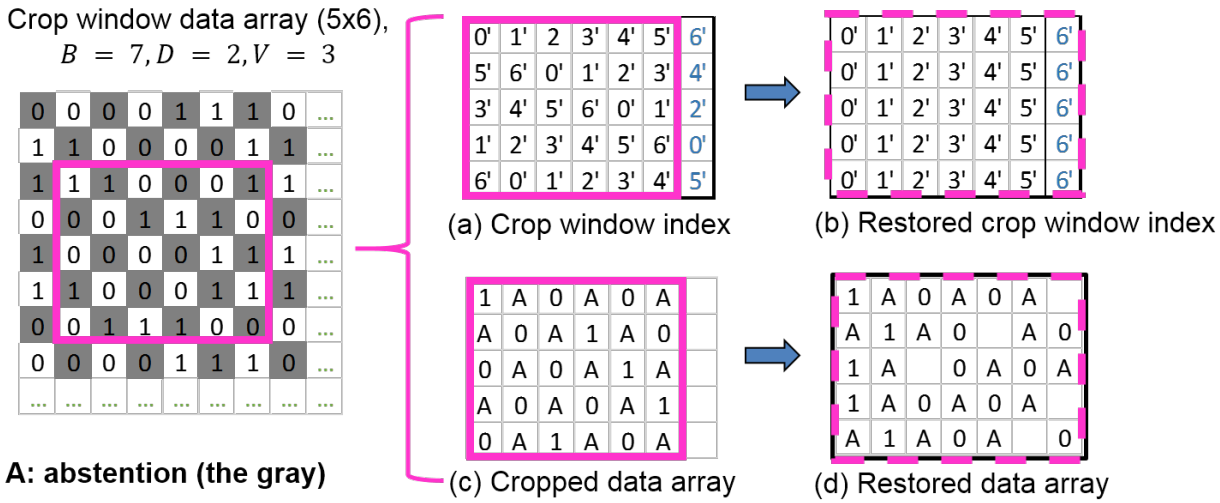


Figure 2.4. *Example of a crop window of data array with size of* 5×6*, with bit length* $B = 7$*, row to row shift* $D = 2$*, and interleaving phase period* $V = 3$*. The restored crop window is expanded to* 5×7 *to accommodate the circular shift of the payload.*

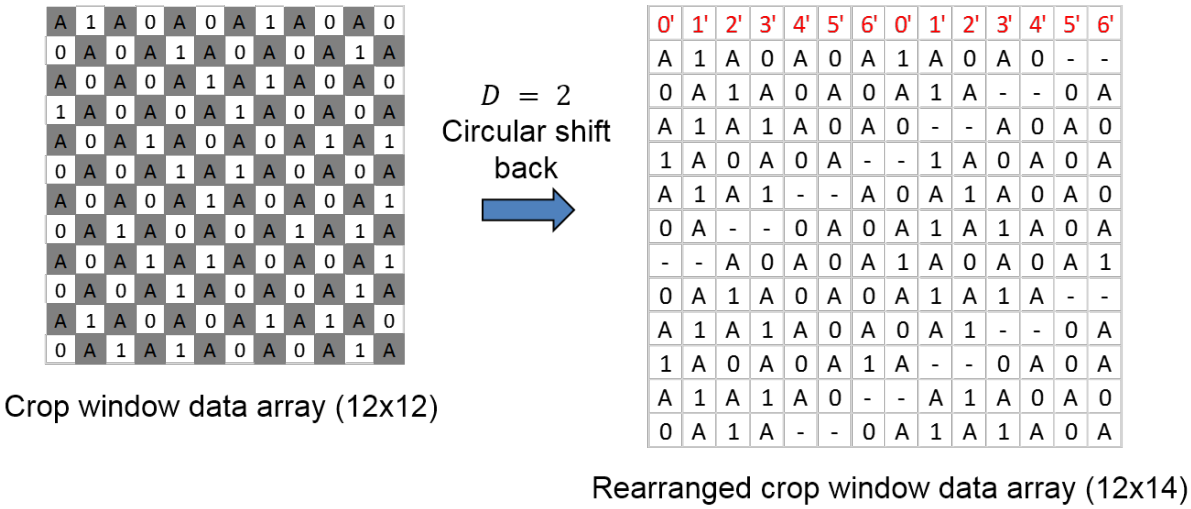


Figure 2.5. *Example of a crop window of data array with size of* 12×12*, with bit length* $B = 7$*, row to row shift* $D = 2$*, and interleaving phase period* $V = 3$*. The restored crop window is expanded to* 12×14 *to accommodate the circular shift of the payload.*

Second, we divide the crop data array into payload and phase rows, and find the confidence for each separation. Let v denote the row index of first phase row in the crop window, then there are totally V possible separations. For the same 12×12 example shown in Figs. 2.4 and 2.5, we illustrate this step in Fig. 2.6. Note that the confidence is calculated as $\mathcal{C} = 1 - \frac{2}{B}\sum_{j=0}^{B-1}\mu(j)$, see Eq. 2.3.

Third, from the candidate payload $\hat{P}$, we can find the standard form of the payload $\hat{S}$ by circularly shifting $\hat{P}$ and selecting the one with the smallest decimal value. Note that during the design of the payload, we will make sure that the standard form of the payload is unique. And the original phase row will be aligned with the standard version of the payload row without any row to row shift. So once we find the standard form of the payload, we also can detect the phase $\hat{U}$.

Fourth, from the phase $\hat{U}$, we will be able to decode the circular shift $\hat{U}$ using the same method that was used to decode the payload, but taking into account the extra replication of bits in each phase row. In this example shown in Fig. 2.6, the circular shift decoded from the phase is $\hat{C} = 4$.

Now that we have the standard form $\hat{S}$ and the circular shift $\hat{C}$, we will be able to reconstruct the payload $\hat{P}$ by circularly shifting $\hat{C}$ bits from the standard form $\hat{S}$. See Fig. 2.7.

2.5 Coding Channel: Embedding Data in Halftone Images

There are various methods [6], [7], [10], [30], [33], [52] that can be used to embed data into halftone images. Halftone images are typically binary images. Each pixel of the halftone image is either on or off, indicating whether ink/toner is deposited on this pixel or not, respectively.

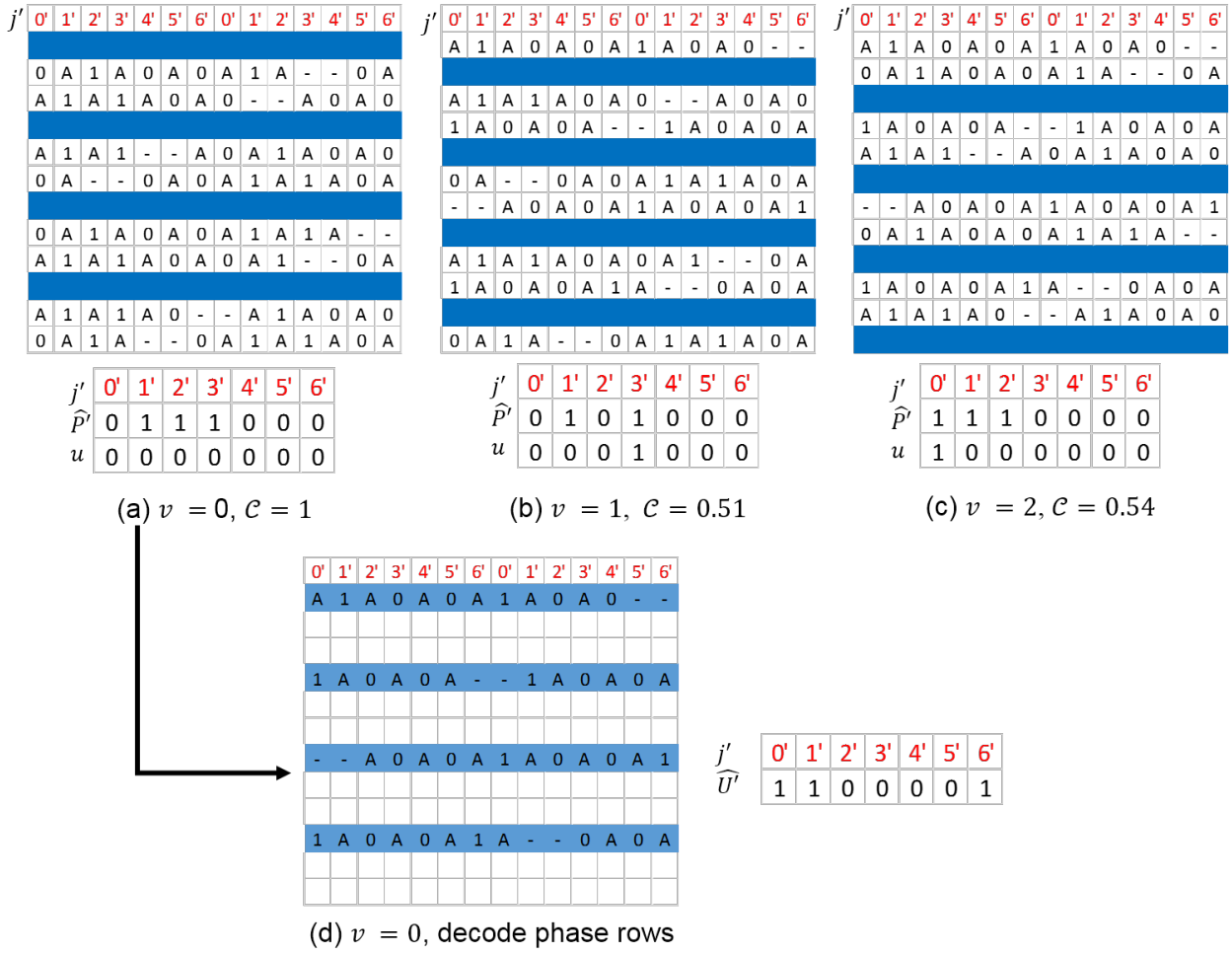


Figure 2.6. *Example of decoding the payload. (a) The restored (rows circularly shifted back) crop window of data array with size of* 12×12*, with bit length* $B = 7$*, row to row shift* $D = 2$*, and interleaving phase period* $V = 3$*. The restored crop window is expanded to* 12×14 *to accommodate the circular shift of the payload. The blue highlighted rows are assumed to be the phase rows and moved for to decode the candidate payload* $\hat{P}$*. Every* V*-th row starting with row* $v = 0$ *in sub figure (a) will be selected and used to decode the candidate phase* $\hat{U}$*.*

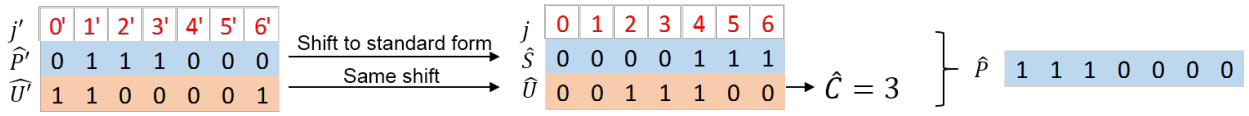


Figure 2.7. *Example of finding the payload from candidate payload and the phase. payload length* $B = 7$*.*

The gray scale image is block thresholded with a screening array $T(k,l)$:

$$T(k,l) = \frac{1}{64} \cdot \begin{bmatrix} 50 & 52 & 48 & 44 & 15 & 13 & 17 & 21 \\ 54 & 64 & 62 & 46 & 11 & 1 & 3 & 19 \\ 56 & 58 & 60 & 42 & 9 & 7 & 5 & 23 \\ 36 & 38 & 40 & 34 & 31 & 27 & 25 & 29 \\ 16 & 14 & 18 & 22 & 49 & 51 & 47 & 43 \\ 12 & 2 & 4 & 20 & 53 & 63 & 61 & 45 \\ 10 & 8 & 6 & 24 & 55 & 57 & 59 & 41 \\ 32 & 28 & 26 & 30 & 35 & 37 & 39 & 33 \end{bmatrix} \tag{2.4}$$

Let $I(m,n)$ denote the grayscale image, which is assumed to be scaled to values between 0 and 1. Then, the halftone image $I^h(m,n)$ is obtained as in Eq. 2.5.

$$I^h(m,n) = \begin{cases} 1, & \text{if } I(m,n) \geq T(\text{Mod}(m,8), \text{Mod}(n,8)) \\ 0, & \text{if } I(m,n) < T(\text{Mod}(m,8), \text{Mod}(n,8)) \end{cases} \tag{2.5}$$

The halftone cell has size 8×8 pixels, and each halftone cell contains four sub-cells, each has size 4×4 pixels. See Fig. 2.8

For a constant gray scale image between gray levels 0 and 1, there are only 33 different halftone patterns, as shown in Fig. 2.9.

In order to make the halftone patterns be limited to these patterns, we will first average the gray scale value with each 4×4 sub-cell.

2.5.1 Abstention and carrying sub-cells

For each sub-cell, if it is all black or all white, we call it an "abstention" sub-cell. Otherwise, if it is white holes surrounded by black, or black surrounded by white, we call these potential "carrying" sub-cells. An example of the sub-cells is shown in Fig. 2.10.

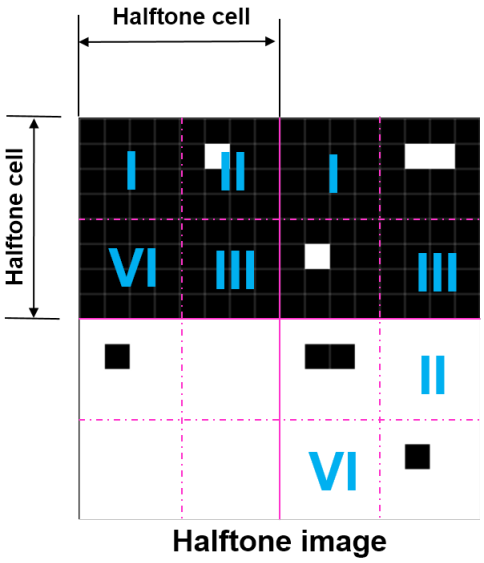


Figure 2.8. *Illustration of the four quadrants of an* 8×8 *halftone cell.*

2.5.2 Highlight and shadow region

For each potential carrying sub-cell, we examine its four neighboring sub-cells: upper, lower, left and right. If this potential carrying sub-cell's neighboring sub-cells are all solid black abstention sub-cells, then we define this potential carrying cell to be in the shadow region. On the other hand, if this potential carrying sub-cell is surrounded by four solid white abstention sub-cells, then we define this carrying cell to be in the highlight region.

There are cases where the upper, lower, left and right neighboring sub-cells are not all solid abstentions. We treat these cases as follows: (1) if there are more white abstentions than black abstentions, then we claim this potential carrying sub-cell is in the highlight region; (2) if there are more black abstentions than white abstentions, then we claim this potential carrying sub-cell is in the shadow region; (3) otherwise, if there are the same number of white and black abstentions in the four neighboring sub-cells, then this potential carrying sub-cell is neither in the highlight nor the shadow region. We will not use this sub-cell as a carrier. Examples of shadow regions and highlight regions are shown in Fig. 2.11.

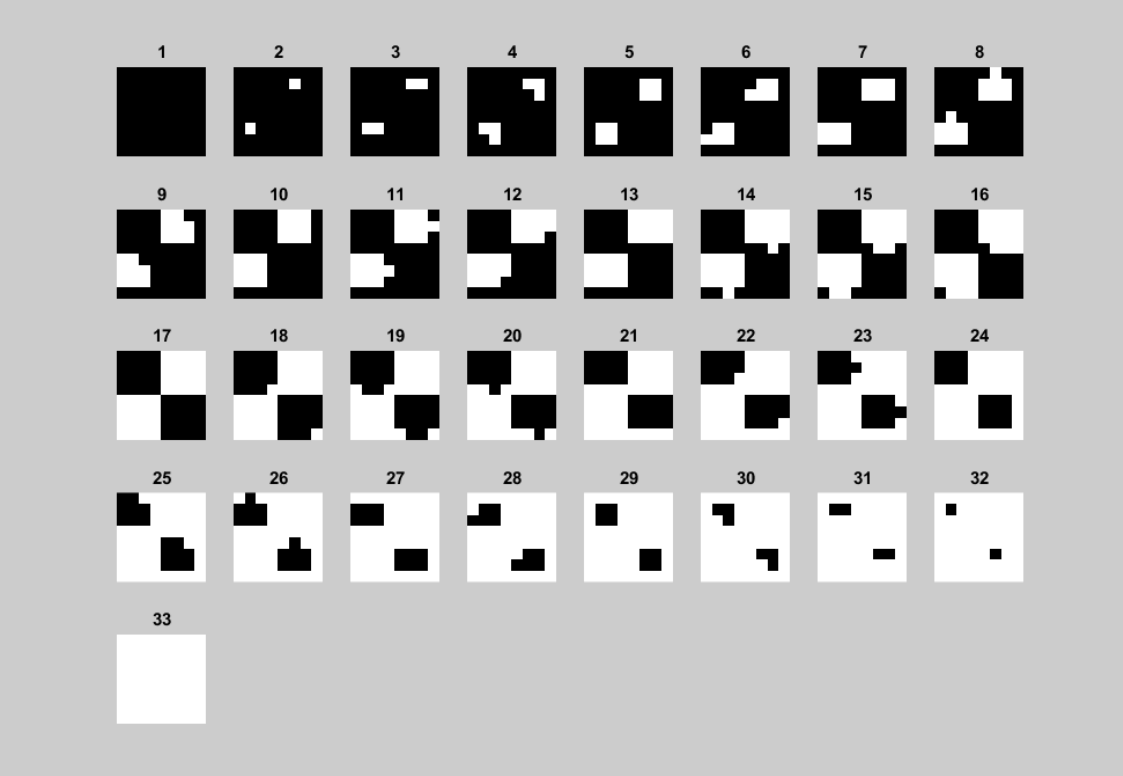


Figure 2.9. *Halftone cells with different gray scale values.*

2.5.3 Embed the data array into a halftone image

The data array symbols are hidden into each of the halftone sub-cells, row by row, and sub-cell by sub-cell. If the sub-cell is an abstention, then we cannot embed any symbol in this sub-cell, but this sub-cell still takes one position in the data array.

For the potential carrier sub-cell, first we need to examine its neighboring sub-cells (see Fig. 2.11) to determine which region (highlight or shadow) it belongs to, and replace this sub-cell with the balance shifted sub-cells to represent zero or one in the data array as shown in Fig. 2.12. Note we have two directions of shift for each symbol value, and alternatively, we select the directions of the shift based on the data array.

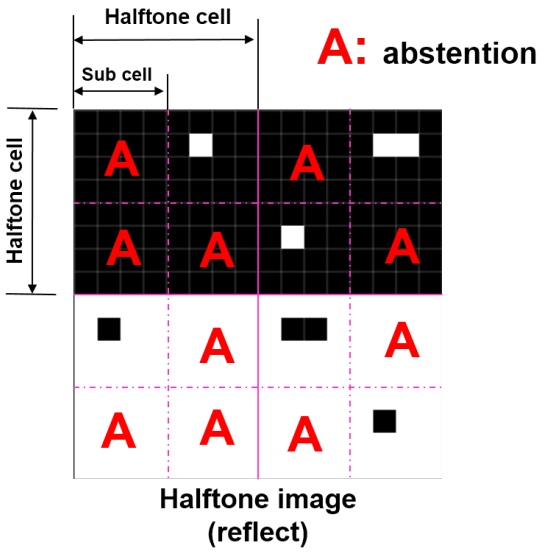


Figure 2.10. *Example of halftone cell and sub-cell. Each halftone cell has size 8 × 8 pixels, and each sub-cell has size 4 × 4 pixels.*

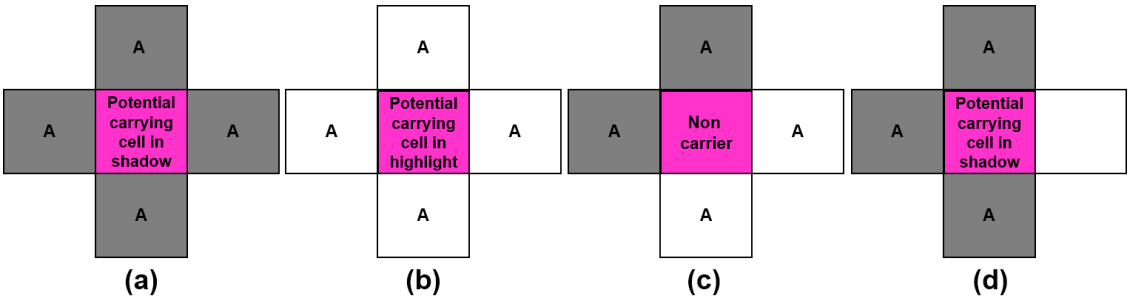


Figure 2.11. *Examples of (a) carrying sub-cell in shadow region, (b) carrying sub-cell in highlight region, (c) non-carrier sub-cell, and (d) carrying sub-cell in shadow region where the majority neighboring sub-cells are shadow.*

2.5.4 Balanced shifting rule

The symbol can be embedded into the halftone cluster by changing the orientation of the cluster [22], [53]. Or it can be embedded into the dot cluster halftone image by shifting the dot cluster within the sub-cell [54]. For example, we can let the un-shifted halftone sub-cell represent a zero, and shift the dot cluster right and down one pixel within the halftone sub-cell to represent a one.

In order to have a homogeneous shift of the dot clusters for the whole encoded halftone image, we use the balanced shift rule. For each gray level, we push the dot cluster either to the north-west or to the south-east direction to represent a zero, and push the dot cluster either to the north-east or to the south-west to represent a one. We alternately select the direction each time we need to encode that bit value. The balanced shifting patterns for shadow and highlight sub-cells are shown in Fig. 2.12.

2.5.5 Example of the image with embedded data array

In Figs. 2.14 and 2.15, we see that the gray-scale image is halftoned first, then the cluster of the black dots or white holes is shifted within each halftone sub-cell to embed the data array.

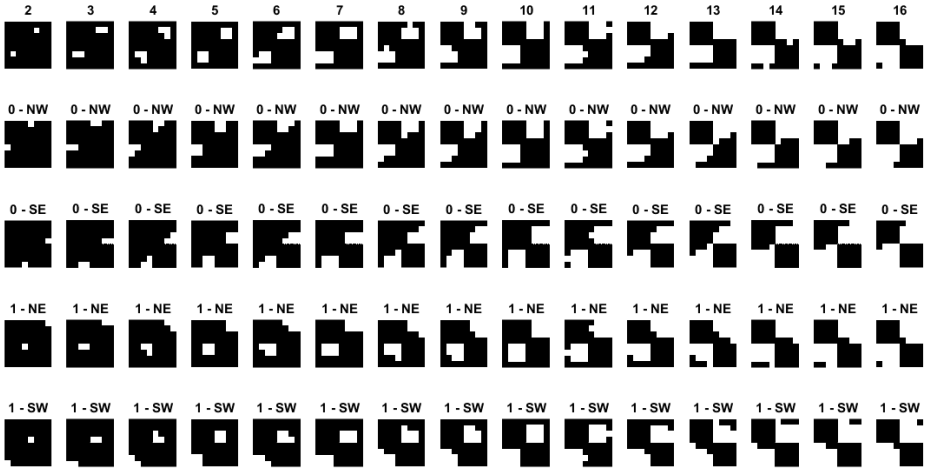
2.6 Coding Channel: Data Retrieval From Captured Image

When the data array embedded halftone image is printed and captured by some device, we might only have a cropped version of the image from which to retrieve the data. In order to decode the payload, the first step is to decode the data array from the halftone image.

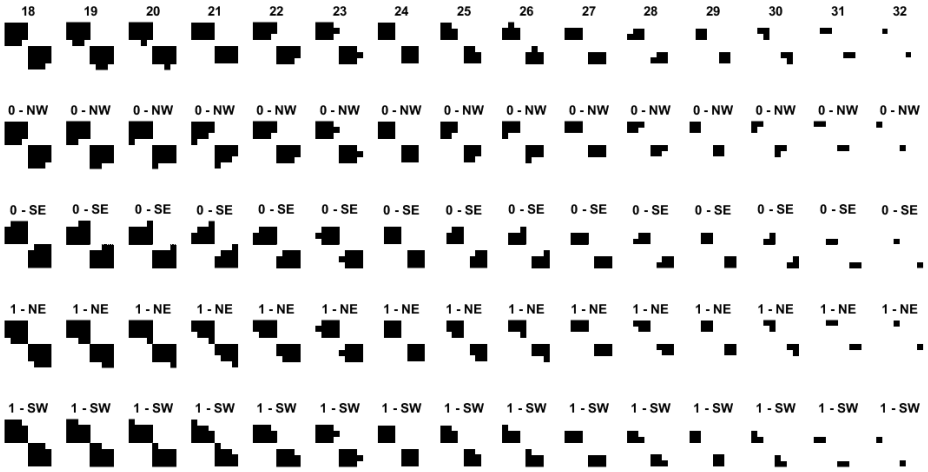
Assume that the captured image is aligned with each halftone cell, and we can retrieve each sub-cell. We tried two methods to recover the data array symbols that have been embedded into this image. One method is to compute the Euclidean distance from each sub-cell to each of the possible halftone patterns (in Fig. 2.12), and to find the best matching pattern, which has the smallest Euclidean distance to the target pattern. Another method is to find the centroid of the black dot clusters in the highlight region or the white hole clusters in the shadow region for each sub-cell, and to determine the symbol value from the centroid location. These two methods are described in detail below.

1. Minimum Euclidean distance pattern matching method to decode the halftone image

   The sub-cell patterns for highlight and shadow regions are shown in Fig. 2.13. Note that the balance shifted version of pattern number 2 is the same as the balance shifted versions of pattern number 18. So are patterns 3 vs 19, patterns 15 vs 31, and patterns

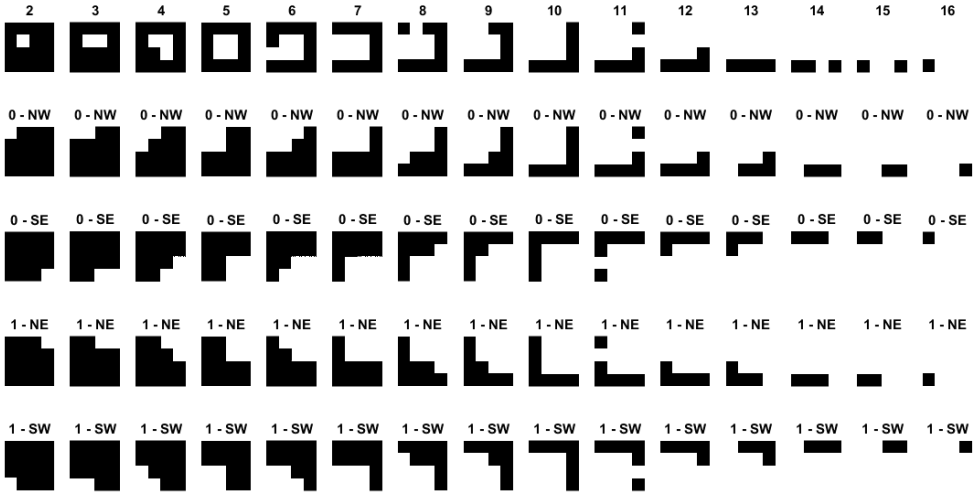


(a)

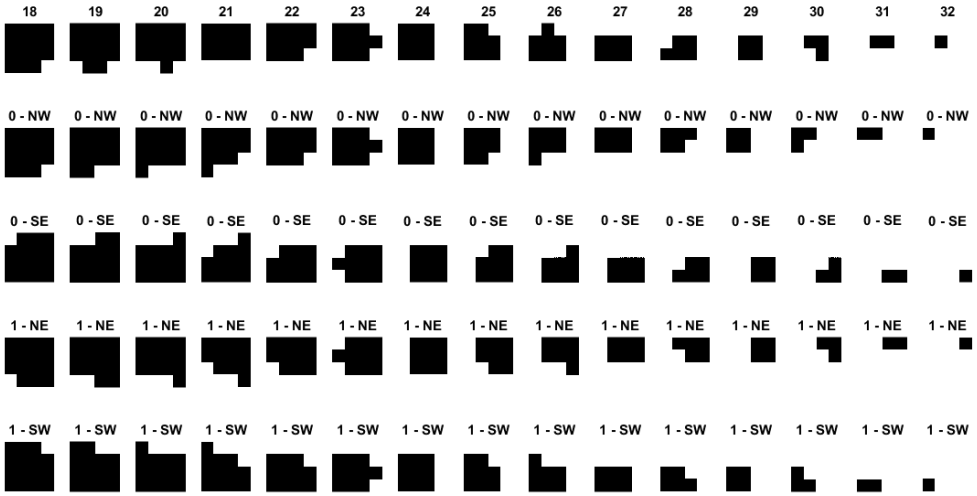


(b)

Figure 2.12. *Balanced shifting pattern for (a) shadow region and (b)highlight region. There are totally 33 different gray levels, indexed from 1 to 33. The first rows in (a) and (b) are the unsifted versions, where indices from 2 to 16 are the dot clusters that can be used to carry information in the shadow regions; indices from 18 to 32 are the dot clusters that can be used to carry information in the highlight regions. The second and third rows illustrate the shifts to the northwest (0-NW) and southeast (0-SE) directions, respectively, to represent the bit value of 0. The third and forth rows illustrate the shift to the northeast (1-NE) and southwest (1-SW) directions, respectively, to represent the bit value of 1. Please note that the dot cluster of index 1 is the whole shadow region; for index 17, it is the perfect checkerboard pattern; and for index 33, it is the highlight region. None of these three cluster dots can be used to carry information, so they are not shown here.*



(a)



(b)

Figure 2.13. *Balanced shifted sub-cell patterns for (a) shadow region and (b) highlight region.*

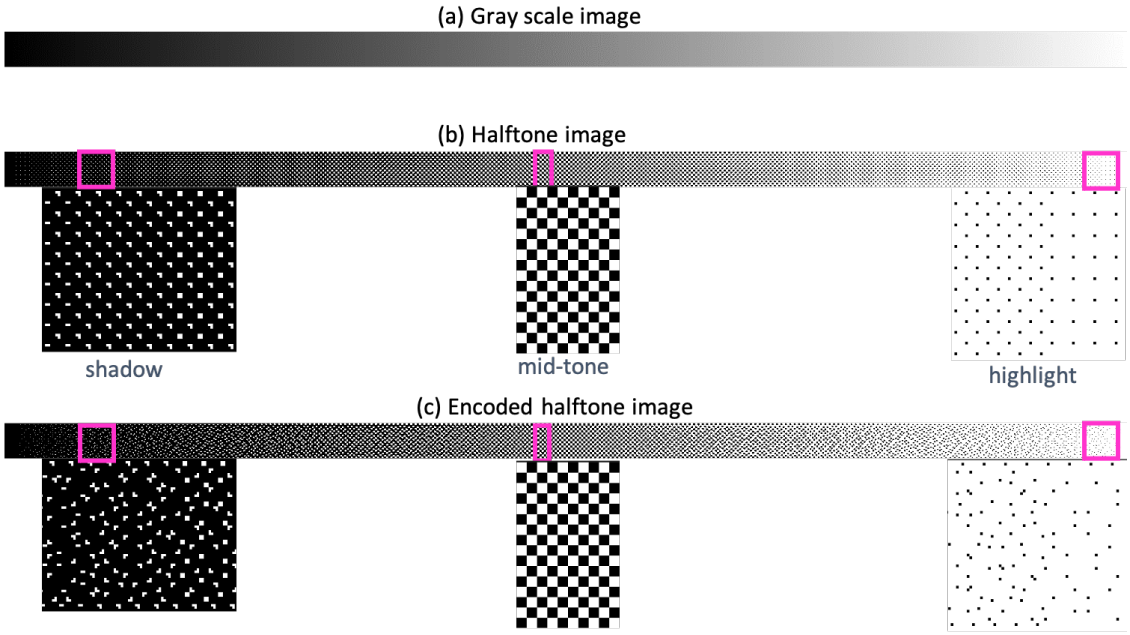


Figure 2.14. *Example illustrating the embedding of a data array into a gray ramp image.* $\mathbf{P} = [10110100001110010], \mathbf{U} = [00001111000000100], B = 17, D = 2, V = 3.$

16 vs 32. But the symbol values they represent are the same. So there are totally 104 different sub-cell patterns for carrying sub-cells, in addition to the two abstention sub-cells.

Let $s(i, j)$ where $0 \leq i, j \leq 3$ denote the sub-cell, and $h(k; i, j)$ where $0 \leq i, j \leq 3$, $0 \leq k \leq 105$ (including the two abstention sub cell patterns) denote the unique sub-cell patterns. Thus, we can compute the Sum of Squared Distance (SSD) from the sub-cell $s(m, n)$ to each sub-cell pattern:

$$SSD(k) = \sum_{i=0}^{i=3} \sum_{j=0}^{j=3} (s(i, j) - h(k; i, j))^2 \tag{2.6}$$

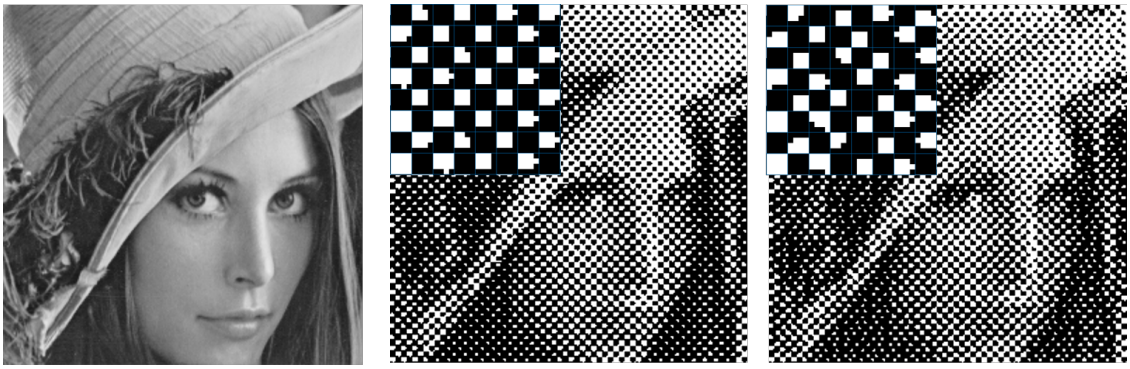


Figure 2.15. *Example illustrating the embedding of a data array into a halftone image.* $\mathbf{P} = [1111111111111111], \mathbf{U} = [0000000000000000], B = 17, D = 2, V = 3$. *(a) gray scale image, (b) halftoned image, (c) halftone image with embedded data array.*

Then the pattern index $\hat{k}(s(i,j))$ for $s(i,j)$ is the one with smallest SSD value.

$$\begin{aligned} \hat{k}(s(i,j)) &= \arg\min_k SSD(k) \\ &= \arg\min_k \sum_{i=0}^{3}\sum_{j=0}^{3}(s(i,j) - h(k;i,j))^2 \end{aligned} \tag{2.7}$$

Once we have determined the sub-cell pattern, we can figure out the symbol value based on Fig. 2.13.

2. Centriod method to decode the halftone image

   The centroid calculations for shadow sub-cell patterns and highlight sub-cell patterns are performed differently. For shadow patterns, we calculate the centroid of the white cluster holes; and for highlight patterns, we calculate the centroid of the black cluster dots.

   From the plot in Fig. 2.16, we can see that symbols 0 and 1 fall into four different quadrants. So we can classify the symbol value for each sub-cell based on the centroid calculation.

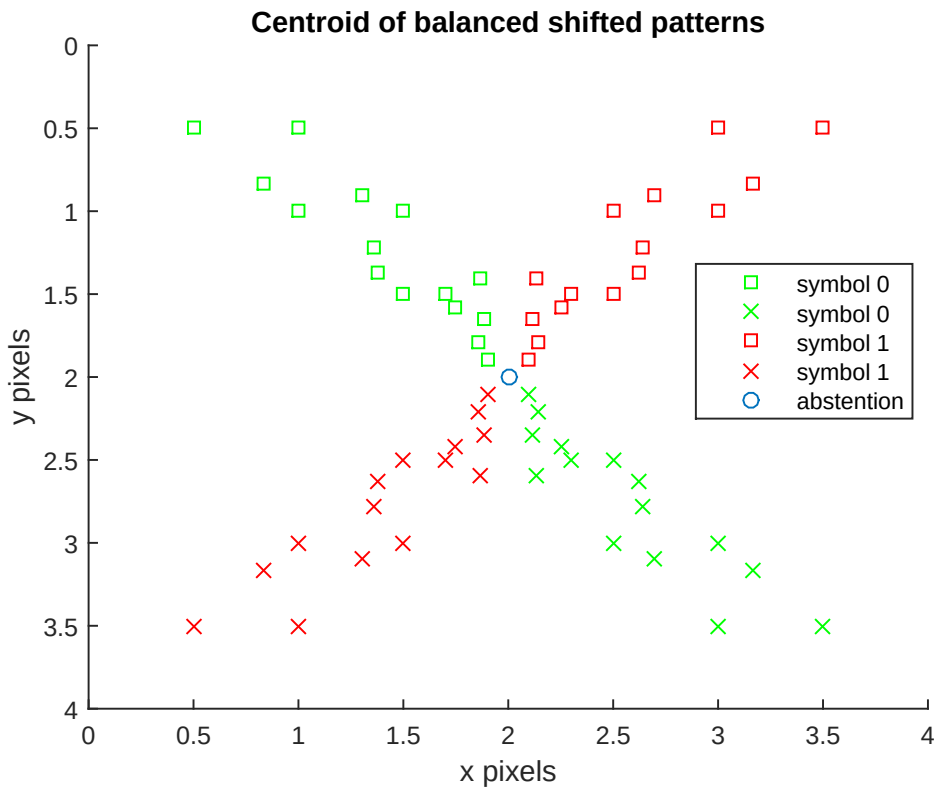


Figure 2.16. *Plot of the centroids for balanced shifted sub-cell patterns and abstentions.*

The centroid of a sub-cell is calculated based on the spatial distribution of the cluster (black dots in the highlight region and white holes in the shadow region). The horizontal and vertical centroids of the sub-cell are given in Eq. 2.8.

$$
\begin{aligned}
C_x &= \frac{\sum_{i=0}^{3}\sum_{j=0}^{3}(i-0.5)g[i,j]\cdot b[i,j]}{\sum_{i=0}^{3}\sum_{j=0}^{3}g[i,j]\cdot b[i,j]} \\
C_y &= \frac{\sum_{i=0}^{3}\sum_{j=0}^{3}(j-0.5)g(i,j)\cdot b[i,j]}{\sum_{i=0}^{3}\sum_{j=0}^{3}g[i,j]\cdot b[i,j]}
\end{aligned}
\tag{2.8}
$$

where $b[i,j]$ is a binary mask generated using the Otsu's method [55]. The array $g[i,j]$ is the reflective value of the white holes in the shadow region, or the opposite of the dark dots in the highlight region at the pixel with coordinates $[i,j]$. The 0.5 pixel offset in Eq. 2.8 shifts the effective coordinate location of each pixel to its center.

The flowchart of data retrieval using the centroid decoding method is shown in Fig. 2.17.

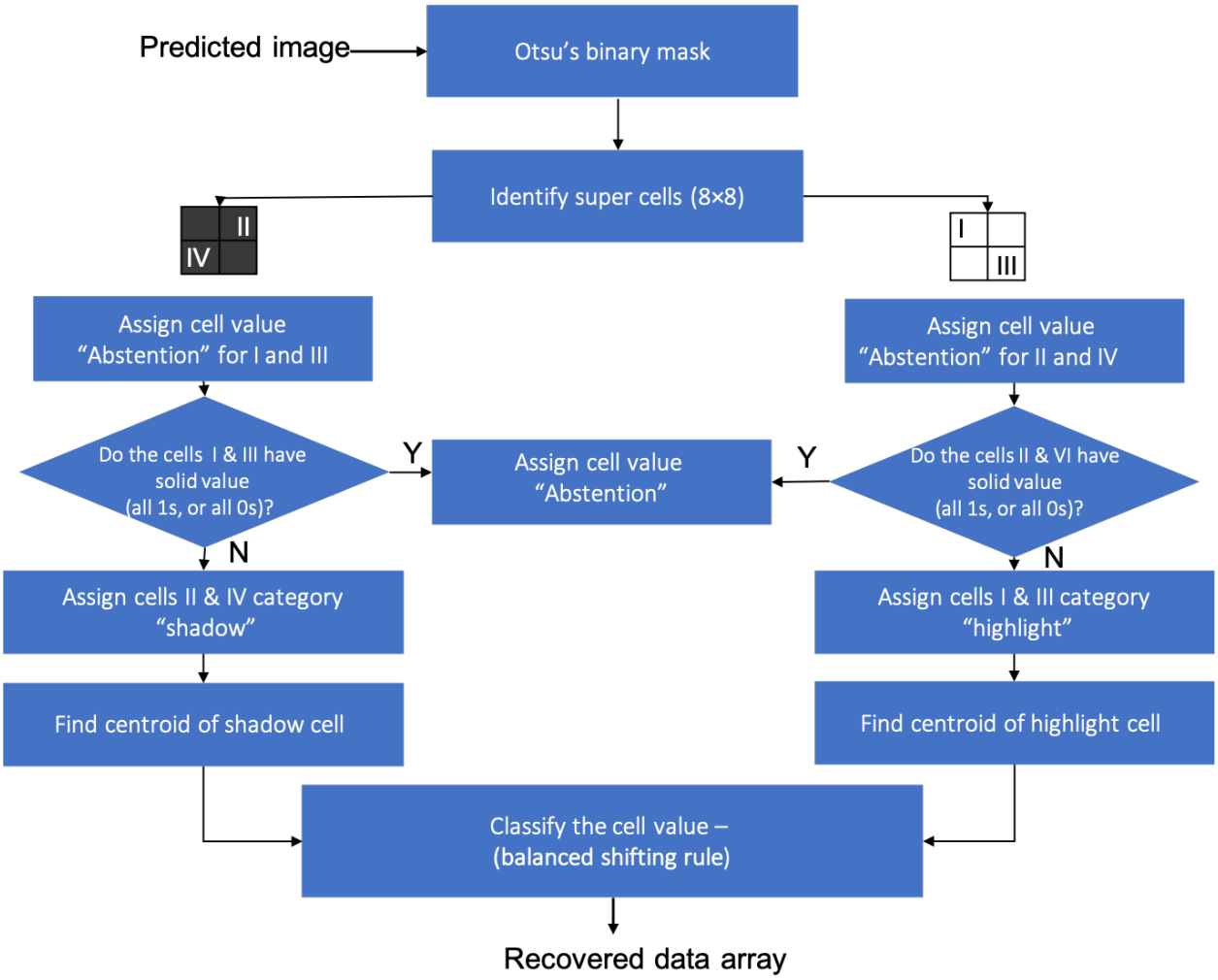


Figure 2.17. *Flowchart of data retrieval from the captured image using the centroid method when the data was embedded according to the balanced shifting rule. See Fig. 2.8 for the illustration of the four sub-cells I, II, III and IV.*

Some further research has been done more recently on the decoding of halftoned images [56], [57].

2.7 Calculate the Decoding Rate

We calculate the candidate payload decoding rate and the payload decoding rate separately. Let $\mathbf{N}$ denote the total number of experiments for each level of transmission error rate and crop window size; let $\mathbf{N}_{\tilde{\mathbf{P}}}$ denote the number of experiments for which every bit in the candidate payload is correct; and let $\mathbf{N}_{\hat{\mathbf{P}}}$ denote the number of experiments for which every bit in the decoded payload is correct. So the candidate payload decoding rate $r_{\tilde{\mathbf{P}}}$ and the payload decoding rate $r_{\hat{\mathbf{P}}}$ can be calculated in Eq. 2.9, respectively.

$$\begin{aligned} r_{\tilde{\mathbf{P}}} &= \frac{\mathbf{N}_{\tilde{\mathbf{P}}}}{\mathbf{N}} \\ r_{\hat{\mathbf{P}}} &= \frac{\mathbf{N}_{\hat{\mathbf{P}}}}{\mathbf{N}} \end{aligned} \tag{2.9}$$

Tables 2.1 and 2.2 show the results of the simulation. Based on the data contained in Table 2.2, we may conclude that the circular coding method is quite effective.

Table 2.1. The candidate payload decoding rate with different levels of transmission error rate, and different sizes of the crop window. Here, the crop window is of size $W \times H$. The average Bit Position Repeating Count (BPRC), to be discussed in Sec. 3.3, is also listed here. Payload bit length $B = 67$, interleaving phase period $V = 4$, row to row shift $D = 4$. Each rate is calculated based on all the possible crop windows of data within the Canonical Crop Window Location Set (CCWLS), to be discussed in Sec. 3.2, and 3 different random samples of transmission error for each crop window of data.

W,H	20	30	40	50	60
BPRC	2	12	22	34	48
Transmission error	Candidate payload decoding rate $r_{\tilde{\mathbf{P}}}$				
10%	61.5%	100%	100%	100%	100%
9%	69.6%	100%	100%	100%	100%
8%	74.4%	100%	100%	100%	100%
7%	78.0%	100%	100%	100%	100%
6%	82.0%	100%	100%	100%	100%
5%	91.3%	100%	100%	100%	100%
4%	91.3%	100%	100%	100%	100%
3%	91.3%	100%	100%	100%	100%
2%	100%	100%	100%	100%	100%
1%	100%	100%	100%	100%	100%
0%	100%	100%	100%	100%	100%

2.8 Conclusion

In this chapter, we reviewed the data embedding framework for Circular Coding, introduced the encoding and decoding method, and illustrated these methods with examples. In addition, we introduced the methods of embedding data into the image, and how to retrieval these information using different approaches. Finally, we designed the framework, simulated the whole process, and predicted the decoding rate with different parameters. From the simulated decoding result, we may select the proper parameter settings with a desired decoding rate.

Table 2.2. The decoded payload decoding rate with different levels of transmission error rate, and different sizes of crop window. The average Bit Position Repeating Count (BPRC),to be discussed in Sec. 3.3, is also listed here. Payload bit length $B = 67$, interleaving phase period $V = 4$, row to row shift $D = 4$. Each rate is calculated based on all the possible crop windows of data within the Canonical Crop Window Location Set(CCWLS), to be discussed in Sec. 3.2, and 3 different random samples of transmission error for each crop window of data. Note, the result of "N/A" means that due to the insufficient number of bit repeats for phase, the final decoding failed.

W,H	20	30	40	50	60
BPRC	2	12	22	34	48
Transmission error	Payload decoding rate $r_{\hat{\mathbf{P}}}$				
10%	N/A	79.4%	96.4%	99.9%	100%
9%	N/A	81.4%	96.8%	99.9%	100%
8%	N/A	85.0%	98.2%	99.9%	100%
7%	N/A	86.5%	99.8%	99.9%	100%
6%	N/A	86.7%	99.6%	99.9%	100%
5%	N/A	89.7%	100%	100%	100%
4%	N/A	93.9%	100%	100%	100%
3%	N/A	95.2%	100%	100%	100%
2%	N/A	98.7%	100%	100%	100%
1%	N/A	99.8%	100%	100%	100%
0%	N/A	100%	100%	100%	100%

3. ANALYZE THE PERFORMANCE IN A NOISE FREE COMMUNICATION CHANNEL

3.1 Find the Bit Position Index of Shifted Locations

In the circular coding algorithm, the bit position index is circularly shifted row by row with a given number D. Given the bit position index at location (m, n) in a circularly shifted data array, we would like to know the bit position index at a shifted location.

The bit position index at location (m, n) is denoted as $\mathcal{P}(m, n)$. Now let us move the bit position to a new location $(m+\Delta m, n+\Delta n)$, so the new bit position is $\mathcal{P}(m+\Delta m, n+\Delta n)$.

3.1.1 Row shift

First, we only consider the row shift. If there are Δm rows shift down, and the row-to-row offset is D, then the total bit shift from location $\mathcal{P}(m, n)$ to $(m+\Delta m, n)$ can be denoted as $f(\Delta m, 0)$.

$$f(\Delta m, 0) = \Delta m \cdot D \tag{3.1}$$

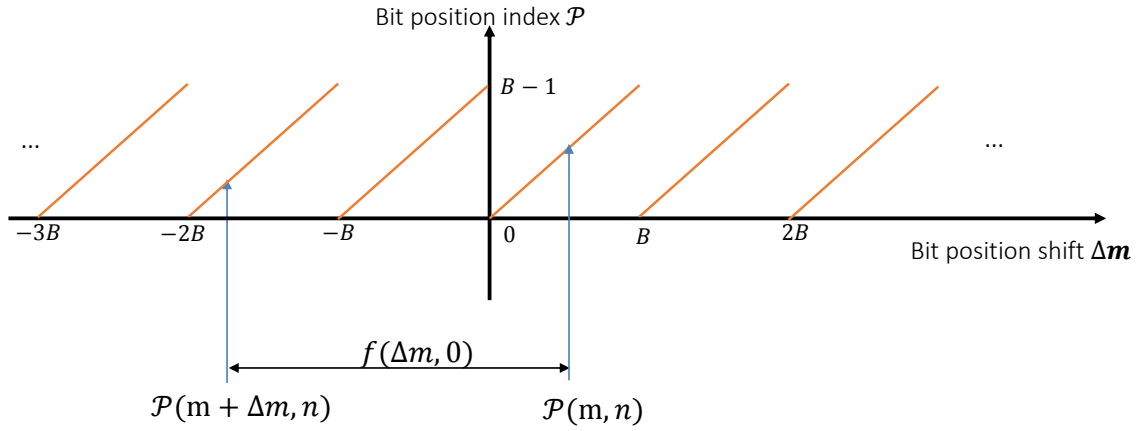


Figure 3.1. *Example of the bit position index for shifted location $(\Delta m, n)$. Assume the bit position index at the un-shifted location is 0.*

If we set the bit position at the original position (m, n) to be zero, then the bit position at the shifted position $(m + \Delta m, n + \Delta n)$ is a function of Δm, the row-to-row shift D, and the payload bit length B. See Fig. 3.1.

$$\begin{aligned} \mathcal{P}(m + \Delta m, n) &= \text{mod}(f(\Delta m, 0)), B) \\ &= \text{mod}(\Delta m \cdot D, B) \end{aligned} \tag{3.2}$$

Now let us consider the bit position at the original position (m, n) to be an arbitrary number from 0 to $B - 1$, then the bit position at the shifted position $(m + \Delta m, n)$ is a function of Δm, the row-to-row shift D, the payload bit length B, and the bit position at the original location (m, n). It is given by

$$\mathcal{P}(m + \Delta m, n) = \mathcal{P}(m, n) - \Delta m \cdot D + \mathcal{Q} \tag{3.3}$$

where

$$\mathcal{Q} = \left\lceil \frac{\Delta m \cdot B - \mathcal{P}(m, n)}{B} \right\rceil \cdot B \tag{3.4}$$

Proof:

In order to find a bit position shifted from the original location with bit position value $\mathcal{P}(m, n)$, first move the origin to the left starting point of the bit position. The number of bits needed to shift is denoted as $\mathcal{Q}$. The distance from the original location to the shifted location, in pixels, is $\Delta m \cdot D$. Since the signal has a slope of one, we have this relationship:

$$\mathcal{Q} - \mathcal{P}(m + \Delta m, n) = \Delta M \cdot D - \mathcal{P}(m, n) \tag{3.5}$$

Rearranging the equation, we get

$$\mathcal{P}(m + \Delta m, n) = \mathcal{P}(m, n) - \Delta M \cdot D + \mathcal{Q} \tag{3.6}$$

3.1.2 Column shift

Now let us consider the column shift. Similarly, let us consider the bit position at the original position (m, n) as an arbitrary number from 0 to $B-1$, then the bit position at the shifted position $(m, n+\Delta n)$ is a function of Δn, the row-to-row shift D, the payload bit length B, and the bit position at the original location (m, n). We find that

$$\mathcal{P}(m, n+\Delta n) = \mathrm{mod}(\mathcal{P}(m, n) + \Delta n, B) \tag{3.7}$$

3.1.3 Combine row shift and column shift

Now, let us combine the column and row shift. We get

$$\begin{aligned} &\mathcal{P}(m+\Delta m, n+\Delta n) \\ &= \mathrm{mod}(\mathcal{P}(m+\Delta m, n) + \Delta n, B) \\ &= \mathrm{mod}\left(\mathcal{P}(m, n) - \Delta m \cdot D + \left\lceil \frac{\Delta m \cdot B - \mathcal{P}(m, n)}{B} \right\rceil \cdot B, B\right) \end{aligned} \tag{3.8}$$

This equation shows that for given B, D and two locations (m_1, n_1) and (m_2, n_2), if their bit position indices are the same, say $\mathcal{P}(m_1, n_1) = \mathcal{P}(m_2, n_2)$, then for any shift $(\Delta m, \Delta n)$ for these two original locations, respectively, the shifted bit position indices will also be the same; and vice versa.

This relationship can be written as

$$\begin{aligned} &\mathcal{P}(m_1, n_1) = \mathcal{P}(m_2, n_2) \\ \Leftrightarrow\ &\mathcal{P}(m_1+\Delta m, n_1+\Delta n) = \mathcal{P}(m_2+\Delta m, n_2+\Delta n) \end{aligned} \tag{3.9}$$

3.2 Canonical Crop Window Location Set (CCWLS)

From Eq. 3.9 we can see that for a given crop window of data cropped from a data array (with payload length B, row-to-row shift D, and interleaving phase period V), the entire set of bit position indices of this cropped data is determined by the starting bit position index at the upper left corner of the crop window.

In order to evaluate every possible cropped data arrangement, we need to consider every unique starting bit position index of the crop window. There exists a minimum size rectangular region of data that includes every unique starting position index for the crop window. We call this minimum size rectangular region the Canonical Crop Window Location Set (CCWLS).

3.2.1 Size of CCWLS

First, let us ignore the interleaving phase period V. We have the parameters of payload length B and row to row shift D. To cover every possible starting index, we note that a rectangular region with size $1 \times B$ covers every possible starting position index from 0 to $B - 1$. So we have the CCWLS rectangular region size of $1 \times B$.

Second, let us consider the interleaving phase period V. Now we have parameters B, V, and D. In this case, the crop window may consist of a combination of payload rows and phase rows. These two sets of rows comprise different data sets. So we need to be able to evaluate every possible cropped data arrangement in this pair of sets.

In a rectangular region of data with row index m, the first phase row may be any one of the V rows with index $0 \leq m \leq V - 1$. In any phase row, the phase code repeats with period B. Thus, the CCWLS only need to contain a length B segment of one phase row. This means that there are V different pairs of payload and phase data sets. So we need to expand the CCWLS rectangular region to size $V \times B$.

Finally, let us consider the sub-sampling mask $\mathcal{M}(m, n)$. It is a checkerboard pattern defined by $\mathcal{M}(m, n) = \text{mod}(m + n, 2)$.

Since the length of the payload is always an odd number, we have that the sub-sampling mask value at one position and the sub-sampling mask value at the position that is shifted by B bits in the row will change. Therefore, any consecutive set of $2B$ rows will include each of the B unique bit position indices in an unmasked position.

The same argument applies when we consider the interleaving phase period V. So we conclude that the CCWLS rectangular region has size $V \times 2B$.

3.2.2 An example of the CCWLS

Let the payload bit length $B = 7$, the bit index $j = 0, 1, ..., 6$. The row to row shift $D = 1$, and the interleaving phase period $V = 3$.

Let the index numbers that are written in the color red indicate that this is a phase position, and the index numbers that are written in black indicate that this is a payload position. Note that half of the data has been masked to simulate the halftoning process that will cause half of the sub-cells to be abstentions.

In the example shown in Fig. 3.2, we can see that for a randomly selected crop window $\mathbf{W}_1$, we will be able to find the related crop window $\mathbf{W}_1^*$ that starts within the CCWLS, and contains the same bit indices for the entire crop window.

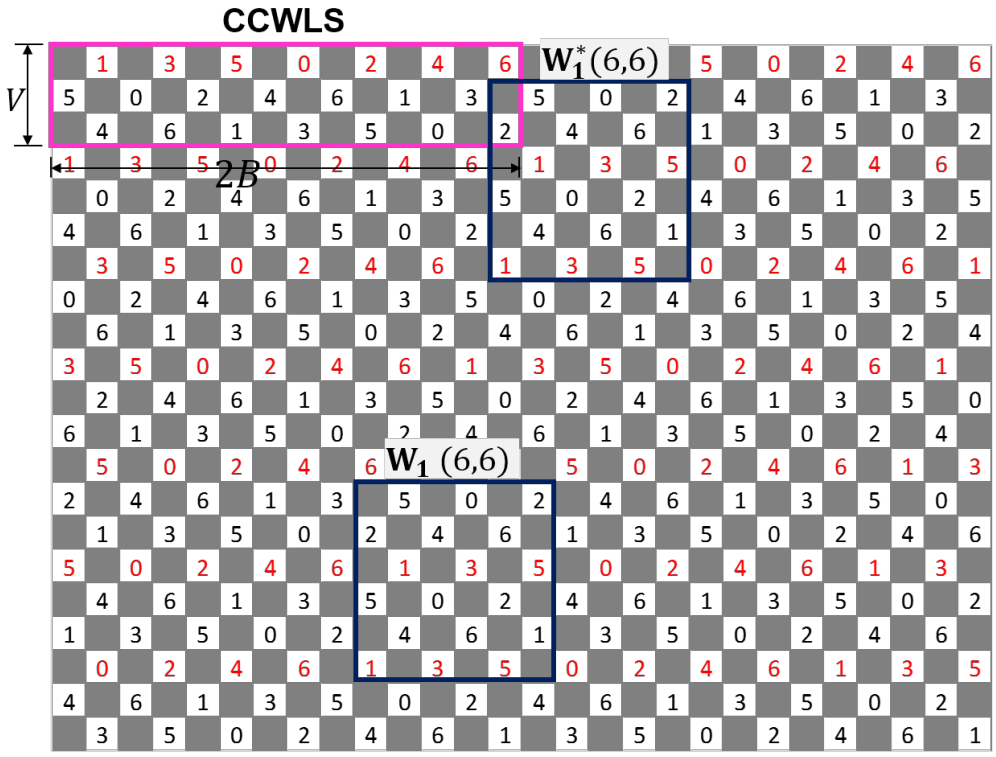


Figure 3.2. *A example of the canonical crop window location set (CCWSL). Here, the payload bit length $B = 7$, and the bit index $j = 0, 1, ..., 6$. The row to row shift $D = 1$, and the interleaving phase period $V = 3$. The CCWLS has size $V \times 2B$. Any crop window $\mathbf{W}$ has a related crop window $\mathbf{W}^*$ with a bit position within the CCWLS that contains exactly the same bit repeating positions, including the phase rows, which can be found in $\mathbf{W}^*$.*

3.3 Calculate the Bit Position Repeating Count (BPRC) in a Crop Window

In a crop window of a data set, the bit position for a bit position index will repeat due to the repeats within rows, and shifts from row to row. We are interested to calculate the bit position repeating count for each bit position index.

3.3.1 Develop a closed form equation to calculate the BPRC in a crop window

Let us denote by $\mathbf{W}$ a crop window with row index $0 \leq h \leq H-1$ and column index $0 \leq w \leq W-1$. Without loss of generality, we assume that the bit position index $j = 0$ at the starting position in $\mathbf{W}$, where $(h, w) = (0, 0)$.

Some notations:

- B: bit length of payload
- D: row to row bit position shift
- V: interleaving phase period
- W, H: width and height of the crop window
- X_j: the original payload value at position j
- Y_j: the estimated payload value at position j
- p: the transmission error

1. We find the column index $q(h, j)$ in row h of the crop window, where the bit position index j first appears. Note that $q(h, j)$ will be inside the crop window $\mathbf{W}$, if $q(h, j) \leq W-1$ and $h \leq H-1$; otherwise, $q(h, j)$ will be outside of $\mathbf{W}$.

   By observing how the bit position index repeats with period B, and is shifted by D from row to row, we find that

$$q(h, j) = \text{mod}(j + hD, B) \tag{3.10}$$

2. For each row in the crop window $\mathbf{W}$, we will find the number of positions $N_{n,j}$ for bit position index j, starting from the column index $q_h(j)$ to the end of row h. Here $h < H$ since we only consider rows within the crop window $\mathbf{W}$. This can be split into two cases, either $q(h,j) \leq W-1$ (inside of $\mathbf{W}$), or $q(h,j) \geq W$ (outside of $\mathbf{W}$), and can be formulated as

$$N_{n,j} = \begin{cases} W - q(h,j), & \text{if } q(h,j) < W \\ 0, & \text{if } q(h,j) \geq W \end{cases} \tag{3.11}$$

The above two cases can be combined to form Eq. 3.12

$$N_{h,j} = W - \min(q(h,j), W) \tag{3.12}$$

For example, let us evaluate the bit repeat count for bit position index $j = 1$ for a crop window size of $H \times W = 6 \times 6$ as shown in Fig. 3.3. The bit position index $j = 0$ is colored in red. In the first row, $q(0,0) = 0$, the first repeat bit position (the red colored position) is within the crop window, and $N_{0,0} = 8$; In the fourth row $q(3,0) = 9$, and the first repeat bit position is outside of the crop window, so $N_{3,0} = 0$.

3. We can find the number of times that the bit position index j appears in row h, denoted as $\mathcal{B}_{\text{row}}(h,j;B,D,W)$, in terms of $N_{h,j}$. This simple relationship can be written as

$$\mathcal{B}_{\text{row}}(h,j;B,D,W) = \left\lceil \tilde{\mathcal{B}}_{\text{row}}(h,j,q;B,D,W) \right\rceil \tag{3.13}$$

where

$$\begin{aligned} \tilde{B}_{\text{row}}(h,j;B,D,W) &= \frac{N_{h,j}}{B} \\ &= \frac{W - \min(q(h,j),W)}{B} \end{aligned} \tag{3.14}$$

which is the bit position repeat count for bit position index j, in row h, and before rounding to the next larger integer.

Let us combine the above two equations to get

$$\begin{aligned}\mathcal{B}_{\text{row}}(h, j; B, D, W) &= \left\lceil \frac{N_{h,j}}{B} \right\rceil \\ &= \left\lceil \frac{W - \min(q(h, j), W)}{B} \right\rceil\end{aligned} \tag{3.15}$$

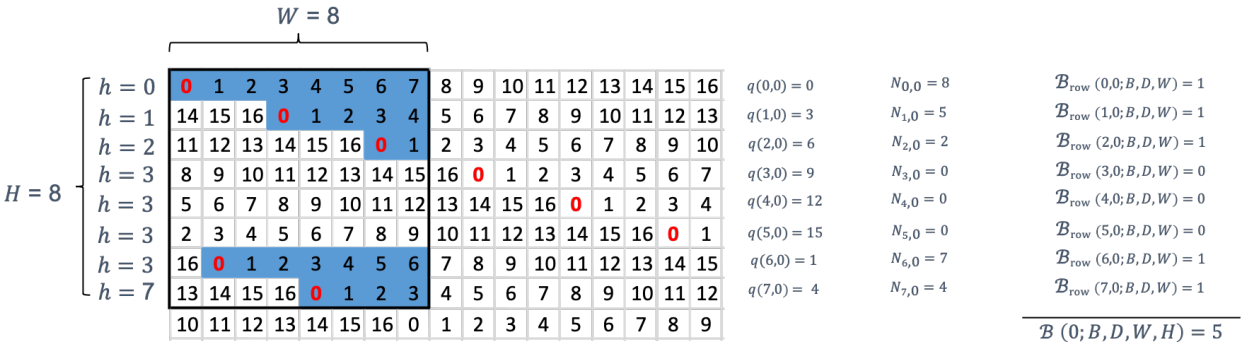


Figure 3.3. *Illustration of the bit repeat count. Here, the crop window size is 8 × 8, and the row to row shift $D = 3$. Total bit repeat count for bit index $j = 0$ is 5.*

The above illustrations are all based on the bit position index $j = 0$. However, the formulas we derived as shown in Eq. 3.15 are applicable to any bit position $j \in [0, B-1]$ within a single row h.

4. To obtain the bit position repeat count for all the rows within the crop window, we sum over the rows.

$$\mathcal{B}(j; B, D, W) = \sum_{h=0}^{H-1} \mathcal{B}_{\text{row}}(h, j, q; B, D, W) \tag{3.16}$$

Then, substituting the expressions for $\mathcal{B}_{\text{row}}(h, j; B, D, W)$ and $q(h, j)$ into the above equation, we get

$$\mathcal{B}(j; B, D, W) = \sum_{h=0}^{H-1} \left\lceil \frac{W - \min(q, W)}{B} \right\rceil \tag{3.17}$$

where $q = \text{mod}(j + hD, B)$.

5. Now let us consider the sub sampling mask $\mathcal{M}[m,n]$. It is a checkerboard pattern defined as

$$\mathcal{M}[m,n] = \text{mod}(m+n,2) \tag{3.18}$$

If we shift the position from (m,n) a number of bits Δm and Δn, respectively, then the new mask value at the shifted location can be shown as

$$\mathcal{M}[m+\Delta m, n+\Delta n] = \begin{cases} \mathcal{M}(m,n), & \text{if mod}(\Delta m+\Delta n,2)=0 \\ 1-\mathcal{M}(m,n), & \text{if mod}(\Delta m+\Delta n,2)=1 \end{cases} \tag{3.19}$$

Let $k \in \{0,1\}$ be the mask value at the upper left corner of the crop window. Then to count the bit position repeat count, we first need to apply the mask to every bit occurrence within the crop window.

There could be more than one occurrence of bit position j in a given row of the crop window. Since B is always odd, we know that if bit position j is masked in the first position of any row, then it will not be masked in its next occurrence in that row, if there is one. This pattern of alternating appearances of bit position j , either masked or unmasked, will repeat with period B until the end of the crop window row.

To account for this alternating pattern, we need to separate the occurrences of bit position j in each row, according to whether they occur in an even-numbered or an odd-numbered B-length period in that row. Here, we assume that the B-length periods are numbered starting from zero.

In a given row, if the occurrences of bit position j occur in even-numbered B-length periods, the first occurrence of bit position j will be at bit location $q(h,j)$. On the other hand, if the occurrences of bit position j occur in odd-numbered B-length periods, the first occurrence of bit position j will be at bit location $q(h,j)+B$.

Then, the total bit position occurrence count for bit position j is summed over all the rows in the crop window:

$$\mathcal{B}(j,k;B,D,W,H) = \sum_{h=0}^{H-1} \mathcal{B}_{\text{row}}(h,j,k;B,D,W,H) \tag{3.20}$$

where now the bit repeat count for each row can be represented as:

$$\begin{aligned}\mathcal{B}_{\text{row}}(h,j,q;B,D,W) = &\left\lceil \frac{\tilde{\mathcal{B}}_{\text{row}}(h,j,q;B,D,W)\cdot g(k,h,q)}{2} \right\rceil \\ &+ \left\lceil \frac{\tilde{\mathcal{B}}_{\text{row}}(h,j,q+B;B,D,W)\cdot g(k,h,q+B)}{2} \right\rceil\end{aligned} \tag{3.21}$$

where

$$\tilde{\mathcal{B}}_{\text{row}}(h,j,q;B,D,W) = \frac{W - \min(q(h,j),W)}{B} \tag{3.22}$$

$$g(k,h,q) = \text{mod}(k+h+q,2) \tag{3.23}$$

$$q(h,j) = \text{mod}(j+hD,B) \tag{3.24}$$

6. Now we consider the interleaving phase with period V. In this case, the crop window $\mathbf{W}$, which has size $W \times H$, may consist of a combination of payload rows and phase rows.

   We define the set of all the row indices in $\mathbf{W}$ as

$$\mathfrak{H} = \{k : 0 \le k \le K-1\} \tag{3.25}$$

These rows will be divided into phase rows and payload rows. There will be V possible partitions of $\mathfrak{H}$ into the sets of payload row indices denoted as $\mathfrak{H}_{\text{pay}}$ and the set of phase row indices denoted as $\mathfrak{H}_{\text{pha}}$, where

$$\mathfrak{H} = \mathfrak{H}_{\text{pay}} \cup \mathfrak{H}_{\text{pha}} \tag{3.26}$$

Let v denote the first index of the first phase row in the crop window $\mathbf{W}$. Then we can define the set of phase row indices as:

$$\begin{aligned} \mathfrak{H}_{\text{pha}} &= \left\{ v + lV : l = 0, 1, \cdots, \left\lfloor \frac{H}{V} \right\rfloor, v = 0, 1, \cdots, V-1 \right\} \\ \mathfrak{H}_{\text{pay}} &= \mathfrak{H} - \mathfrak{H}_{\text{pha}} \end{aligned} \tag{3.27}$$

To account for the role of the interleaving phase in our analysis of the bit repeat count, we separate the summation over the rows h in our previous expression for the bit repeat count according to the partition between phase rows and payload rows.

So the number of times a given bit position index j repeats within the crop window $\mathbf{W}$ is given by

$$\mathcal{B}_{\text{pay}}(j, k; B, D, W, H) = \sum_{h \in \mathfrak{H}_{\text{pay}}} \mathcal{B}_{\text{row}}(h, j, k; B, D, W, H) \tag{3.28}$$

$$\mathcal{B}_{\text{pha}}(j, k; B, D, W, H) = \sum_{h \in \mathfrak{H}_{\text{pha}}} \mathcal{B}_{\text{row}}(h, j, k; B, D, W, H) \tag{3.29}$$

where $\mathcal{B}_{\text{row}}(h, j, k; B, D, W, H)$ is defined in Eq. 3.21.

3.3.2 Validate the formula

To validate the correctness of our formula in Eq. 3.28, we simulate the circular coding bit repeat, crop the window in different locations, and count how many bits are actually repeated in the crop window. We choose $B = 17$, $V = 3$, and the test row to row shift from 0 to 16 bits. The crop window size is tested from 0 to 22 bits. The minimum bit repeat

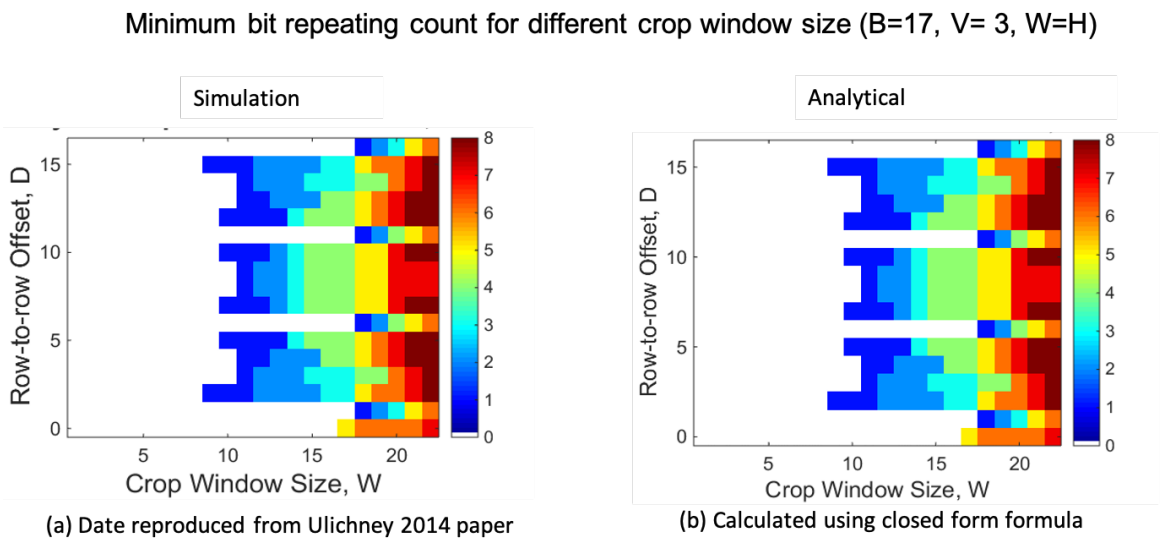


Figure 3.4. *Minimum bit repeat count for $B = 17$, $V = 3$ (a) from reproduced from simulation [21] and (b) from closed form Eq. 3.17. These two approaches show the same results.*

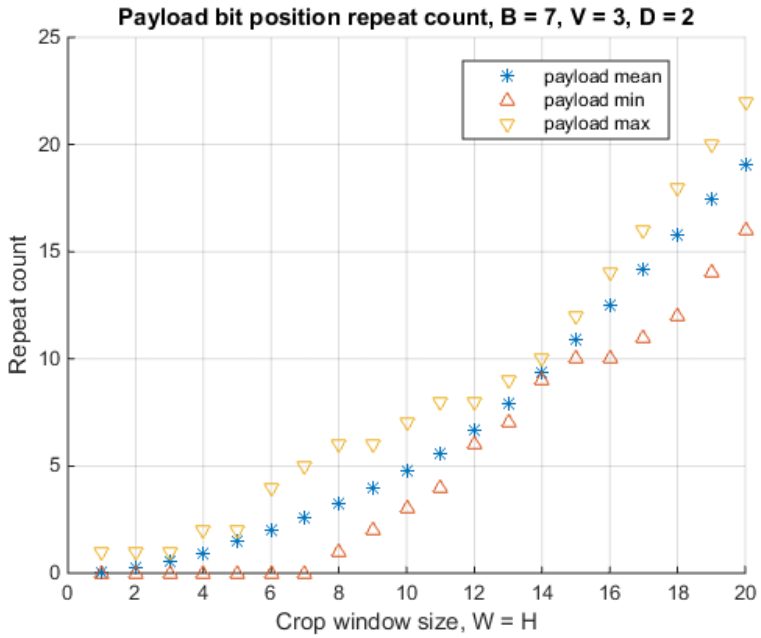
count is checked from all the starting positions of the crop window within the CCWLS set. The results for both closed form calculation and simulation are the same as can be seen in Fig. 3.4.

3.4 Performance Review

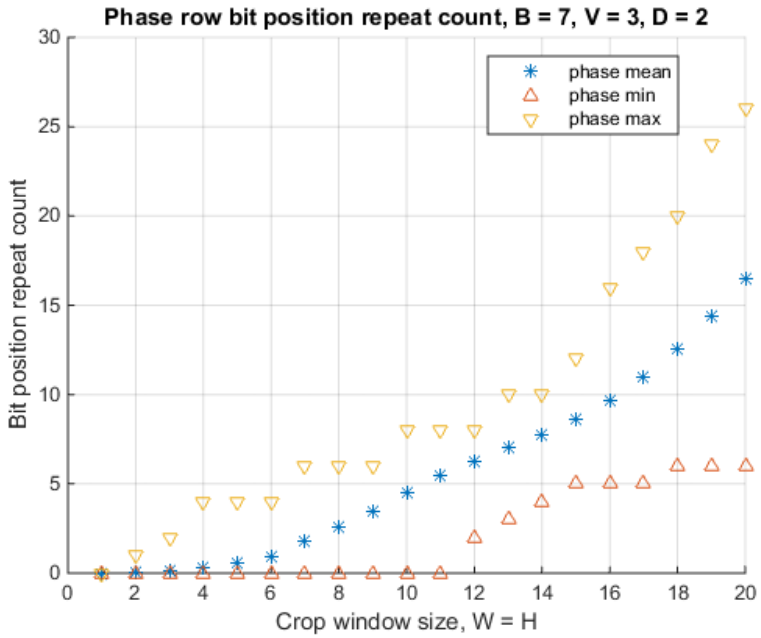
To simulate the decoding rate for each level of transmission error and crop window size, we use the crop window of data array that starts at every possible position in the CCWLS, and add different trials of transmission error rate to each crop window of data. Recall that the CCWLS size is $V \cdot 2B$.

In the decoding process, we will be able to decode the candidate payload $\tilde{\mathbf{P}}$, which is a circularly shifted version of the payload from the payload rows and the circular shift $\tilde{C}$ from the phase rows. Then using $\tilde{\mathbf{P}}$ and $\tilde{C}$, we will be able to figure out the decoded payload $\hat{\mathbf{P}}$. Without loss of generality, we use the square crop window size $W = H$ in the experiments.

Using the formulas developed in Eq. 3.28 and Eq. 3.29, and examining every possible crop window starting position in the CCWLS, we can calculate the BPRC for the payload



(a)



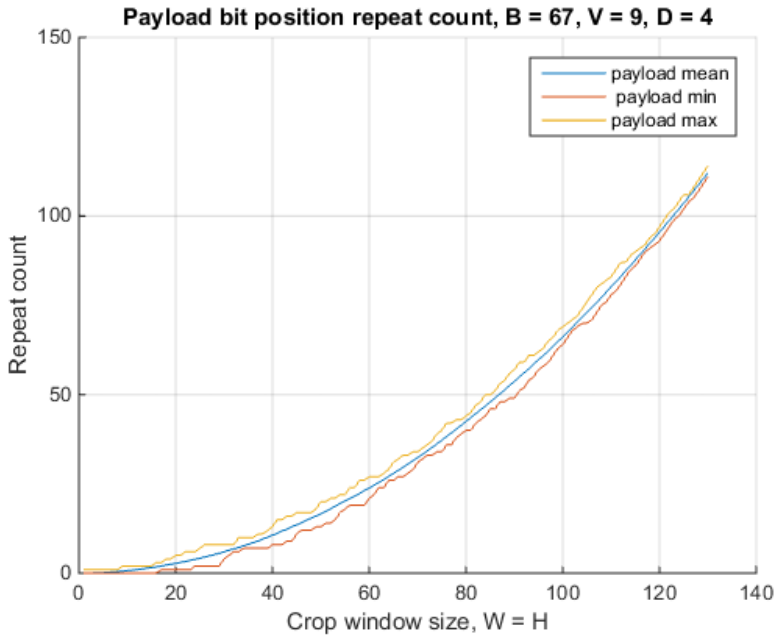
(b)

Figure 3.5. *Bit position repeat count for (a) payload and (b) phase for each bit position and for all possible starting locations of the crop window within the CCWLS. Payload bit length $B = 7$, row to row shift $D = 2$, interleaving phase period $V = 3$. The yellow star is the maximum bit repeat count over all the crop windows that start at the different starting positions in the CCWSL; the orange star is the minimum bit repeat count over all the crop windows in the CCWLS; the blue star is the mean.*

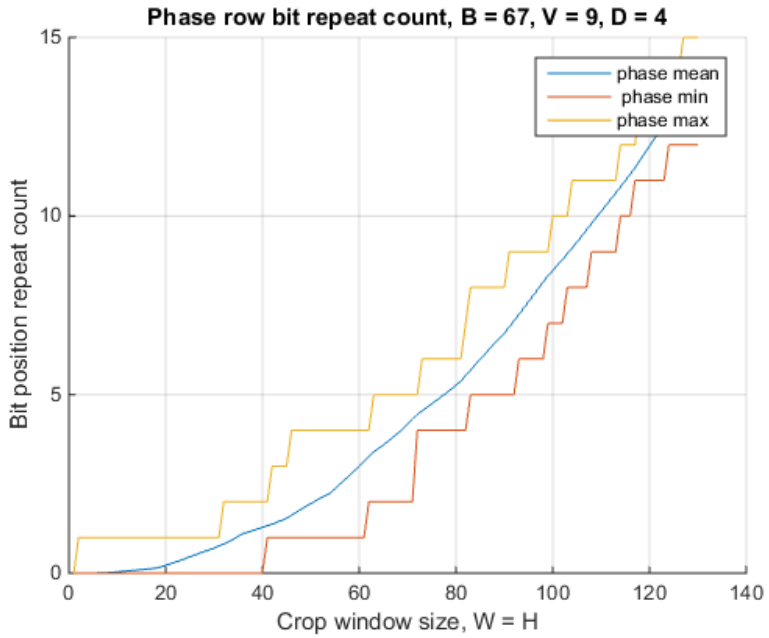
and phase for every bit position. The payload and phase bit repeat count for payload bit length B of 7 is calculated and shown in Fig. 3.5 (a) and (b), respectively. Then we increase the bit length B to 67 and ran the same simulation, see Fig. 3.6.

3.5 Conclusion

In this chapter, we analyze decoding performance of the circular coding method in a noise free communication channel. For each bit in a shifted location, we mathematically determined the bit position index, and introduced the concept of the Canonical Crop Window Location Set (CCWLS). In addition, we developed a closed form equation to calculate the Bit Position Repeating Count (BPRC) in each crop window. Now we are able to predict the number of bit repeats for each size of the crop window without the simulation. The theoretical result was validated with the simulation result.



(a)



(b)

Figure 3.6. *Bit position repeat count for (a) payload and (b) phase for each bit position and for all possible starting locations of the crop window within the CCWLS. Payload bit length $B = 67$, row to row shift $D = 4$, interleaving phase period $V = 6$. The yellow star is the maximum bit repeat count over all the crop windows that start at the different starting positions in the CCWSL; the orange star is the minimum bit repeat count over all the crop windows in the CCWLS; the blue star is the mean.*

4. ANALYZE THE DECODING PERFORMANCE IN A NOISY CHANNEL USING PROBABILITY MODELING

We use simulation combined with a theoretical framework to study the payload decoding rate with different levels of the transmission error rate [58].

The process of developing this methodology consists of four steps:

1. Model the communication channel and transmission error;
2. Represent the decoding process;
3. Develop a closed form solution for the probability of successful decoding rate;
4. Simulate each step in this encoding/decoding process and validate the decoding result using the closed form solution.

4.1 Model the Communication Channel and Transmission Error

If the information sequence has length of k bits, and the code word has length of n bits, then the ratio $R = \frac{k}{n}$ is called the code rate. This is the number of information bits entering the encoder for each transmission symbol. If the output of n bits code words only depends on the k bit information message, then the encoding is called **memory-less**.

Regarding the types of errors in memory-less channels, the noise affects each transmitted symbol independently. For example, in a binary-symmetric-channel, each transmitted bit has a probability of p being received incorrectly and a probability of $1 - p$ of being received correctly, independently of other transmitted bits [34]. See Fig. 4.1.

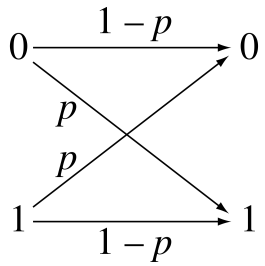


Figure 4.1. *Binary Symmetric Channel with probability of transmission error p.*

The message embedded in the image is memory-less, so we use a Binary Symmetric Channel to model the communication channel. It has binary input and output, with a probability of transmission error ϵ, i.e. the probability of switching values between 0 and 1. So the probability of successful transmission of one bit is $P(\text{Success}) = 1 - \epsilon$, and the probability of failure of transmission of one bit is $P(\text{Fail}) = \epsilon$. We assume that the probability of successful transmission at each position is independent and identically distributed (i.i.d.).

4.2 Represent the Decoding Process

As noted in Section 2.3, the crop window of the data array $\hat{\mathbf{v}}$ is mixed with payload rows and phase rows, with a fixed interleaving period V. But the starting row index of the phase is unknown.

In order to separate the phase from the payload, we try every possible starting row index of the phase (V possible starting row indices), and select the one with highest confidence. Once we try to remove M "phase" rows in the data array, either correctly or incorrectly, there will be $R - M$ rows of data left. There are only two possible cases:

- Case 1: get all the phase rows out, leave $R - M$ repeats of payload rows. There is only one chance over the V possible positions that this case will occur.
- Case 2: get none of the phase rows out, leave $R - M$ rows that are a mixture of payload and phase rows. There will be $V - 1$ chances for this to occur. So the number of payload rows left is $R - 2M$. Define $N = R - 2M$.

Thus, the relationship between $R, M,$ and N is illustrated in Fig. 4.2. And an example of the two cases is shown in Fig. 4.3

In either case, the remaining data array contains the common N rows of payload repeats, and the M rows of payload or phase repeats.

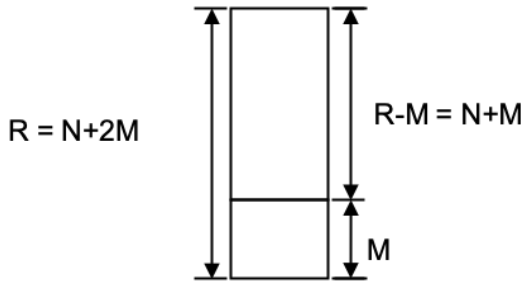


Figure 4.2. *Illustration of bit position repeat count. The total bit position repeat count in a crop window of data is R, the phase bit position repeat count in a crop window of data is M, and the payload bit position repeat count in a crop window of data is $R - M$. We define $N = R - 2M$. Thus, $R - M = N + M$.*

4.2.1 Model the status of change as a random variable

Once the data is transferred in the communication channel, the bit values might be switched due to a transmission error. We model the switch of a bit value as a random variable, and it does not depend on the bit position, so it is i.i.d. for each bit.

Then, for each case (Case 1 gets all the phase rows out, or Case 2 gets none of the phase rows out), here is the mathematical representation of the random variables (switch value for or not) at each data array position:

- For each column of data (each bit position index j) and row of data (each bit repeat index i), let the status of switching its original value be the random variables $X_i^{(j)}$ and $Z_i^{(j)}$ for payload and phase, respectively. So $j = 0, \cdots, B-1$, and $i = 1, 2, \cdots, M$ for Z; and $j = 0, \cdots, B-1$ and $i = 1, 2, \cdots, R - M = N + M$ for X.
- $X_i^{(j)} = 0$: status NOT changed at bit position j and row position i. $P\{X_i^{(j)} = 0\} = 1 - \epsilon$.
- $X_i^{(j)} = 1$: payload status changed at bit position j and row position i. $P\{X_i^{(j)} = 1\} = \epsilon$.
- Similarly for $Z_i^{(j)}$, $Z_i^{(j)} = 1$: phase status changed at bit position j and row position i.

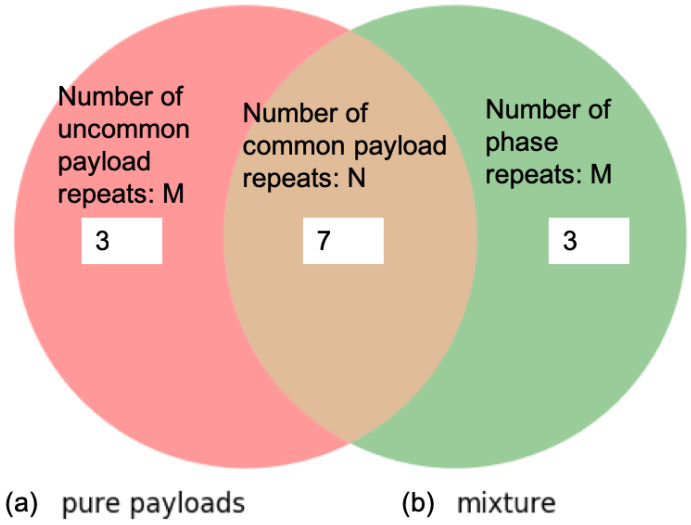


Figure 4.3. *Illustration of the (a) pure payload subset in Case 1 and (b) mixture of payload and phase subset in Case 2, using a Venn diagram. In this example* $N = 7$, $M = 3$, *and* $R = 13$.

Thus, the two sets (pure payload set and mixture of payload and phase set) can be written as shown in in Eqs. 4.1 and 4.2, respectively.

$$\mathcal{S}_1 = \begin{bmatrix} X_1^{(0)} & X_1^{(1)} & ... & X_1^{(j)} & ... & X_1^{(B-1)} \\ X_2^{(0)} & X_2^{(1)} & ... & X_2^{(j)} & ... & X_2^{(B-1)} \\ X_3^{(0)} & X_3^{(1)} & ... & X_3^{(j)} & ... & X_3^{(B-1)} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ X_N^{(0)} & X_N^{(1)} & ... & X_N^{(j)} & ... & X_N^{(B-1)} \\ \\ X_{N+1}^{(0)} & X_{N+1}^{(1)} & ... & X_{N+1}^{(j)} & ... & X_{N+1}^{(B)} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ X_{N+M}^{(0)} & X_{N+M}^{(1)} & ... & X_{N+M}^{(j)} & ... & X_{N+M}^{(B-1)} \end{bmatrix} \tag{4.1}$$

$$\mathcal{S}_2 = \begin{bmatrix} X_1^{(0)} & X_1^{(1)} & ... & X_1^{(j)} & ... & X_1^{(B-1)} \\ X_2^{(0)} & X_2^{(1)} & ... & X_2^{(j)} & ... & X_2^{(B-1)} \\ X_3^{(0)} & X_3^{(1)} & ... & X_3^{(j)} & ... & X_3^{(B-1)} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ X_N^{(0)} & X_N^{(1)} & ... & X_N^{(j)} & ... & X_N^{(B-1)} \\ \\ Z_1^{(0)} & Z_1^{(1)} & ... & Z_1^{(j)} & ... & Z_1^{(B-1)} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ Z_M^{(0)} & Z_M^{(1)} & ... & Z_M^{(j)} & ... & Z_M^{(B-1)} \end{bmatrix} \tag{4.2}$$

Note that for a crop window of data $\hat{\mathbf{r}}$, we are able to calculate the bit repeat count for each bit position index. The bit repeat count for each bit position index might be slightly different, depending on the size and position of the crop window. Here, we assume that each bit position index has the same repeat number R.

4.2.2 Model the data array after corruption by transmission errors as a sequence of random variables

For the original payload P and the phase U both with length B bits, we can write their original values as the data arrays:

$$P = [P^{(0)}, P^{(1)}, \ldots, P^{(B-1)}] \tag{4.3}$$

$$U = [U^{(0)}, U^{(1)}, \ldots, U^{(B-1)}] \tag{4.4}$$

Note that we already defined the status of switching value as the random variables in Eqs. 4.1 and 4.2, so we can define the value of the data sets in Cases 1 and 2 at each position, as the original value XOR the random variable of a status change at that position, as shown in Eqs. 4.5 and 4.6.

$$\tilde{\mathcal{S}}_1 = \begin{bmatrix} \tilde{X}_1^{(0)} & \tilde{X}_1^{(1)} & \ldots & \tilde{X}_1^{(j)} & \ldots & \tilde{X}_1^{(B-1)} \\ \tilde{X}_2^{(0)} & \tilde{X}_2^{(1)} & \ldots & \tilde{X}_2^{(j)} & \ldots & \tilde{X}_2^{(B-1)} \\ \tilde{X}_3^{(0)} & \tilde{X}_3^{(1)} & \ldots & \tilde{X}_3^{(j)} & \ldots & \tilde{X}_3^{(B-1)} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ \tilde{X}_N^{(0)} & \tilde{X}_N^{(1)} & \ldots & \tilde{X}_N^{(j)} & \ldots & \tilde{X}_N^{(B-1)} \\ \\ \tilde{X}_{N+1}^{(0)} & \tilde{X}_{N+1}^{(1)} & \ldots & \tilde{X}_{N+1}^{(j)} & \ldots & \tilde{X}_{N+1}^{(B)} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ \tilde{X}_{N+M}^{(0)} & \tilde{X}_{N+M}^{(1)} & \ldots & \tilde{X}_{N+M}^{(j)} & \ldots & \tilde{X}_{N+M}^{(B-1)} \end{bmatrix} \tag{4.5}$$

where $\tilde{X}_i^{(j)} = X_i^{(j)} \bigotimes P^{(j)}$, for $i = 1, 2, \ldots, N+M$ and $j = 0, 1, ..., B-1$.

Similarly,

$$\tilde{\mathcal{S}}_2 = \begin{bmatrix} \tilde{X}_1^{(0)} & \tilde{X}_1^{(1)} & ... & \tilde{X}_1^{(j)} & ... & \tilde{X}_1^{(B-1)} \\ \tilde{X}_2^{(0)} & \tilde{X}_2^{(1)} & ... & \tilde{X}_2^{(j)} & ... & \tilde{X}_2^{(B-1)} \\ \tilde{X}_3^{(0)} & \tilde{X}_3^{(1)} & ... & \tilde{X}_3^{(j)} & ... & \tilde{X}_3^{(B-1)} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ \tilde{X}_N^{(0)} & \tilde{X}_N^{(1)} & ... & \tilde{X}_N^{(j)} & ... & \tilde{X}_N^{(B-1)} \\ \\ \tilde{Z}_1^{(0)} & \tilde{Z}_1^{(1)} & ... & \tilde{Z}_1^{(j)} & ... & \tilde{Z}_1^{(B-1)} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ \tilde{Z}_M^{(0)} & \tilde{Z}_M^{(1)} & ... & \tilde{Z}_M^{(j)} & ... & \tilde{Z}_M^{(B-1)} \end{bmatrix} \tag{4.6}$$

where $\tilde{X}_i^{(j)} = X_i^{(j)} \otimes P^{(j)}$, for $i = 1, 2, \ldots, N$ and $j = 0, 1, ..., B-1$.

And $\tilde{Z}_i^{(j)} = Z_i^{(j)} \otimes U^{(j)}$, for $i = 1, 2, \ldots, M$ and $j = 0, 1, ..., B-1$.

Here $\otimes$ denotes the XOR operation.

4.2.3 Separate the payload and phase and then decode the payload

When we compare any two sets (the pure payload set in Case 1 and the mixture of payload and phase set in Case 2), we calculate their confidence values, and then select the one with higher value as the pure payload set.

For the pure payload data set in Eq. 4.1 and for the mixture of payload and phase data set in Eq. 4.2, the methods to estimate the bit value are the same. That is, we calculate the confidence value of each data set, and select the one with higher confidence as the pure payload set. It includes the following four steps:

1. Calculate the sample mean of each subset. Let $\bar{Y_1}^{(j)}$ denote the sample mean for the pure payload subset and $\bar{Y_2}^{(j)}$ for the mixture payload and phase subset:

$$\bar{Y_1}^{(j)} = \frac{1}{N+M} \left\{ \sum_{i=1}^{N} \tilde{X}_i^{(j)} + \sum_{i=N+1}^{N+M} \tilde{X}_i^{(j)} \right\} \tag{4.7}$$

$$\bar{Y_2}^{(j)} = \frac{1}{N+M} \left\{ \sum_{i=1}^{N} \tilde{X}_i^{(j)} + \sum_{i=1}^{M} \tilde{Z}_i^{(j)} \right\} \tag{4.8}$$

2. Calculate the estimated bit value for each data set:

$$\hat{Y_k}^{(j)} = \begin{cases} 1, & \text{if } \bar{Y_k}^{(j)} > 0.5 \\ 0, & \text{if } \bar{Y_k}^{(j)} < 0.5 \\ \text{random choice of 0 or 1}, & \text{if } \bar{Y_k}^{(j)} = 0.5 \end{cases} \tag{4.9}$$

$$\Delta_k^{(j)} = \mid \hat{Y_k}^{(j)} - \bar{Y_k}^{(j)} \mid \tag{4.10}$$

for $k = 1, 2$.

For the values of bit position repeats, we define the minority bits as the bits with value that appear less frequently than bits with the other value. The other bits with the value that appeared more than half the time are called majority bits. Thus, $\Delta^{(j)}$ is actually calculating the proportion of the minority bits for bit position index j.

3. The confidence value in Eq. 2.3 then can be calculated for each data set as Eq. 4.11:

$$\mathcal{C}_k = 1 - \frac{2}{B} \sum_{j=0}^{B-1} \Delta_k^{(j)} \tag{4.11}$$

for $k = 1, 2$.

Note that the confidence is in the range of 0 and 0.5, where the higher the confidence is, the more likely this selection is a pure payload data set. For the pure payload data set in Case 1, $k = 1$; for the mixture of payload and phase data set in Case 2, $k = 2$. And the confidence is negatively proportional to the sum of $\Delta^{(j)}$, $j = 0, 1, \ldots, B-1$.

Select the payload data set with the higher confidence value, and assign each bit its value that has been calculated in Step 2.

$$[\hat{Y}^0, \hat{Y}^1, ..., \hat{Y}^{B-1}] = \begin{cases} [\hat{Y}_1^0, \hat{Y}_1^1, ..., \hat{Y}_1^{B-1}], & \text{if } \mathcal{C}_1 > \mathcal{C}_2 \\ [\hat{Y}_2^0, \hat{Y}_2^1, ..., \hat{Y}_2^{B-1}], & \text{if } \mathcal{C}_1 < \mathcal{C}_2 \\ [\hat{Y}_1^0, \hat{Y}_1^1, ..., \hat{Y}_1^{B-1}], & \text{if } \mathcal{C}_1 = \mathcal{C}_2 \end{cases} \tag{4.12}$$

There will be totally V possible starting row positions for the phase rows. For a successful decoding, it is required that the pure payload data set has higher confidence than any mixture payload data set in each comparison.

4.3 Develop a Closed Form Solution for the Probability of Successful Decoding

Recall that the decoding process has three steps that involves the error estimation: 1. separate the phase from the payload; 2. decode the standard form of the payload; 3. decode the phase. We need to estimate the success rate of each of these three steps, and combine the success rate of these three to get the final decoding rate.

4.3.1 Step 1: Compute the probability of separating the phase from the payload

The phase rows and payload rows are selected based on the confidence estimation, see Eqs. 4.11 and 4.12. Note that in order to correctly decode the payload, we can conclude that at least half of the bits need to retain their original value during the transmission. So we can assume that the probability of transmission error $\epsilon < 0.5$. Higher confidence is equivalent to lower uncertainty, and equivalent to fewer minority bits. Thus, the probability of separating the phase from the payload can be computed by summing the probabilities that the pure payload data set has fewer minority bits than the mixed phase and payload data set, for each possible number of bits that switched value in the pure payload data set.

We first consider the easier case that the payload only has one bit. Then, we extend the payload bit length to multiple bits.

Payload bit length of one: $B = 1$

Let the payload length be one bit only, i.e. $B = 1$. So the phase is also one bit. Note this cannot be done in the real case.

The original payload value is $P^{(1)}$, and the phase value is $U^{(1)}$. So there are two possible situations: the original payload value is either the same or different, compared with the phase value (i.e. $P^{(1)} = U^{(1)}$, or $P^{(1)} \neq U^{(1)}$).

After we rearrange the crop window of the data array, let us assume that there are $M+N$ bits of repeats. Thus, the bit repeats in case 1 (get all phases out) and in case 2 (get none of the phases out) will be simplified as $\mathcal{S}_1|_{B=1}$ and $\mathcal{S}_2|_{B=1}$, defined in Eq. 4.13.

$$\mathcal{S}_1|_{B=1} = \begin{bmatrix} X_1 \\ X_2 \\ X_3 \\ \vdots \\ X_N \\ \\ X_{N+1} \\ \vdots \\ X_{N+M} \end{bmatrix} \text{ and } \mathcal{S}_2|_{B=1} = \begin{bmatrix} X_1 \\ X_2 \\ X_3 \\ \vdots \\ X_N \\ \\ Z_1 \\ \vdots \\ Z_M \end{bmatrix} \tag{4.13}$$

Let $\mathbb{A}$ represent the data set in Case 1, i.e. the pure payload data set, and let $\mathbb{B}$ represent the data set in Case 2, i.e. the mixture of payload and phase.

For the comparison of the bit repeats in Cases 1 and 2, there are always N bits of payload $X_1, X_2, \ldots, X_N$ shared in common. In other words, if any bits have switched value in the common payload bits area, they will be the same for the data sets $\mathbb{A}$ and $\mathbb{B}$. But the status of switching value in the remaining parts are different and independent of each other.

Our goal is to find the probability that the total number of minority bits in data set $\mathbb{A}$ is less than the number of minority bits in data set $\mathbb{B}$.

We define α as the number of minority bits in data set $\mathbb{A}$, and β as the number of minority bits in data set $\mathbb{B}$. And we assume that fewer than half of the bits have switched value. Then, the number of minority bits is the same as the number of bits that switched value in each data set. For each number of minority bits in data set $\mathbb{B}$, there should be fewer minority bits in data set $\mathbb{A}$, or $\alpha < \beta$. Also, the number of minority bits α in data set $\mathbb{A}$ should be fewer than half of the total number of bit repeats. Thus, we define:

$$\mathcal{K} = \lfloor \frac{M+N}{2} - 1 \rfloor \tag{4.14}$$

So the number of minority bits α and β can be any number between zero and $\mathcal{K}$. The α bits of positions that switched value could be in either the N bits of common payload or the M bits of uncommon payload in data set $\mathbb{A}$.

By the nature of the circular encoding process, the number of payload bit repeats N is much larger than the number of phase bit repeats M in each data set, i.e. $N \gg M$. So if the number of bits whose values are switched in the data set $\mathbb{A}$ is no more than the number M, i.e $\alpha \in [0, M]$, then these bits can be anywhere in the payload or phase. Otherwise, if the number of bits with switched values is greater than M, there will be at maximum M bits with switched values in the uncommon payload in data set $\mathbb{A}$ or the phase in data set $\mathbb{B}$, and the remaining bits that switched value will be in the common payload.

As noted before, with the assumption that fewer than half of the bits switched value, the number that switched value in the pure payload data set is the minority bit number α. Among these α bits that switched value, let m denote the number of bits that switched value in the uncommon payload. Thus,

- Part 1: When $\alpha \in [0, M]$, then the number of bits m that switched value in the data set $\mathbb{A}$ in the uncommon payload could be any number between 0 and α, or $m \in [0, \alpha]$. And the number of bits that switched value in the data set $\mathbb{A}$ in the common payload is $\alpha - m$.

- Part 2: When $\alpha \in (M, \mathcal{K}]$, then m could be any number between 0 and the minimum of α and M, or $m \in [0, M]$. And the number of bits that switched value in the data set $\mathbb{A}$ in the common payload is also $\alpha - m$.

Let k be the number of bits that switched value in the M bits of the phase in the data set $\mathbb{B}$. In order to successfully separate the pure payload data set, it is required that the minority bit number α in the data set $\mathbb{A}$ be less than the minority bit number β in data set $\mathbb{B}$.

First, if the original payload value is the same as the phase value ($P^{(1)} = U^{(1)}$), then the number α of bits that switched can take any value between 0 and half of the total number of bit repeats $\mathcal{K}$. So we sum the probability that data set $\mathbb{A}$ has fewer minority bits than data set $\mathbb{B}$ for each α value. It includes two parts. Part 1: $\alpha \in [0, M]$; Part 2: $\alpha \in (M, \mathcal{K}]$.

For the data set $\mathbb{A}$, the minority bits are the ones that switched value. There are α such bits. For the data set $\mathbb{B}$, the number of bits that switched value must be greater than α, but smaller than the total number of the bits in each data set minus α, or $M + N - \alpha$. In other words, in order to have the number of bits that switched value in the data set $\mathbb{A}$ be smaller than the number of bits that switched value in the data set $\mathbb{B}$, there should be fewer bits that switched value in the uncommon payload in data set $\mathbb{A}$ than the number that switched value in the payload in data set $\mathbb{B}$, given the condition that the payload and phase have the same original value. Thus, these five conditions need to be met:

$$\begin{cases} \alpha \in [0, M] \\ \alpha < \beta < M + N - \alpha \\ \beta = (\alpha - m) + k \\ m < k \\ m \in [0, \alpha] \end{cases}$$

From these five conditions, we can obtain the range of k:

$$m < k < M + N - 2\alpha + m \tag{4.15}$$

For Part 2, $\alpha \in (M, \mathcal{K}]$. Since there are only M bits in the uncommon payload part, there will be a maximum of M bits that could switch value in the uncommon payload, the remaining $\alpha - m$ bits must switch value in the common payload. The conditions for data set $\mathbb{B}$ are the same as those as those in Part 1. Please refer to Table 4.1 for details. Note that $\mathcal{K} = \lfloor \frac{M+N}{2} - 1 \rfloor$.

Next, we consider the case where the original payload value is different from the phase value. This is very similar to the case where the original payload value is the same as the phase value, except that for the mixture payload and phase data set $\mathbb{B}$, the phase data will be originally treated as the minority bits. This is because the number of payload bits is usually much larger than the number of phase bits, or $N \gg M$. So for any bit position j, we expect that there will be fewer phase rows than payload rows. Also, we assume that

Table 4.1. The conditions under which the pure payload selection $\mathbb{A}$ has fewer minority bits than the mixture of payload and phase selection $\mathbb{B}$, when the phase original value is the same as the payload value.

Original payload value is **same** as the phase value		
	$\mathbb{A}$ has α minority bits	$\mathbb{B}$ has β minority bits , $\beta > \alpha$
Part 1: $\alpha \in [0, M]$	m bits switch value in the uncommon payload, and $\alpha - m$ bits switch value in the common payload, where $m \in [0, \alpha]$	the same $\alpha - m$ bits switch value in the common payload, and k bits switch value in the phase, where $\beta = \alpha - m + k$, $\alpha < \beta < (M + N - \alpha)$
Part 2: $\alpha \in (M, \mathcal{K}]$	m bits switch value in the uncommon payload, and $\alpha - m$ bits switch value in the common payload, where $m \in [0, M]$	the same $\alpha - m$ bits switch value in the common payload, and k bits switch value in the phase, where $\beta = \alpha - m + k$, $\alpha < \beta < (M + N - \alpha)$

fewer than half of the bits switched their values. So the minority bits contain the bits that switched value in the common payload part in data set $\mathbb{B}$ and the bits that retained their original value in the phase part of data set $\mathbb{B}$. Thus, we again have five conditions that need to be met:

$$\begin{cases} \alpha \in (M, \mathcal{K}] \\ \alpha < \beta < M + N - \alpha \\ \beta = (\alpha - m) + (M - k) \\ m < M - k \\ m \in [0, M] \end{cases}$$

From these five conditions, we can obtain the range of k:

$$2\alpha - m - N < k < M - m \tag{4.16}$$

The details are summarized in Table 4.2. Note that $\mathcal{K} = \lfloor \frac{M+N}{2} - 1 \rfloor$.

Now we can write out the probability that the pure payload set will be correctly distinguished from the mixture subset.

Table 4.2. The conditions under which the pure payload selection $\mathbb{A}$ has fewer minority bits than the mixture of payload and phase selection $\mathbb{B}$, when the phase original value is different from the payload value.

Original payload value is **different** from the phase value		
	$\mathbb{A}$ has α minority bits	$\mathbb{B}$ has β minority bits , $\beta > \alpha$
Part 1: $\alpha \in [0, M]$	m bits switch value in the uncommon payload, and $\alpha - m$ bits switch value in the common payload, where $m \in [0, \alpha]$	the same $\alpha - m$ bits switch value in the common payload, and $M - k$ bits switch value in the phase, where $\beta = (\alpha - m) + (M - k)$, $\alpha < \beta < (M + N - \alpha)$
Part 2: $\alpha \in (M, \mathcal{K}]$	m bits switch value in the uncommon payload, and $\alpha - m$ bits switch value in the common payload, where $m \in [0, M]$	the same $\alpha - m$ bits switch value in the common payload, and $M - k$ bits switch value in the phase, where $\beta = (\alpha - m) + (M - k)$, $\alpha < \beta < (M + N - \alpha)$

First, let us define the probability mass function for the binomial distribution. We define the status of switching value at each bit position as a random variable with a binomial distribution. The probability of getting exactly a successes in A independent Bernoulli trials is given by the probability mass function:

$$P(A, a, \xi) = \begin{pmatrix} A \\ a \end{pmatrix} \xi^a (1 - \xi)^{A-a} \tag{4.17}$$

This calculates the probability that for every A bit positions, a bits switch value, when the probability of switching value at each position is ξ.

Under the assumption that the original payload value is the same as the phase value, i.e $P^{(1)} = U^{(1)}$, the probability that the number of minority bits α in the data set $\mathbb{A}$ being less than the number of minority bits β in the data set $\mathbb{B}$ is $P\{\alpha < \beta | P^{(1)} = U^{(1)}\}$. It is the cumulative distribution that for every possible number of bits i that switched value, where $i \in ([0, M-1] \cup [M, \mathcal{K}])$, the number of bits α that switched value in the data set $\mathbb{A}$ is less than the number of bits β that switched value in the data set $\mathbb{B}$.

From Table 4.1, we add the probabilities for first and second parts. For the first part, $i \in [0, M-1]$, the probability that m bits switched value in the M bits of uncommon payload is $P(M, m, \epsilon)$. For the remaining bits that switched value in the common payload area, the

probability is $P(N, i-m, \epsilon)$; and the probability that k bits switched value in the phase data set is $P(M, k, \epsilon)$. So the Part 1 conditional probability $P\{(\alpha < \beta) \cap (0 \leq \alpha \leq M) | B = 1, P^{(0)} = U^{(0)}\}$ that there are fewer minority bits in the pure payload data set $\mathbb{A}$ than in the mixture of payload and phase data set $\mathbb{B}$, when the original payload value is the same as phase value, is shown in Eq. 4.18. It is similar for Part 2, which is shown in Eq. 4.19. See Table 4.1 for the definition of Parts 1 and 2.

$$P\{(\alpha < \beta) \cap (0 \leq \alpha \leq M) | B = 1, P^{(0)} = U^{(0)}\} = \sum_{i=0}^{M-1} \left\{ \sum_{m=0}^{i} P(M, m, \epsilon) \cdot P(N, i-m, \epsilon) \cdot \sum_{k=m+1}^{M+N-2i+m-1} P(M, k, \epsilon) \right\} \tag{4.18}$$

$$P\{(\alpha < \beta) \cap (M < \alpha \leq \mathcal{K}) | B = 1, P^{(0)} = U^{(0)}\} = \sum_{i=M}^{\lfloor \frac{M+N}{2} - 1 \rfloor} \left\{ \sum_{m=0}^{M} P(M, m, \epsilon) \cdot P(N, i-m, \epsilon) \cdot \sum_{k=m+1}^{M+N-2i+m-1} P(M, k, \epsilon) \right\} \tag{4.19}$$

Then, we sum the probabilities of these two parts to get the probability that the data set $\mathbb{A}$ has fewer minority bits than $\mathbb{B}$ when the original phase value is the same as payload value, defined as $P\{\alpha < \beta | B = 1, P^{(0)} = U^{(0)}\}$ in Eq. 4.20.

$$\begin{aligned} P\{\alpha < \beta | B = 1, P^{(0)} = U^{(0)}\} = {} & P\{(\alpha < \beta) \cap (0 \leq \alpha \leq M) | B = 1, P^{(0)} = U^{(0)}\} \\ & + P\{(\alpha < \beta) \cap (M < \alpha \leq \mathcal{K}) | B = 1, P^{(0)} = U^{(0)}\} \end{aligned} \tag{4.20}$$

Under the assumption that the original payload value is different from the phase value, i.e $P^{(0)} \neq U^{(0)}$, the calculation of the probability that the data set $\mathbb{A}$ has fewer minority bits than the data set $\mathbb{B}$ is very similar to the assumption of $P^{(0)} = U^{(0)}$, except that we need to count the number of bits that retain their original value in the phase data in the data set $\mathbb{B}$

as the minority bits. The probability that the data set $\mathbb{A}$ has fewer minority bits than set $\mathbb{B}$ is calculated in Eqs. 4.21 and 4.22 for Parts 1 and 2, respectively.

$$P\{(\alpha < \beta) \cap (0 \leq \alpha \leq M) | B = 1, P^{(0)} \neq U^{(0)}\} = \\ \sum_{i=0}^{M-1} \left\{ \sum_{m=0}^{i} P(M, m, \epsilon) \cdot P(N, i-m, \epsilon) \cdot \sum_{k=2i-m-N+1}^{M-m-1} P(M, k, \epsilon) \right\} \tag{4.21}$$

$$P\{(\alpha < \beta) \cap (M < \alpha \leq \mathcal{K}) | B = 1, P^{(0)} \neq U^{(0)}\} = \\ \sum_{i=M}^{\lfloor \frac{M+N}{2} - 1 \rfloor} \left\{ \sum_{m=0}^{M} P(M, m, \epsilon) \cdot P(N, i-m, \epsilon) \cdot \sum_{k=2i-m-N+1}^{M-m-1} P(M, k, \epsilon) \right\} \tag{4.22}$$

We sum the probabilities of these two parts to get the probability that the data set $\mathbb{A}$ has fewer minority bits than set $\mathbb{B}$, defined as $P\{\alpha < \beta | B = 1, P^{(0)} \neq U^{(0)}\}$ in Eq. 4.23.

$$\begin{aligned} &P\{\alpha < \beta | B = 1, P^{(0)} \neq U^{(0)}\} \\ &= P\{(\alpha < \beta) \cap (0 \leq \alpha \leq M) | B = 1, P^{(0)} \neq U^{(0)}\} \\ &+ P\{(\alpha < \beta) \cap (M < \alpha \leq \mathcal{K}) | B = 1, P^{(0)} \neq U^{(0)}\} \end{aligned} \tag{4.23}$$

Then, the total probability that the data set $\mathbb{A}$ has fewer minority bits than the data set $\mathbb{B}$ is calculated in Eq. 4.24 based on the total probability law.

$$\begin{aligned} P\{\alpha < \beta | B = 1\} = &P\{\alpha < \beta | B = 1, P^{(0)} = U^{(0)}\} \cdot P\{P^{(0)} = U^{(0)}\} \\ &+ P\{\alpha < \beta | B = 1, P^{(0)} \neq U^{(0)}\} \cdot P\{P^{(0)} \neq U^{(0)}\} \end{aligned} \tag{4.24}$$

Extend the number of payload bits from $B = 1$ to $B > 1$

When the payload length B is greater than one bit, the confidence can be calculated in Eq. 4.11. The confidence is the sum of Δ^j, $j \in (0, B-1)$. So the probability that the confidence is greater in the data set $\mathbb{A}$ than in the data set $\mathbb{B}$ is the same as the probability that the sum of Δ^j in the data set $\mathbb{A}$ is smaller than in the data set $\mathbb{B}$ in Eq. 4.25, where

$\mathcal{C}_1$ is the confidence value for pure payload set, and $\mathcal{C}_2$ is the confidence value for mixture of payload and phase set.

$$P\{\mathcal{C}_1 > \mathcal{C}_2\} = P\left\{\sum_{j=0}^{B-1} \Delta_1^j < \sum_{j=0}^{B-1} \Delta_2^j\right\} \tag{4.25}$$

Note that we already discussed that for bit payload length $B = 1$, if we assume that fewer than half of the bits switched value for a given number of bit repeats R, the confidence is just the proportion of how many bits switched value over the total number of bit repeats. Now the payload is more than one bit in length, and we assume that fewer than half of the bits switched value for each bit position. Thus, the confidence, calculated as the sum of Δ^j, $j \in (0, B-1)$ for the data set $\mathbb{A}$ and the data set $\mathbb{B}$, is the sum of the number of bits that switched value among the total number of bit positions.

In Eq. 4.24, we already developed a closed form solution for each bit position assuming that the pure payload set has fewer minority bits α than the mixture of payload and phase, which has β minority bits. It is a function of the common payload bit repeat count N, the phase bit repeat count M in the mixture of payload and phase data sets, and the bit error probability ϵ. So for $B = 1$ we can rewrite Eq. 4.24 as Eq. 4.26:

$$P\{\alpha < \beta | B = 1\} \equiv \Psi(\alpha < \beta; N, M, \epsilon) \tag{4.26}$$

For $B > 1$, with the assumption that fewer than half of the bits switched value in each bit position index, the number of minority bits in each bit position index is the same as the number of bits that switched value in that bit position index. In addition, from our experiments to be discussed in Sec. 4.6, we found that when the payload bit length is large (i.e. $B > 63$), the probability that the payload value is the same as the phase value at each bit position index is about 0.5. Thus, the sum of the minority bits for the whole payload bit length B can be separated into two parts: one for those bit position indices where the phase bit has the same original value as the payload, and the other for those bit position indices where the phase original value is different from the payload.

Since the random variables that determine whether or not the bits switch value are i.i.d., at each bit repeat, the minority bit difference mainly comes from the second part: those bits where the payload and phase have different original values.

So, we can rewrite the probability $P(\mathcal{C}_1 > \mathcal{C}_2)$ using the formula developed for payload bit length $B = 1$, as shown in Eq. 4.26, but with the new variables in Eq. 4.27. This approximation is validated using simulation.

$$P\{\mathcal{C}_1 > \mathcal{C}_2\} = P\{\alpha < \beta\} \approx \Psi(\alpha < \beta; \frac{N}{2}B, \frac{M}{2}B, \epsilon) \tag{4.27}$$

4.3.2 Step 2: Compute the conditional probability of successfully decoding the payload

Now assume that we already correctly separated the phase from the payload; so for each bit position, we have $N + M$ bit repeats for data sets $\mathbb{A}$ and $\mathbb{B}$. We will compute the probability of successfully decoding the payload conditioned on the event $\mathcal{C}_1 > \mathcal{C}_2$.

The conditional probability that the detected bit value $\hat{\mathbf{P}}^{(j)}$ is the same as the original bit value $\mathbf{P}^{(j)}$ is equal to the conditional probability that fewer than half of the bits switched value. It can we calculated as:

$$\begin{aligned} P\{\hat{\mathbf{P}}^{(j)} &= \mathbf{P}^{(j)} | \mathcal{C}_1 > \mathcal{C}_2\} \\ &= \sum_{k=0}^{\mathcal{K}} \begin{pmatrix} (N+M) \\ k \end{pmatrix} (\epsilon)^k (1-\epsilon)^{(N+M)-k} \end{aligned} \tag{4.28}$$

where $\mathcal{K}$ is half of the number of bit repeats for the payload:

$$\mathcal{K} = \left\lfloor \frac{(N+M)}{2} - 1 \right\rfloor \tag{4.29}$$

The conditional probability that the entire payload is correctly decoded is the joint conditional probability that every bit in the payload is correctly decoded. Recall that we model the transmission error as identical and independent at each bit position, so the joint

probability of successfully decoding the entire payload is just the product of the probabilities of successfully decoding at each bit position.

$$
\begin{aligned}
&P\{\hat{\mathbf{P}} = \mathbf{P} | \mathcal{C}_1 > \mathcal{C}_2\} \\
&= \prod_{j=0}^{B-1} P\{\hat{\mathbf{P}}^{(j)} = \mathbf{P}^{(j)} | \mathcal{C}_1 > \mathcal{C}_2\} \\
&= \left\{ \sum_{k=0}^{\mathcal{K}} \begin{pmatrix} (N+M) \\ k \end{pmatrix} (\epsilon)^k (1-\epsilon)^{(N+M)-k} \right\}^B
\end{aligned}
\tag{4.30}
$$

4.3.3 Step 3: Compute the conditional probability of successfully decoding the phase

Note that for the phase encoding, we will encode the minimum number of bit shifts to go from the standard form S to the payload P. This number is denoted as C. We transfer the decimal value C to a binary string, denoted as $\dot{\mathbf{U}}$. The maximum number of bits c needed to represent the decimal value C can be calculated in Eq. 4.31. The method we use to encode the phase row is to repeat the string $\dot{\mathbf{U}}$ until all the B bits are used to form the phase. So for a phase row $\mathbf{U}$ with bit length B, the actual bit-repeat count $\dot{M}$ can be calculated as shown in Eq. 4.31, since in each phase row, we repeat each bit in the binary representation of C approximately $\frac{B}{c}$ times.

$$
c = \lceil \log_2 B \rceil \tag{4.31}
$$

$$
\dot{M} \approx \frac{B}{c} M \tag{4.32}
$$

Let's denote the decoded circularly shifted phase as $\hat{\dot{\mathbf{U}}}$; so we have:

$$
\hat{\mathbf{U}} = [\hat{\mathbf{U}}^{(0)}, \hat{\mathbf{U}}^{(1)}, \ldots, \hat{\mathbf{U}}^{(c-1)}] \tag{4.33}
$$

$$
\hat{\dot{\mathbf{U}}} = [\hat{\dot{\mathbf{U}}}^{(0)}, \hat{\dot{\mathbf{U}}}^{(1)}, \ldots, \hat{\dot{\mathbf{U}}}^{(c-1)}] \tag{4.34}
$$

Similar to the requirement to correctly decode the payload, in order to decode the phase, we require that fewer than half of the phase bits change their value during the transmission. Note that here we assume that each phase bit has the same number M of repeat rows. Thus,

$$\dot{\mathcal{K}} = \left\lfloor \frac{\dot{M}}{2} - 1 \right\rfloor \tag{4.35}$$

$$\begin{aligned} P\{\hat{\dot{\mathbf{U}}} = \dot{\mathbf{U}} | \mathcal{C}_1 > \mathcal{C}_2\} &= \prod_{j=0}^{c-1} P\{\hat{\dot{\mathbf{U}}}^{(j)} = \dot{\mathbf{U}}^{(j)} | \mathcal{C}_1 > \mathcal{C}_2\} \\ &= \left\{ \sum_{k=0}^{\dot{\mathcal{K}}} \begin{pmatrix} M \\ k \end{pmatrix} (\epsilon)^k (1-\epsilon)^{M-k} \right\}^c \end{aligned} \tag{4.36}$$

4.3.4 Step 4: Compute the final decoding rate

Without correctly separating the payload and phase data array, the chance to correctly decode the payload and phase bits is very low. Thus, we can approximate the probability of successfully decoding the payload as the product of the conditional probability of successfully decoding the payload and phase given that the payload and phase data are successfully separated, and the probability of successfully separating the payload and phase data.

Thus, the final decoding rate can be computed as the product of Eqs. 4.25, 4.30, and 4.36:

$$\begin{aligned} &P\{\hat{\mathbf{P}} = \mathbf{P}\} = P\{\hat{\mathbf{P}} = \mathbf{P}\} \cdot P\{\hat{\dot{\mathbf{U}}} = \dot{\mathbf{U}}\} \\ &= P\{\hat{\mathbf{P}} = \mathbf{P} | \mathcal{C}_1 > \mathcal{C}_2\} \cdot P\{\hat{\dot{\mathbf{U}}} = \dot{\mathbf{U}} | \mathcal{C}_1 > \mathcal{C}_2\} \cdot P\{\mathcal{C}_1 > \mathcal{C}_2\} \end{aligned} \tag{4.37}$$

4.4 Validate the Closed Form Solution

The closed form solution of successful decoding the payload probability in Eq. 4.37 is validated with experimental results.

4.4.1 Design of the simulation process

When the encoded halftone image is printed and captured by some image capturing device, and then the symbols (0 or 1) in the halftone sub-cell are decoded, there might be errors involved. We need to study the probability of successfully decoding with a predetermined transmission error. We model the the transmission error (switch value) as an i.i.d. random variable at each bit position. Here is the design of the process to simulate the probability of successfully decoding according to the following steps:

1. Randomly generate one sequence of payload data with a given payload length as in Sec. 2.3.1.

2. Encode the data array as in Sec. 2.3.

3. Mask half the data in a checkerboard pattern to simulate information embedding in a halftone image. See Sec. 2.5.

4. Generate an i.i.d. sequence of error values.

5. Crop the data array at every possible position within the Canonical Crop Window Location Set (CCWLS), so it will include every possible number of repeats for each bit position. See Sec. 3.2.

6. Decode the payload, including separating the payload and phase, decoding the standard form of the payload and decoding the phase. See Sec. 2.4. Then the final payload will be recovered.

7. Calculate the average decoding success rate. See Sec. 2.7.

Multiple choices of random errors at different percentages are simulated to calculate the statistical decoding rate. See the flowchart in Fig. 4.4.

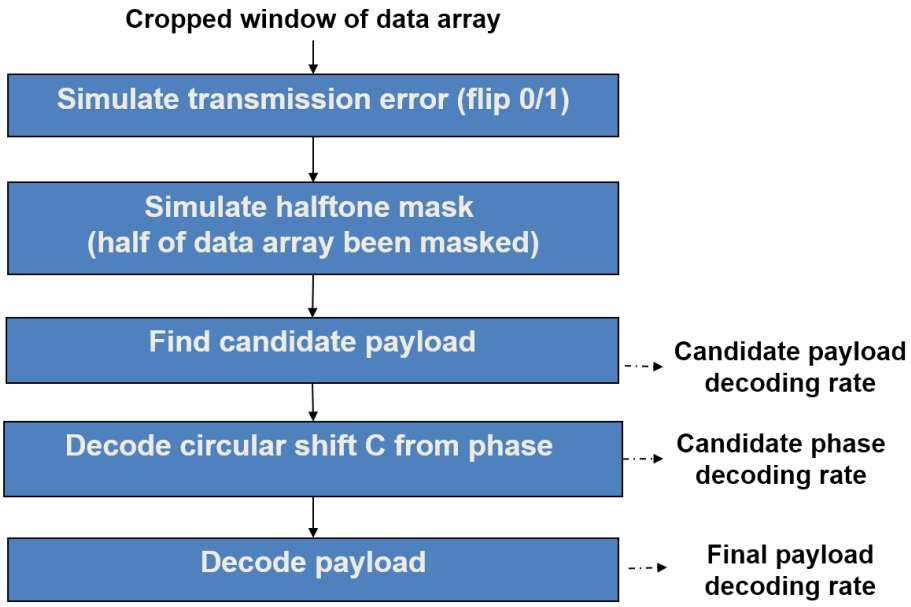


Figure 4.4. *Flowchart to simulate the payload decoding rate.*

4.4.2 Validate simulation results with theoretical results

The number of bit repeats for each position might be different due to the crop window size and location. To simplify the calculation, we set the crop window height H to be an integer, which is a multiple of the interleaving phase period V. Thus, among the H rows of data in the crop window, there are H/V rows of phase, and $H \cdot (V-1)/V$ rows of payload. In addition, we assume that the number of columns W is also an integer multiple of the payload length B. So for each row in the crop window, there will be the same number of occurrences for each value of the bit position index j.

We start from the simplest case, $B = 1$, and then extend it to many bits, i.e. $B > 1$. The transmission error ϵ is randomly generated, and sampled from 1% to 50%, with a step size of 1%. The validation is done for each major equation, including:

- Probability of separating payload and phase data set in Eq. 4.27 (refer to Figs. 4.5, 4.6, 4.7). Here N is the number of payload bit repeats, M is the number of phase bit repeats. See Sec. 4.3.1. The simulated curve closely matches the theoretical curve, which proves the closed-form formula is well approximated.

- Conditional probability of decoding the payload in Eq. 4.30. This part was validated by Sun *et al.* [59].
- Conditional probability of decoding the phase in Eq. 4.36. The closed form solution and validation process is very similar to that for Eq. 4.30.
- The final probability of decoding the original payload in Eq. 4.37. The validation of the simulation result with theoretical solution is shown in Fig. 4.9. Here it also shows that the theoretical and simulated results are well matched.

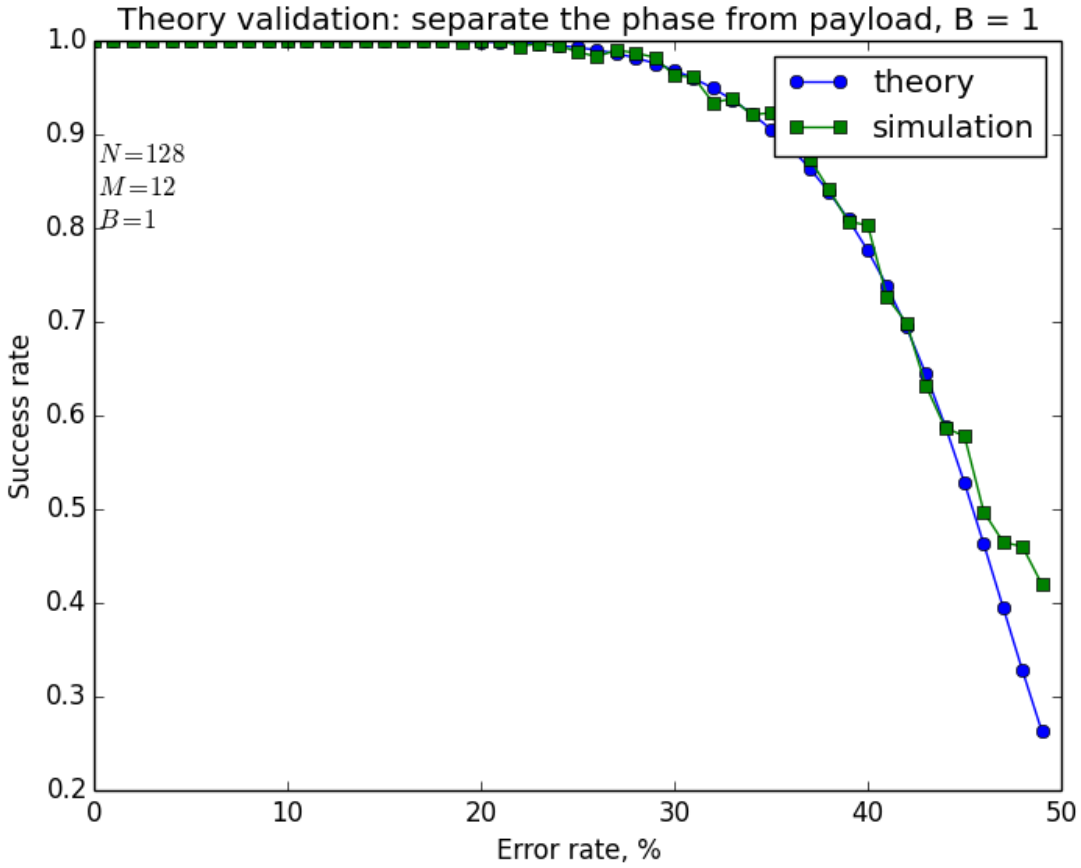


Figure 4.5. *Comparison of probability of successfully separating payload and phase bits based on theory and simulation for $N = 128$, $M = 12$, $B = 1$. Here, the phase period V, row to row shift D, crop window size W and H are not relevant.*

Note that for the simulation, when the confidence is exactly 50%, the decoder will select the first phase row. But in the formula, we require that fewer than half of the repeating bits

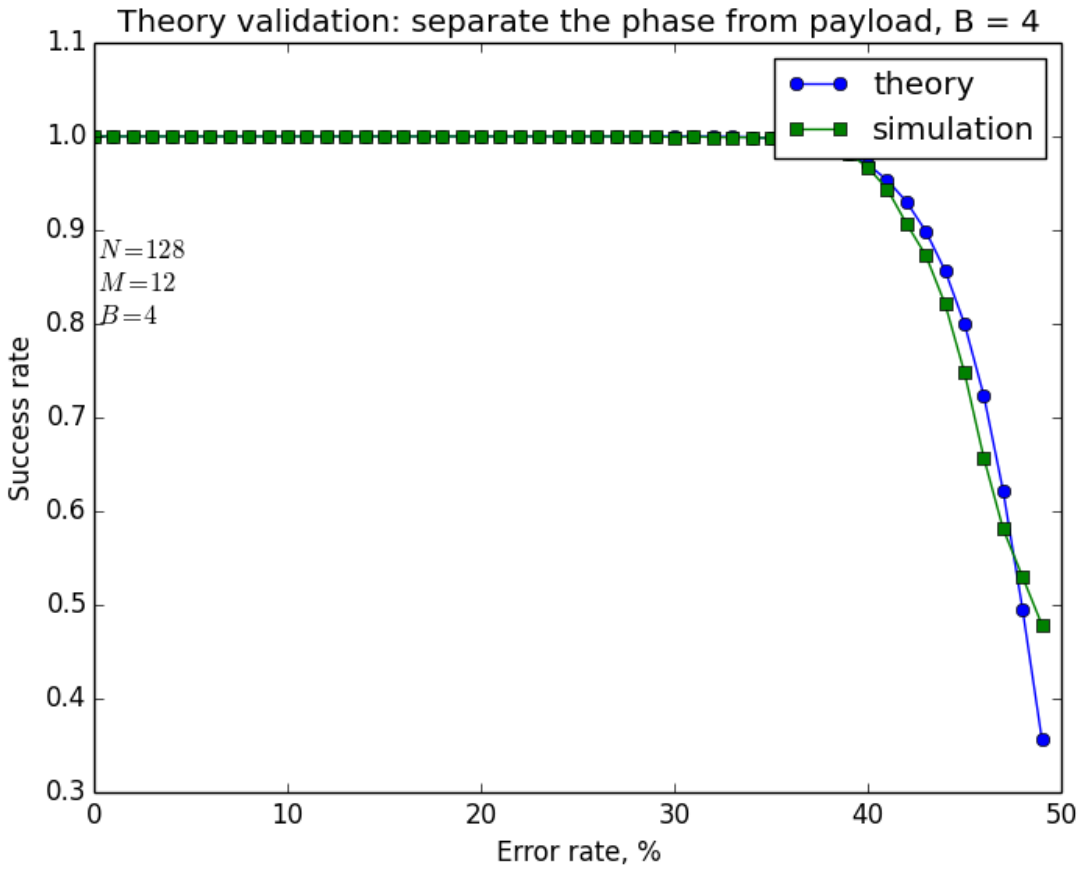


Figure 4.6. *Comparison of probability of successfully separating payload and phase bits based on theory and simulation for $N = 128$, $M = 12$, $B = 4$. Here, the phase period V, row to row shift D, crop window size W and H are not relevant.*

switch value. To accommodate this situation, we take an average of the floor and ceiling operations of the bits that switched value for which the confidence is 50%. The simulation result is shown in Fig. 4.8.

The Euclidean distance between the simulated and theoretical results will decrease when the number of simulation trials increases (10k $\rightarrow$ 40k $\rightarrow$ 100k), which is shown in Fig. 4.10. In other words, the simulation approaches the theory asymptotically.

When the bit length becomes very large ($B \geq 67$), the probability P_1 of separating the payload and phase becomes very high compared to the probability P_2 and P_3 of decoding

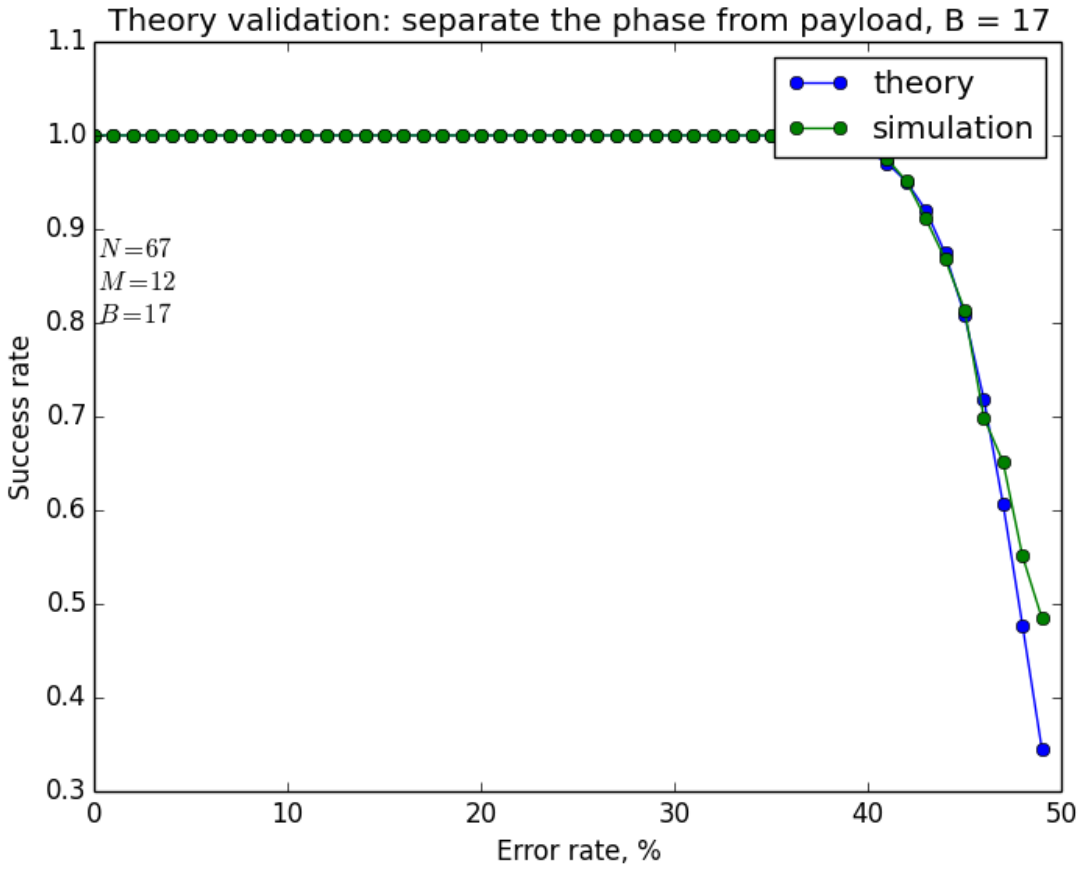


Figure 4.7. *Comparison of probability of successfully separating payload and phase bits based on theory and simulation for $N = 67$, $M = 12$, $B = 17$. Here the phase period V, row to row shift D, crop window size W and H are not relevant.*

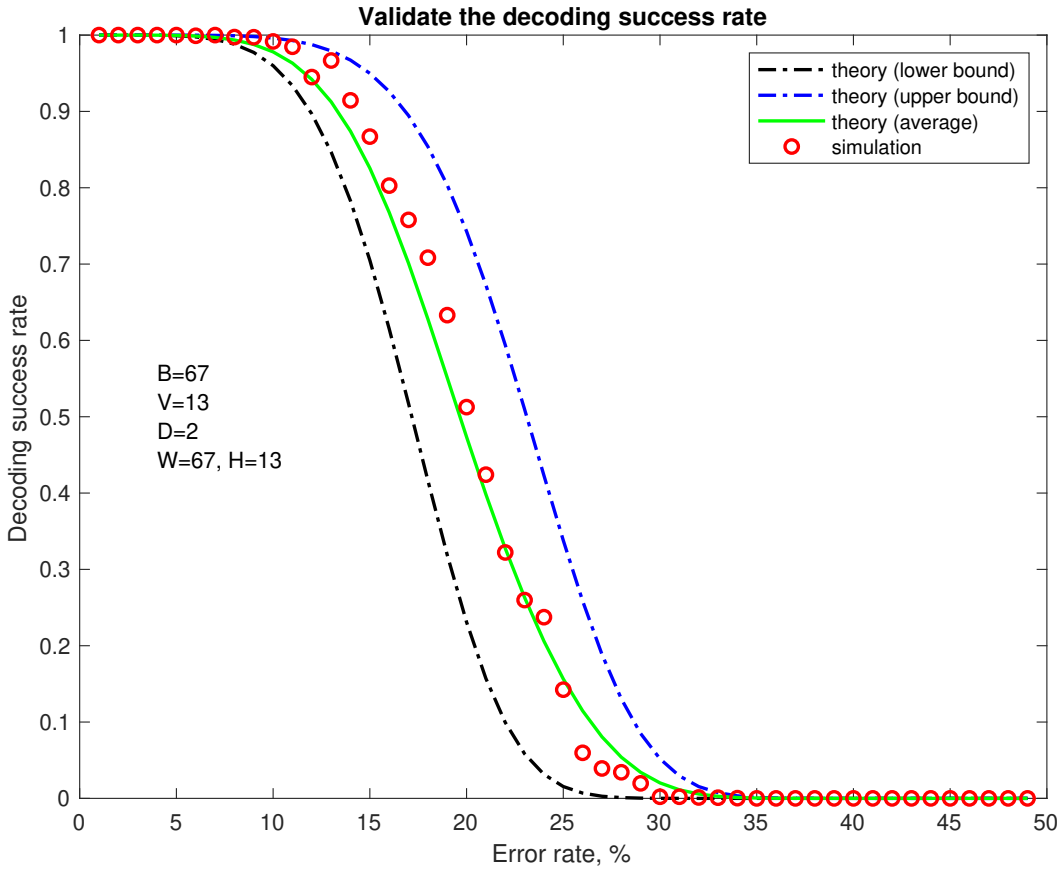


Figure 4.8. *Validation of the theory by simulation: the final decoding rate. The simulated decoding success rate is the average of 40k different samples of error at each transmission error rate.* $B = 67, W = 13, H = 13, V = 13$. *The lower bound and upper bound results are achieved when the confidence is exactly 50%. We use the floor and ceiling, respectively, of the bits that need to switch value to determine a successful decoding result. And the average decoding rate is the average of the lower bound and upper bound results.*

the payload and the phase, respectively. So we can use $P_2P_3 \approx P_1P_2P_3$ to simplify the computation (refer to Fig. 4.11).

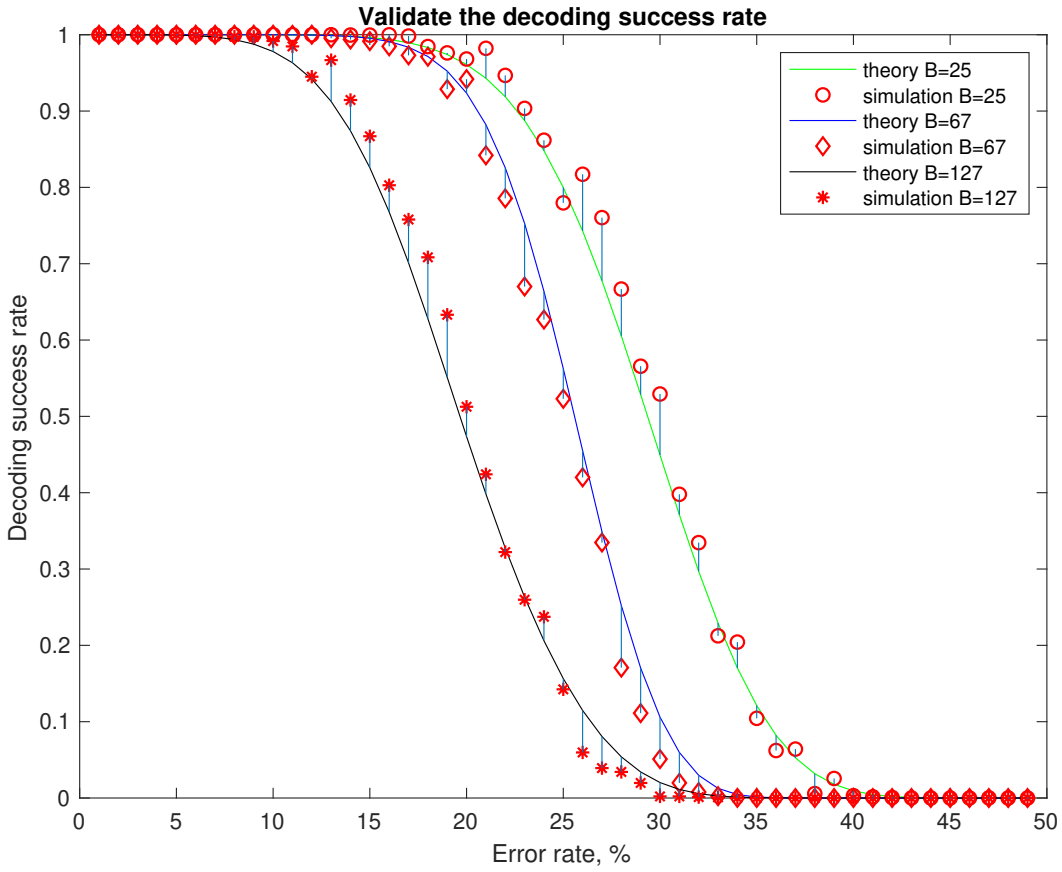


Figure 4.9. *Validation of the theory by simulation: the final decoding rate. The simulated decoding success rate is the average of 40k different samples of error at each transmission error rate. For the first comparison group (the green curve),* $B = 25$, $N = 15$, $M = 4$, *and* $V = 5, D = 2, W = 25, H = 25$*; for the second comparison group (the blue curve),* $B = 67$, $N = 14$, *and* $M = 2$; $V = 13, D = 2, W = 13, H = 13$ *for the last comparison group (the black curve),* $B = 127$, $N = 22$, $M = 1$, *and* $V = 24, D = 2, W = 127, H = 24$.

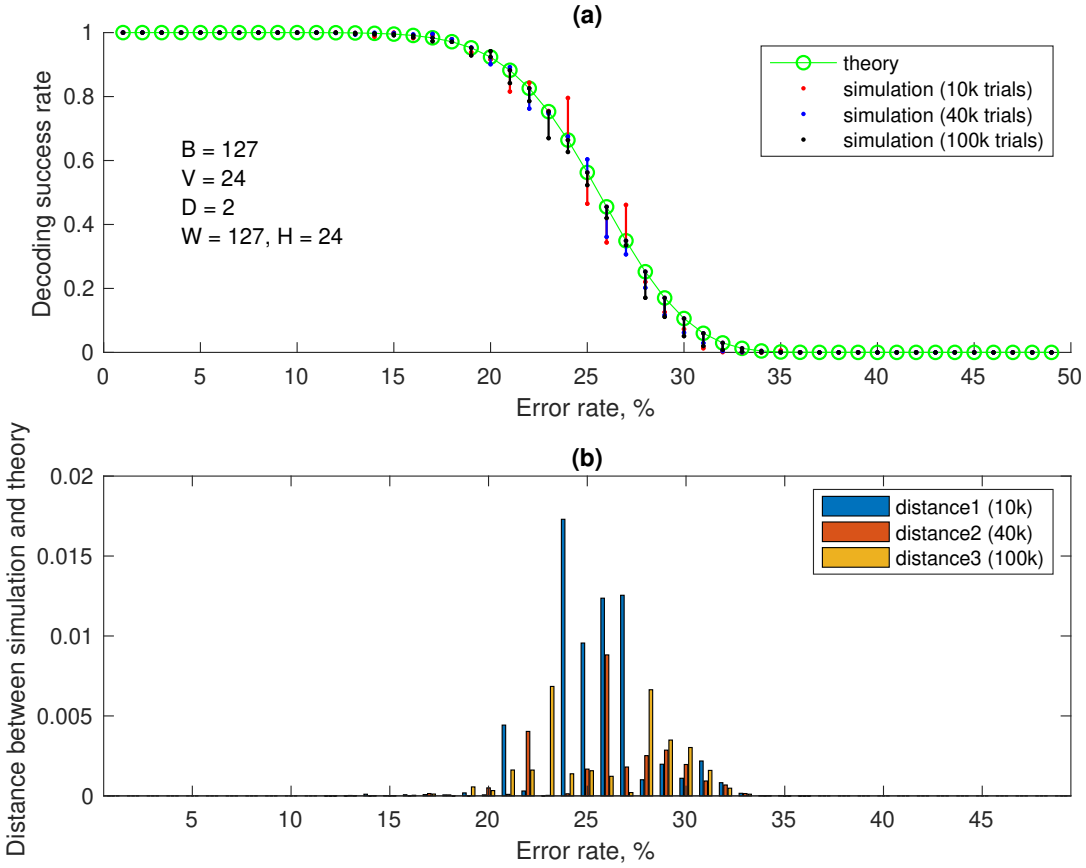


Figure 4.10. *Effect of increasing the number of simulation trials on the match between the theoretical ad simulation results. (a) Decoding success rate as a function of error rate. (b) The Euclidean distance between the simulated and theoretical results.*

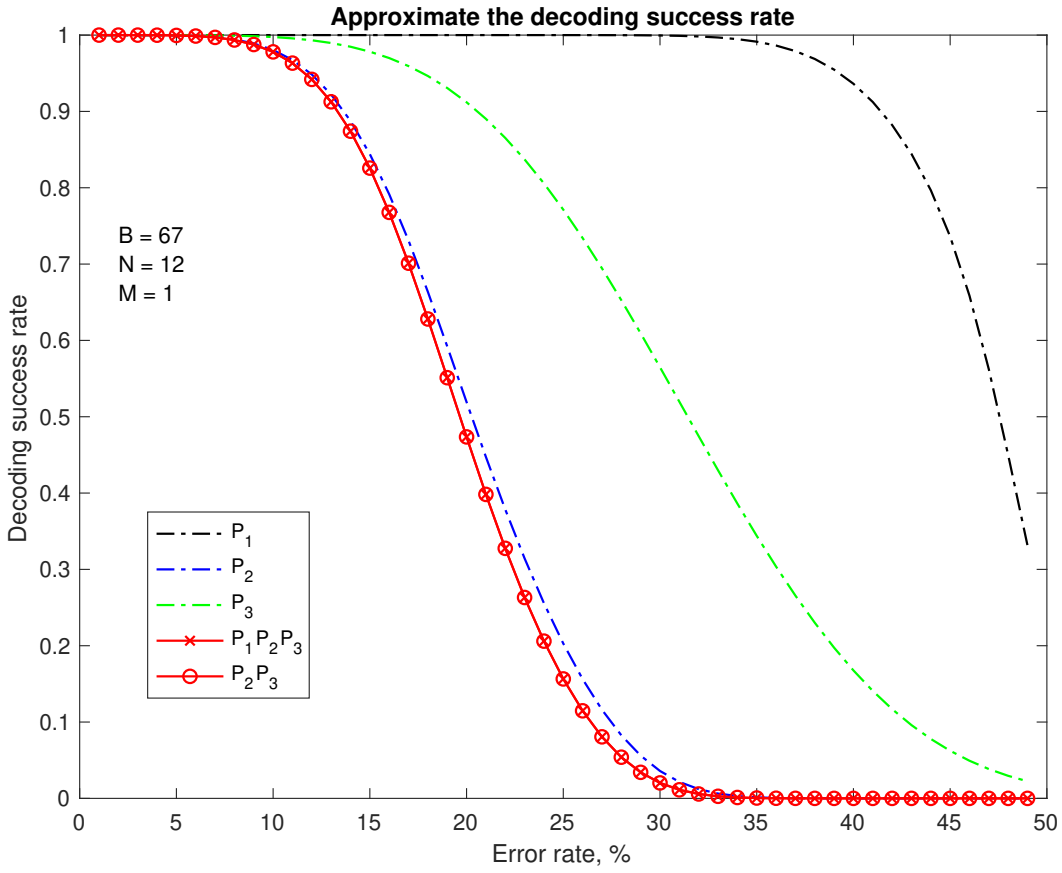


Figure 4.11. *The approximation of the final decoding rate. P_1: the probability of separating payload and phase defined in Eq. 4.27; P_2: the conditional probability of decoding the payload in Eq. 4.30; P_3: the conditional probability of decoding the phase in Eq. 4.36; $P_1P_2P_3$: the final probability of decoding the original payload in Eq. 4.37. P_2P_3: the approximation of the final probability of decoding the original payload in Eq. 4.37. The simulated decoding success rate is the average of 40k different samples of the error at each transmission error rate.*

4.5 Summary of the Assumptions

Without loss of generality, we have made some assumptions for easier derivation of the closed form solutions in this chapter:

- During the design of the payload, we will check make sure that the standard form of the payload is unique by circularly shifting the bits. And the length of the payload is always an odd number.

- When halftoning the carrier image for embedding bit shifts, in order to make the halftone patterns be limited to the patterns shown in Fig. 2.9, we will first average the gray scale value within each 4×4 sub-cell.
- We assume that each bit position index has the same repeat number R. Because for a crop window of data $\hat{\mathbf{r}}$, we are able to calculate the bit repeat count for each bit position index. The actual bit repeat count for each bit position index might be slightly different, depending on the size and position of the crop window.
- For a successful detection, we assume that the probability of transmission error $\epsilon < 0.5$. Thus, the error rate is less than 50%.
- We assume that fewer than half of the bits switched their values. Since the number of payload bits is usually much larger than the number of phase bits, or $N \gg M$, for any bit position j, we expect that there will be fewer phase rows than payload rows.
- Without correctly separating the payload and phase data array, the chance to correctly decode the payload and phase bits is very low. Thus, we can approximate the probability of successfully decoding the payload as the product of the conditional probability of successfully decoding the payload and phase given that the payload and phase data are successfully separated, and the probability of successfully separating the payload and phase data.

4.6 Examine the Similarity of the Bit Values Between Payload and Phase

The bit value for each repeating bit position, if it is belongs to a payload row, can be denoted as $\boldsymbol{P}(j)$; and if this bit is in a phase row, then the bit value can be denoted as $\boldsymbol{U}(j)$.

The phase code depends on the payload and encoding method. We define the similarity as the number of bits between the payload and the phase that have the same value, divided by the total number of the payload (or the phase) bits.

$$\text{similarity} = \frac{\text{number of same bits between payload and phase}}{\text{total number of bits}} \tag{4.38}$$

4.6.1 Design the experiment to examine the similarity

The similarity between payload and phase depends on the payload value, and also depends on the method of how the phase is encoded. However, we can still examine the similarity using the some experiments. Here is how to set up the experiment:

1. Set payload bit length B.
2. Generate every possible form of payload. There will be totally 2^B different possible forms of the payload.
3. Generate the sequence of phase bits for each selected sequence of payload bits.
4. Calculate the similarity for each payload and phase pair based on Eq. 4.38.
5. Calculate the average similarity rate.

4.6.2 Experiment and result

We examined the similarity from payload bit length 7 to 512, and calculated their similarity. For the bits length smaller than 27, we examined every possible payload value; for bit length from 27 to 512, we sampled 10,000 different payload values and calculated the average similarity. The result is shown in Fig. 4.12.

So we can assume that the bit values of the payload and phase are independent. That is, the probability that the payload bit value is the same as phase bit value for a particular bit position index j is 0.5, and so is the probability that the payload bit value is different from the phase bit value.

$$\begin{aligned} P\{\boldsymbol{U}(j) = \boldsymbol{P}(j)\} &= 0.5; \\ P\{\boldsymbol{U}(j) \neq \boldsymbol{P}(j)\} &= 0.5; \end{aligned} \tag{4.39}$$

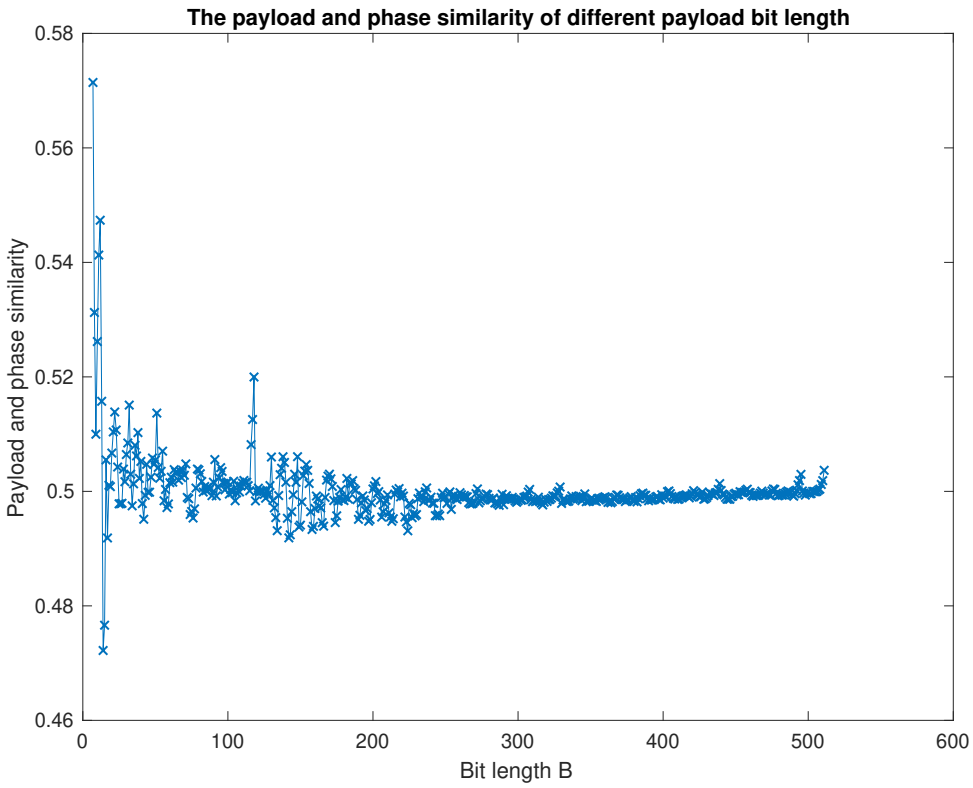


Figure 4.12. *The similarity of the payload and phase, using the double bit encoding method, see Sec. 2.3.1.*

4.7 Conclusion

There are various ways for information embedding. One of the popular approaches is through the bar code. However, the methods covered here are called information coding, which hide information under the carrier information and can only be read when the decoding key is provided. These kinds of methods add an extra layer of security and are not easy to notice unless someone scans the carrying signal with a key.

Conventional information coding methods, such as Hamming Code [37], Reed Solomon Code [38], Luby Code [46], Raptor Code [49] are error-prone, since they encode information in a single dimension and may fail to be corrected if there is a bit loss. This dissertation applies a 1D information method in a 2D image, which allows more error correction tolerance and enhances the robustness of information embedding.

Secondly, in the conventional 1D approach, bit positional synchronization is required, where the positional information must be retained for recovery. Whereas in this proposed 2D approach, this synchronization is no longer required, since the positional information is also included in the encoding process.

The information is repetitively embedded in every location in the image, except in the following areas: the shadow regions (too few hole clusters), highlight regions (too few dot clusters) and the mid-tone regions (no place to shift dot clusters), see Sec. 2.5.2. Therefore, to scan the information, the user can literally start from any place in the image.

Given a particular region of the image at a specific resolution, the confidence level can be theoretically computed, to suggest the user to either zoom in (increase the resolution on a smaller region) or zoom out (decrease the resolution but enlarge the scanning region) for a better confidence score. The developed formula has been experimentally proven to approximate all kinds of simulated scenarios with different parameters, such as bit error rate, information length, resolution in dpi, etc. It can be efficiently used to estimate the results for various simulations, without performing time-consuming experiments.

5. WEB-BASED PRINT QUALITY TROUBLESHOOTING (PQTS)

5.1 Problem Description

This project is aimed to develop web-based self-diagnostic Print Quality Troubleshooting (PQTS) tools, that can help HP Inc.'s customers not only to self-diagnose their printer's quality issues, but also to solve those problems. As these high-end printers are sold world-wide, customer service becomes important and necessary. In addition, since the print quality issues are difficult to describe verbally with the traditional customer phone service, it is very difficult to identify what is really the problem that the customer is facing, and it will be a time-consuming and costly process. The PQTS tool provides simulated print quality examples to help the customer identify their issue, and also follows with step-by-step instructions to solve most of the common problems. Based on the web page viewing history, we can see that this tool has been widely used.

5.2 Structure of the PQTS Tool

The PQTS tool was fist described in a journal paper by Santos [60], *et al.*, and our team members have continued to work on this project for many printer models [61]–[67]. This work built on earlier work with print defect simulation [68], and is only one example of a web-based troubleshooting tool. Another example is [69]. It uses a three-layer architecture diagnostic model that gives easy and practical problem-solving solutions:

- First layer: Problem Initialization. In this layer, the user prints out test pages, and then check the defects in the printed test pages on each color test page, and determines which color(s) need to be examined for further diagnostics.
- Second layer: Issue Identification. In this layer, the user observes more detailed quality issue simulations under each color category, and links the particular issues with potential causes by checking the simulated examples of defects.

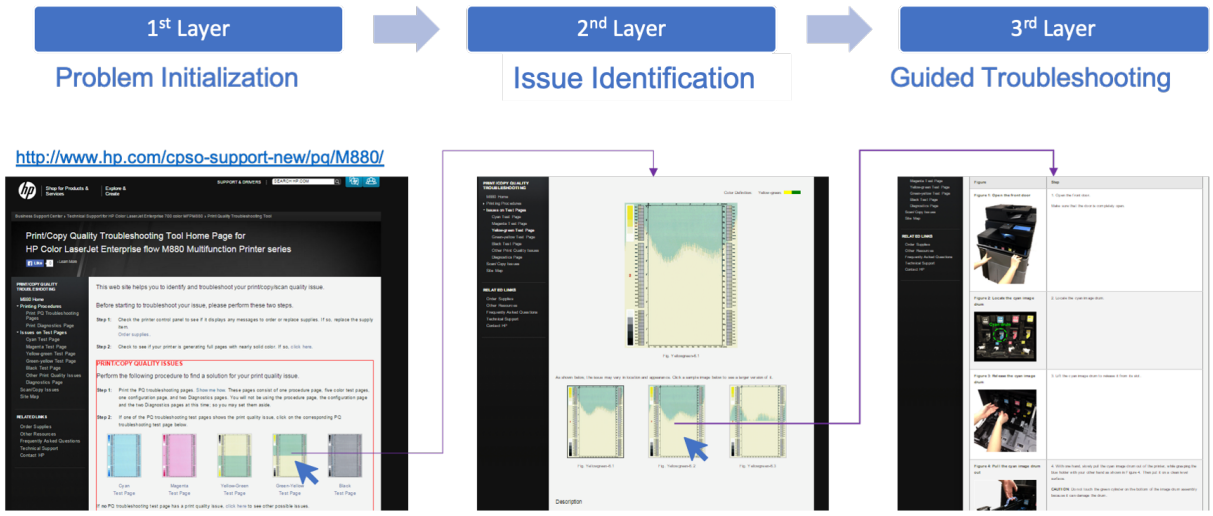


Figure 5.1. *The PQTS three-layer architecture illustration.*

- Third layer, Guided Troubleshooting. Here, a step-by-step solution is provided to solve these common issues.

For example, in the first layer, the customer may notice some abnormality on the green-yellow test page. So he or she clicks this test page and goes to the second layer. On the second layer, there are different simulations of possible defects. By comparing these simulation with what he or she has, he or she may identify the issue that the boundary between where the toner is present or not creates a wavy pattern. By clicking the most similar simulated detect page, the website will direct the customer to the third layer, where possible causes are listed. In this example, there are two possible causes:

1. Toner level is low in the cyan image drum;

2. There is a problem replenishing the toner from the cyan toner cartridge.

Also, the troubleshooting and solutions are provided to the user, including step-by-step instructions with hands-on examples (see Fig. 5.1).

Table 5.1. The print quality trouble shooting products that been developed.

Model	Release dates	PQTS tool URL
CP5525	1/17/2014	http://www.hp.com/go/printquality/cljcp5525
M775	1/15/2014	http://www.hp.com/go/printquality/m775
M551	10/21/2013	http://www.hp.com/go/printquality/lj500colorM551
M575	8/14/2013	http://www.hp.com/go/printquality/lj500colorMFPM575
CM4540	7/1/2013	http://www.hp.com/go/printquality/cljcm4540mfp

5.3 PQTS Tool Development

Our team had developed the PQTS tools for totally eighteen printers models before I joined the group. Since each printer has its own PQTS tool, for the newly released models, I reproduced and generalized the website content, including:

1. Write the trouble shooting procedures;
2. Simulate the print quality diagnostic pages;
3. Write the problem solving procedures;
4. Capture the step-by-step problem solving illustrations

During Summer 2013 to Spring 2014, I developed an additional five tools for the current models. Here is the list of each model, its release dates, and the URL for its Print Quality Troubleshooting Tool at HP's website, see Table 5.1.

5.4 Conclusion

Most of the print quality problems are very difficult to accurately describe verbally, especially for customers that lack printing quality knowledge training. The web-based self-diagnostic tool enables the customers to print out the test pages, and compare their printing quality with the most common issues, which are simulated and shown on the website. So the customer can diagnose the problem without contacting the customer service specialist, which saves both the time and cost. From Fig. 5.2, we can see that there were more than sixteen thousand views during Jun. 1, 2012 to Mar. 14, 2014.

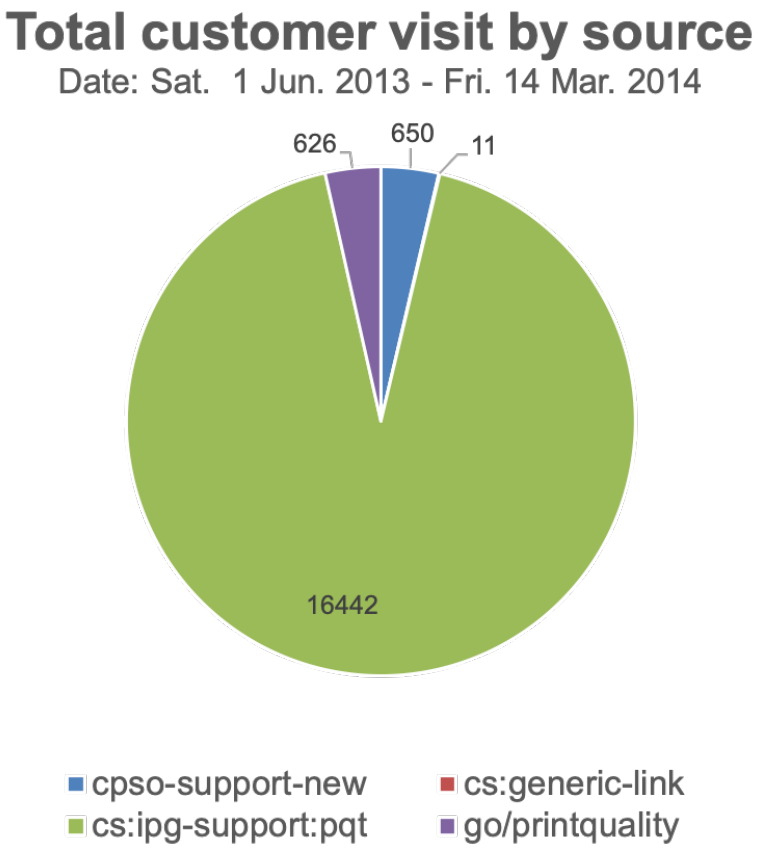


Figure 5.2. *The total page views for PQTS products during the time period of Jun. 1, 2013 to Mar. 14, 2014. There are totally four different websites that link to the PQTS tools: cpso-support-new, cs:generic-link, cs:ipg-support:pqts and go/printquality. The statistical data was provided by HP Inc.*

6. TEXT LINE DETECTION

6.1 Problem Description

Text line detection with low computational cost and high accuracy is a critical step for applications in document image processing. There are many methods have been proposed for text line detection. One category is the Hough transform-based text line detection method. [70]–[72] This method proposes the text lines as hypotheses in the Hough transform domain and validates them in the image domain. This "hypothesis-validation" strategy is very computationally expensive, and often pre-processing and post-processing is required for a higher detection accuracy.

Another category is smearing methods [73], [74]. Different from the Hough transform method which is a"top-to-down" approach, the smearing method is "bottom-up": it grows the text line region by recursively finding and incorporating the closest characters. So the smearing methods handle curved text lines better than the Hough transform based method, but since it searches only a small region, the smearing methods are more sensitive to noise, and they require accurate parameters. Some more text line detection methods can be found in [75]–[77].

In this project, a novel method has been developed by Dr. Yandong Guo [78]. First, we find the connected components of the image as symbols. Then, we find features of these symbols to classify them into different classes, such as text, table, logo, etc. Next, a cost function is designed to estimate the local text line direction and the relationship of character pairs within the local region, based on the observation that the text line is typically formed by a set of characters densely distributed along a smooth curve in a region. Once the text line in the local region is found, then a graphical model is built: we model each character as a node, and then group the characters into text lines by separating the graph into sub graphs based on the estimation result in the first step.

Compared with the Hough transform based method, our method is more computationally efficient. In addition, since we find the text line segment in a local region, it handles the curved text line better than the Hough transformed method. Compared with the smearing

method, our approach use a global optimization to group text line segments into text lines, which yields a better detection accuracy.

Experiments with a variety of images demonstrate that the proposed method is very fast and robust.

6.2 Text Line Detection Pipeline Review

The text line detection flowchart is shown in Fig. 6.1.

To better understand the whole pipeline, we use the following diagrams to explain how each step is executed. Fig. 6.2 is the main flowchart, and Figs. 6.3, 6.4, 6.5 , 6.6 are the sub figures to discuss the details.

6.3 Implementation of the Algorithm

The input and output of the text line decoding program are listed here:

- Input: Supported PBM (P4), PGM (P5) or PPM (P6) formats.
- Output: Gray-scale image in PGM (P5) format that only includes the detected text lines. It also includes some optional files that can be used for debugging purposes, including: Feature vectors including the meta and histogram files, debug images for detected text lines, bounding boxes, and other detected logos/tables/small dots in different colors.

The average Running time is 2.79 seconds per image (per page) based on total 261 images, see Fig. 6.7.

6.4 Conclusion

In this project, I continued working with the text line decoding algorithm that was first developed by Dr. Yandong Guo, developed the pipeline, changed the libraries to suit our sponsor HP's requirements, and ran more experiments to analyze the performance.

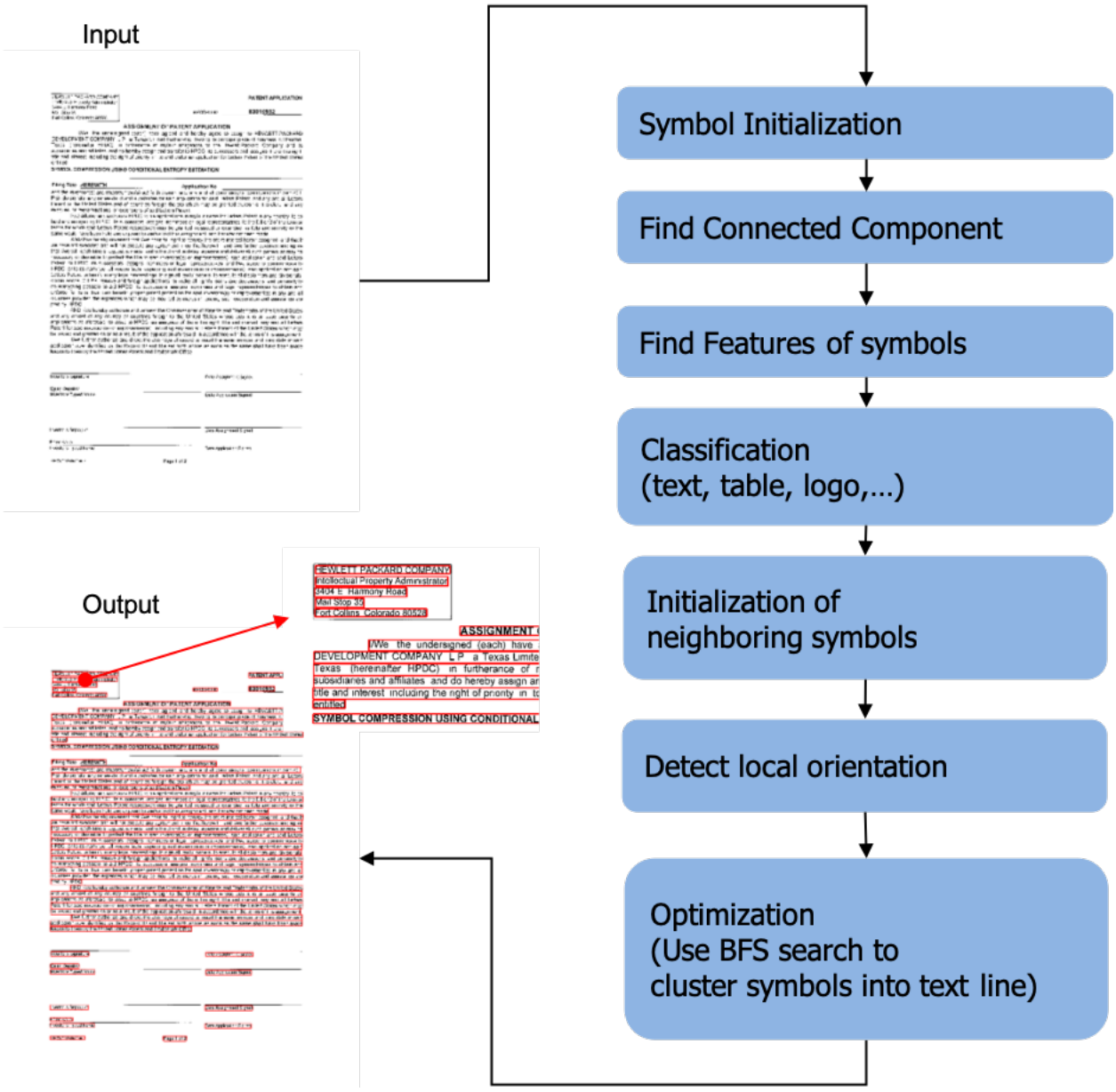


Figure 6.1. *The flowchart of the text line detection process*

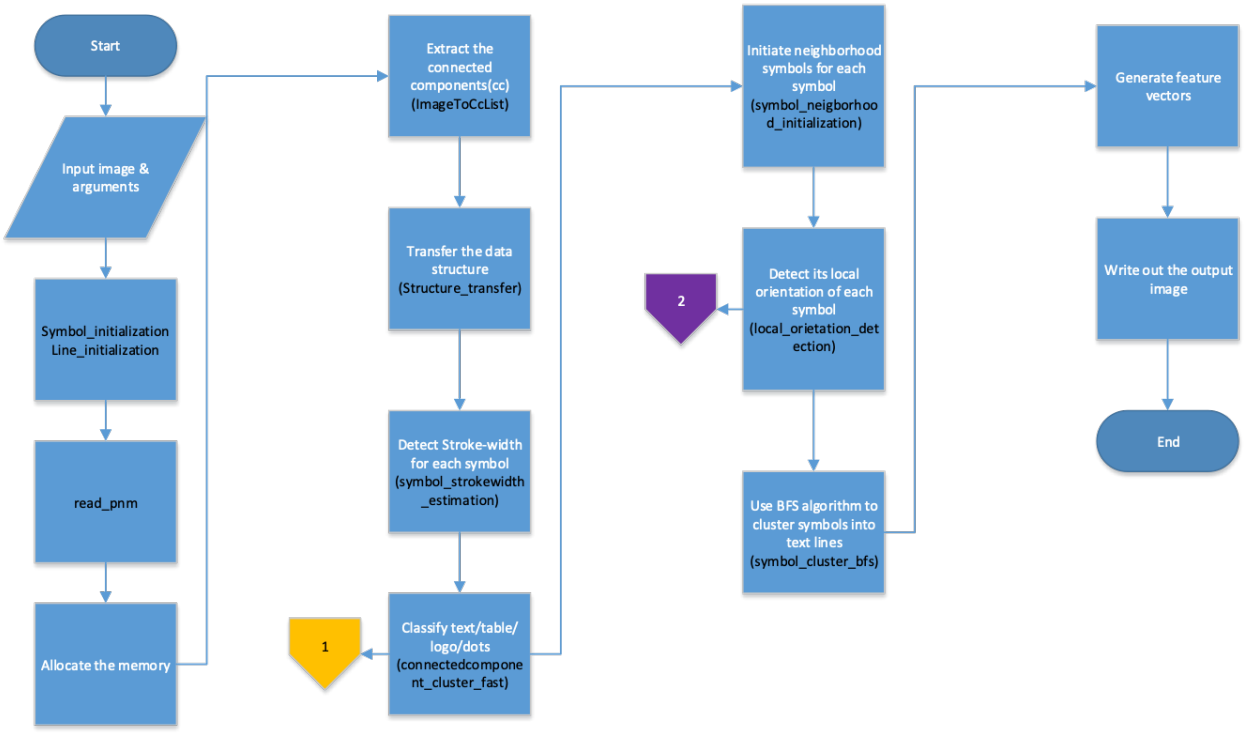


Figure 6.2. *The flow chart - main*

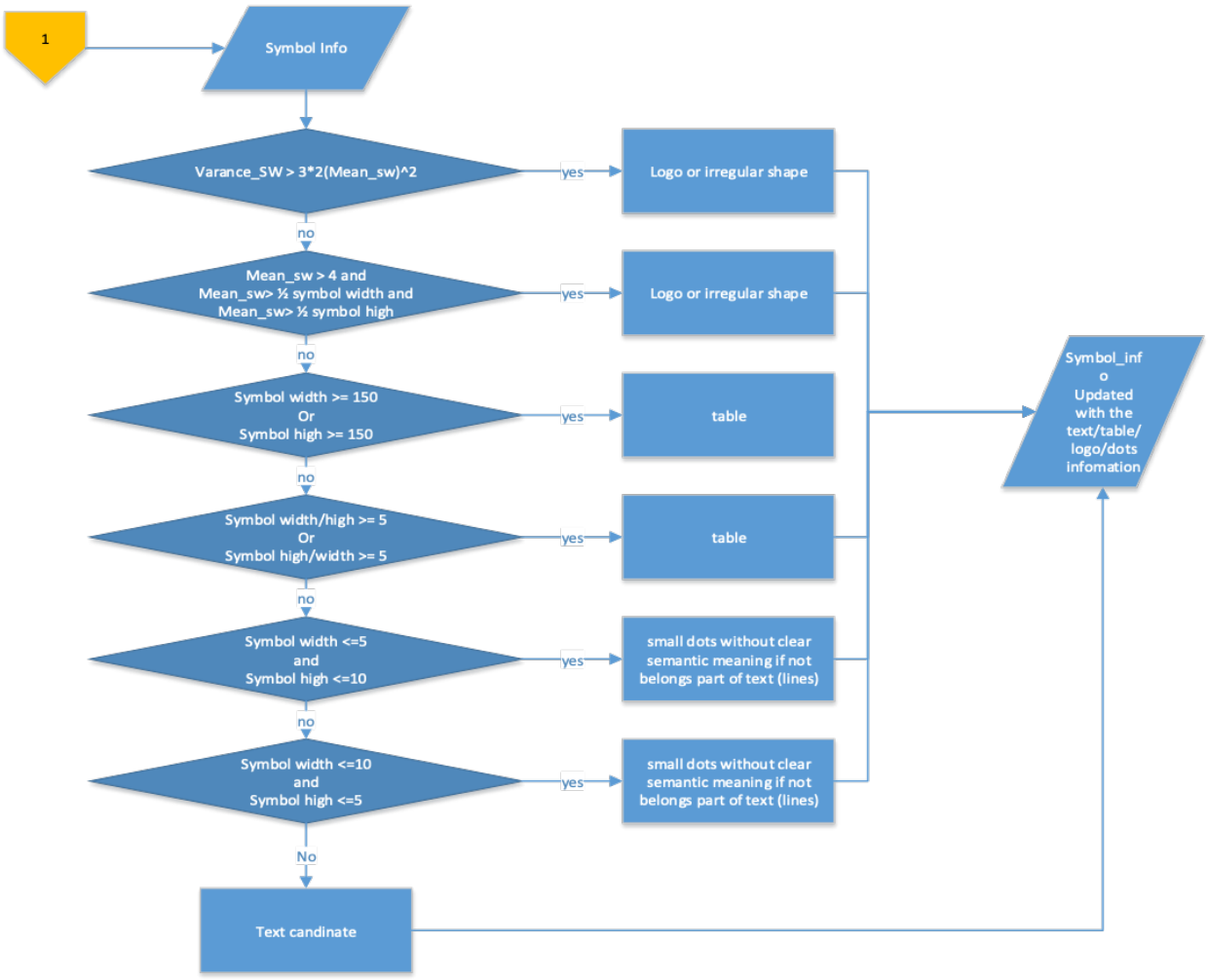


Figure 6.3. *The flow chart - reference 1*

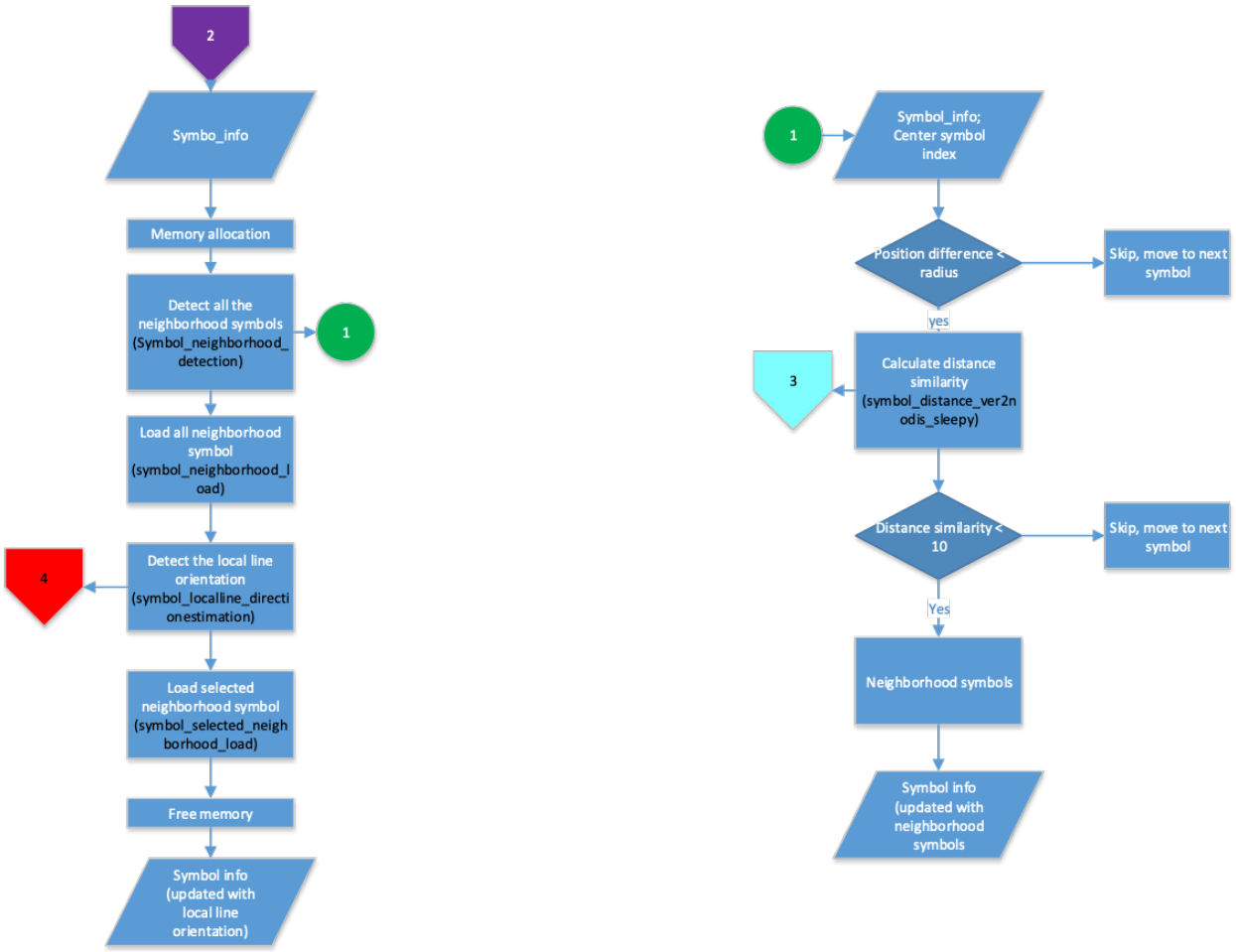


Figure 6.4. *The flow chart - reference 2*

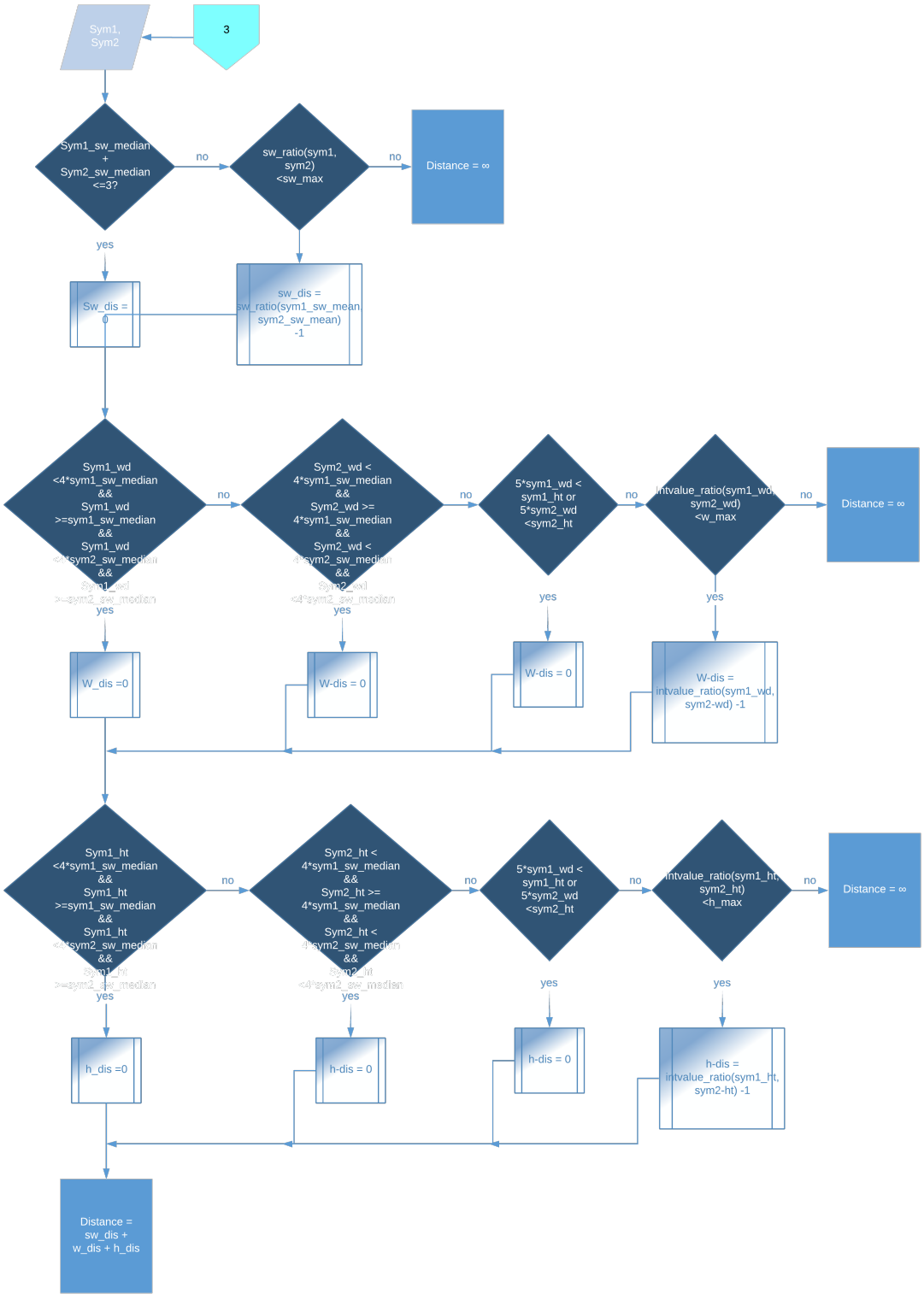


Figure 6.5. *The flow chart - reference 3*

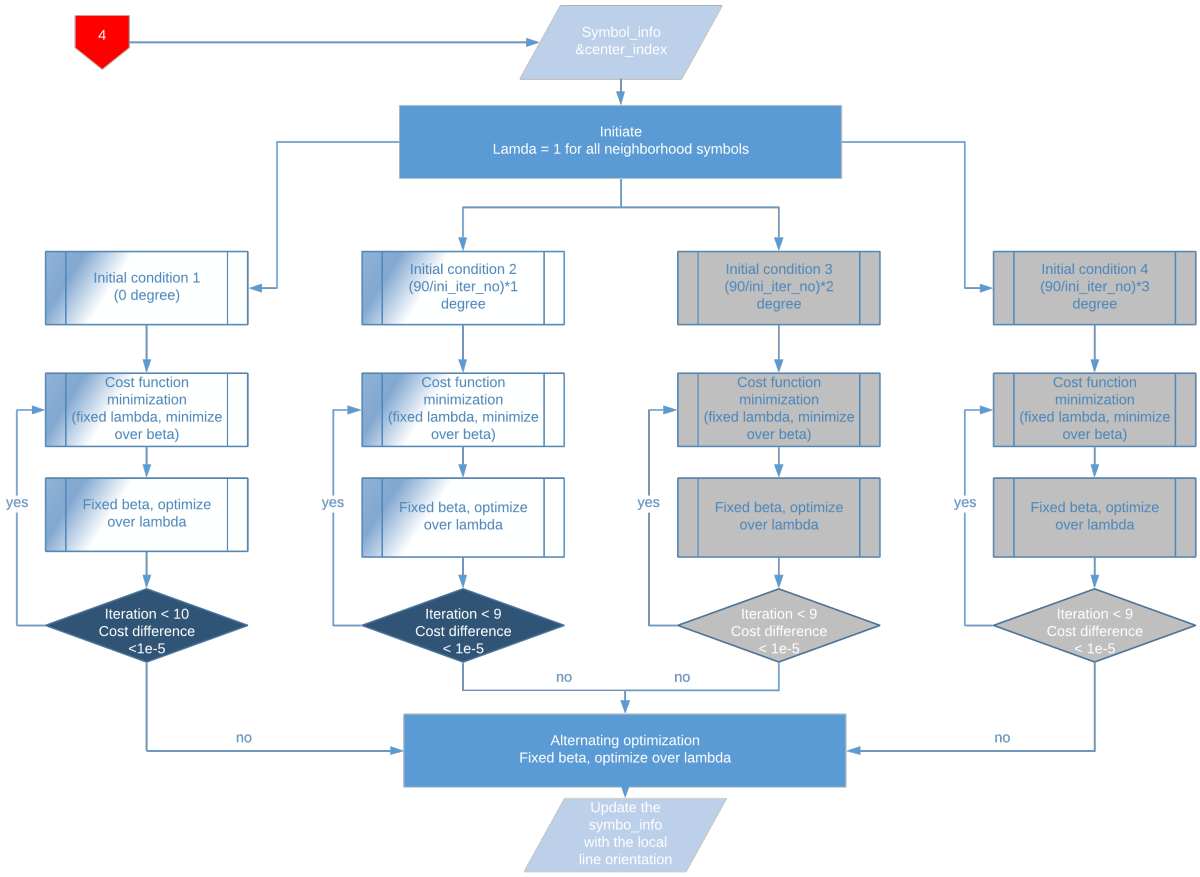


Figure 6.6. *The flow chart - reference 4*

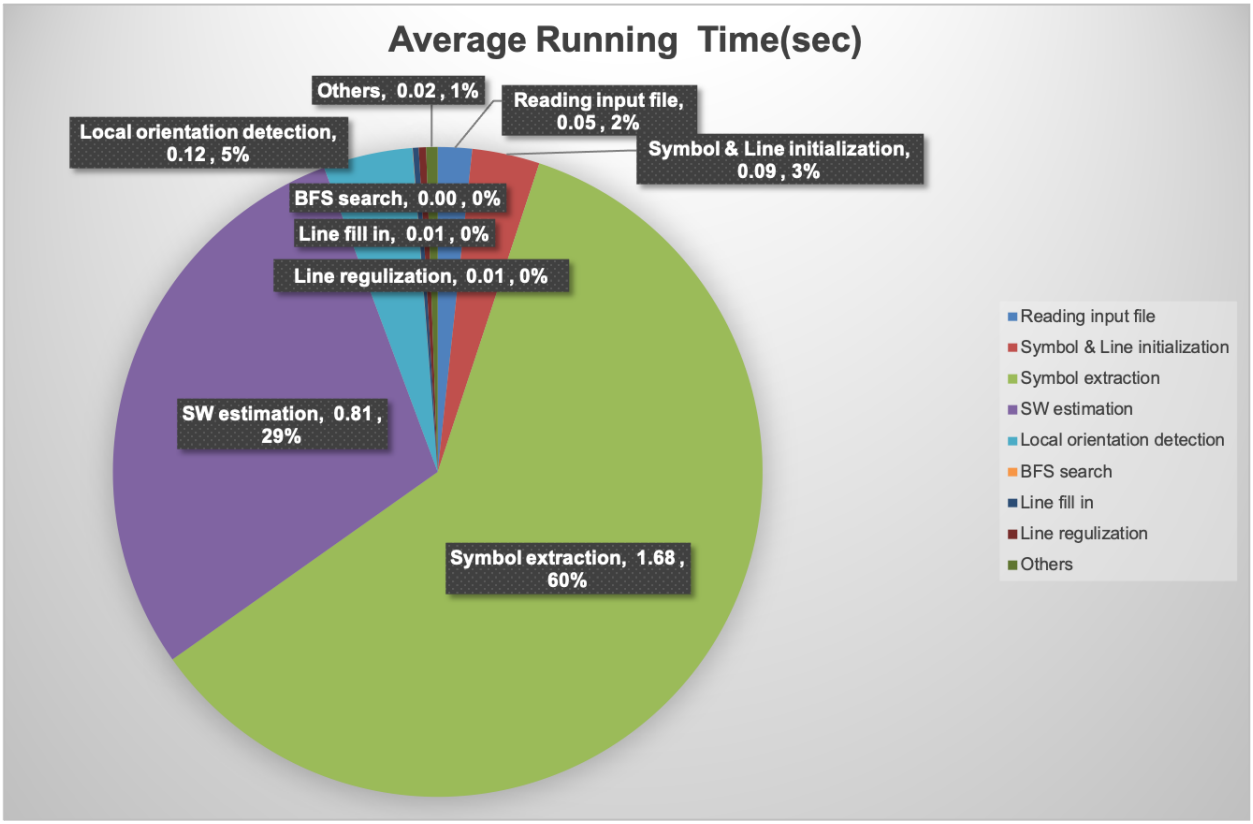


Figure 6.7. *Average Running time is 2.79 seconds per image (per page) based on total 261 images detection result, including (1) Mixed pictures and text; (2) Horizontal vertical lines; (3) Skewed text lines and (4) Different fonts, contents.*

7. SUMMARY

This dissertation mainly discussed the circular coding algorithm and its performance prediction, and also some other image processing projects that have been done during my Ph.D. journey, including the web-based printing quality troubleshooting tool project, and the text line detection project.

The literature review about channel coding and current data embedding methods is written in Chapter 1.

In Chapter 2 , We introduced the encoding and decoding systems and investigated the performance of the methods for noisy and distorted images.

In Chapter 3, we validated that the experimental payload decoding rates are consistent with their theoretical results, given particular parameters and with various cropped-window sizes. Therefore, given the required decoding rate and anticipated transmission error, we can compute the minimum requirement for the number of repeats or the corresponding cropped-window size.

In Chapter 4, we modeled the transmission error as a stochastic random process. Then we developed a closed-form solution for the payload decoding rate step by step, following the procedures of the decoding process.

Also, we designed the simulation of the decoding process to validate the closed-form solution.

The contributions to the circular coding project include the following:

- Developed a model for the circular coding encoding and decoding system.
- Developed a closed form formula to calculate the bit repeat count for the given crop window of data.
- Developed the Canonical Crop Window Location Set (CCWLS) to identify the minimum requirement of the bit repeat counts.
- Analyzed the performance of the circular coding method in a noisy channel. A closed-form solution was developed to calculate the decoding success rate for a given message payload length and bit position repeat count under different transmission error rates.

- Validated this closed-form solution by simulating the decoding process with noisy samples. With this decoding rate prediction, we can design the encoding/decoding system with the desired performance under different given transmission error rates. On the other hand, for a given encoding/decoding system, we will have the expected success rate as a measure of confidence for users.

In addition, some other image processing projects were discussed. The web-based Print Quality Troubleshooting Tool (PQTS) is discussed in Chapter 5. For this project, we reproduced and generalized the PQTS websites for newly released printers, and reduced the troubleshooting time for HP's customer supporting team, according to HP's statistical data.

The text line detection and its application is discussed in Chapter 6. In this chapter, the pipeline of the text line detection method was developed, we modified the code to suit the sponsor HP's requirements, and ran more experiments to analyze the performance.

REFERENCES

[1] P.-J. Chiang, N. Khanna, A. K. Mikkilineni, M. V. O. Segovia, S. Suh, J. P. Allebach, G. T.-C. Chiu, and E. J. Delp, "Printer and Scanner Forensics," *IEEE Signal Processing Magazine*, vol. 26, no. 2, pp. 72–83, 2009.

[2] M. V. O. Segovia, G. T.-C. Chiu, and J. P. Allebach, "Using Forms for Information Hiding and Coding in Electrophotographic Documents," in *Information Forensics and Security, 2009. WIFS 2009. First IEEE International Workshop on*, IEEE, 2009, pp. 136–140.

[3] P.-J. Chiang, N. Khanna, A. K. Mikkilineni, M. V. O. Segovia, J. P. Allebach, G. T. Chiu, and E. J. Delp, "Printer and Scanner Forensics: Models and Methods," in *Intelligent Multimedia Analysis for Security Applications*, Springer, 2010, pp. 145–187.

[4] G. N. Ali, A. K. Mikkilineni, J. P. Allebach, E. J. Delp, P.-J. Chiang, and G. T. Chiu, "Intrinsic and Extrinsic Signatures for Information Hiding and Secure Printing with Electrophotographic Devices," in *NIP & Digital Fabrication Conference*, Society for Imaging Science and Technology, vol. 2003, 2003, pp. 511–515.

[5] G. N. Ali, A. K. Mikkilineni, E. J. Delp, J. P. Allebach, P.-J. Chiang, and G. T. Chiu, "Application of Principal Components Analysis and Gaussian Mixture Models to Printer Identification," in *NIP & Digital Fabrication Conference*, Society for Imaging Science and Technology, vol. 2004, 2004, pp. 301–305.

[6] P.-J. Chiang, A. K. Mikkilineni, E. J. Delp, J. P. Allebach, and G. T.-C. Chiu, "Development of an Electrophotographic Laser Intensity Modulation Model for Extrinsic Signature Embedding," in *NIP & Digital Fabrication Conference*, Society for Imaging Science and Technology, vol. 2007, 2007, pp. 561–564.

[7] S. Suh, J. P. Allebach, G. T.-C. Chiu, and E. J. Delp, "Printer Mechanism-level Data Hiding for Halftone Documents," in *NIP & Digital Fabrication Conference*, Society for Imaging Science and Technology, vol. 2006, 2006, pp. 436–440.

[8] S. Suh, J. P. Allebach, G. T.-C. Chiu, and E. J. Delp, "Printer Mechanism-Level Information Embedding and Extraction for Halftone Documents–New Results," in *NIP & Digital Fabrication Conference*, Society for Imaging Science and Technology, vol. 2007, 2007, pp. 549–553.

[9] Z. Li, W. Jiang, D. Kenzhebalin, A. Gokan, and J. Allebach, "Intrinsic Signatures for Forensic Identification of SOHO Inkjet Printers," in *NIP & Digital Fabrication Conference*, Society for Imaging Science and Technology, vol. 2018, 2018, pp. 231–236.

[10] P.-J. Chiang, J. P. Allebach, and G. T.-C. Chiu, "Extrinsic Signature Embedding and Detection in Electrophotographic Halftoned Images Through Exposure Modulation," *IEEE Transactions on Information Forensics and Security*, vol. 6, no. 3, pp. 946–959, 2011.

[11] A. K. Mikkilineni, G. N. Ali, P.-J. Chiang, G. T.-C. Chiu, J. P. Allebach, and E. J. Delp, "Signature-embedding in Printed Documents for Security and Forensic Applications," in *Security, Steganography, and Watermarking of Multimedia Contents*, 2004, pp. 455–466.

[12] A. K. Mikkilineni, P.-J. Chiang, G. N. Ali, G. T. Chiu, J. P. Allebach, and E. J. Delp III, "Printer Identification Based on Graylevel Co-occurrence Features for Security and Forensic Applications," in *Security, Steganography, and Watermarking of Multimedia Contents VII*, International Society for Optics and Photonics, vol. 5681, 2005, pp. 430–440.

[13] N. Khanna, A. K. Mikkilineni, A. F. Martone, G. N. Ali, G. T.-C. Chiu, J. P. Allebach, and E. J. Delp, "A Survey of Forensic Characterization Methods for Physical Devices," *Digital Investigation*, vol. 3, pp. 17–28, 2006.

[14] Y.-Y. Chen, R. Ulichney, M. Gaubatz, S. Pollard, C.-J. Tai, and J. P. Allebach, "Stegatone Performance Characterization," in *Media Watermarking, Security, and Forensics 2013*, International Society for Optics and Photonics, vol. 8665, 2013, 86650Q.

[15] Y. Xu and J. P. Allebach, "Printed Image Watermarking with Synchronization Using Direct Binary Search," *Electronic Imaging*, vol. 2019, no. 5, pp. 526–1, 2019.

[16] O. Bulan, V. Monga, and G. Sharma, "High Capacity Color Barcodes using Dot Orientation and Color Separability," in *Media Forensics and Security*, International Society for Optics and Photonics, vol. 7254, 2009, p. 725 417.

[17] V. Sebastian, R. Fisher, S. Voloshynovskyy, O. Koval, and T. Pun, "Multilevel 2D Bar Codes: Towards High Capacity Storage Modules for Multimedia Security and Management," vol. 1, no. 4, pp. 405–420, 2005. [Online]. Available: http://cvml.unige.ch/publications/postscript/2005/VillanVoloshynovskiyKovalPun_SPIE2005.pdf.

[18] D. L. Hecht, "Printed Embedded Data Graphical User Interfaces," *Computer*, vol. 34, no. 3, pp. 47–55, 2001.

[19] D. L. Hecht, "Embedded Data Glyph Technology for Hardcopy Digital Documents," *SPIE-Color Hard Copy and Graphics Arts III*, vol. 2171, pp. 341–352, 1994.

[20] R. Ulichney, M. Gaubatz, and S. Simske, "Circular Coding for Data Embedding," in *NIP & Digital Fabrication Conference*, Society for Imaging Science and Technology, vol. 2013, 2013, pp. 142–147.

[21] R. Ulichney, M. Gaubatz, and S. Simske, "Circular Coding with Interleaving Phase," in *Proceedings of the 2014 ACM Symposium on Document Engineering*, ACM, 2014, pp. 21–24.

[22] O. Bulan, G. Sharma, and V. Monga, "Orientation Modulation for Data Hiding in Clustered-dot Halftone Prints," *IEEE Transactions on Image Processing*, vol. 19, no. 8, pp. 2070–2084, 2010.

[23] J. T. Brassil, S. Low, N. F. Maxemchuk, and L. O'Gorman, "Electronic Marking and Identification Techniques to Discourage Document Copying," *IEEE Journal on Selected Areas in Communications*, vol. 13, no. 8, pp. 1495–1504, 1995.

[24] Z. Baharav and D. Shaked, "Watermarking of Dither Halftoned Images," in *Security and Watermarking of Multimedia Contents*, vol. 3657, 1999, pp. 307–316.

[25] S.-G. Wang, *Digital Watermarking using Phase-shifted Stoclustic Screens*, US Patent 6,252,971, Jun. 2001.

[26] G. Sharma and S.-G. Wang, "Show-through Watermarking of Duplex Printed Documents," in *Security, Steganography, and Watermarking of Multimedia Contents*, SPIE, Jan. 2004, pp. 670–684.

[27] R. L. de Queiroz, K. M. Braun, and R. P. Loce, "Detecting Spatially Varying Gray Component Replacement with Application in Watermarking Printed Images," *Journal of Electronic Imaging*, vol. 14, no. 3, pp. 033 016–033 016, 2005.

[28] F. Wang and J. P. Allebach, "Printed Image Watermarking Using Direct Binary Search Halftoning," in *Image Processing (ICIP), 2016 IEEE International Conference on*, IEEE, 2016, pp. 2727–2731.

[29] A. K. Mikkilineni, P.-J. Chiang, S. Suh, G. T. Chiu, J. P. Allebach, and E. J. Delp, "Information Embedding and Extraction for Electrophotographic Printing Processes," in *Security, Steganography, and Watermarking of Multimedia Contents VIII*, International Society for Optics and Photonics, vol. 6072, 2006, p. 607 210.

[30] A. K. Mikkilineni, P.-J. Chiang, G. T.-C. Chiu, J. P. Allebach, and E. J. Delp, "Data Hiding Capacity and Embedding Techniques for Printed Text Documents," in *NIP & Digital Fabrication Conference*, Society for Imaging Science and Technology, vol. 2006, 2006, pp. 444–447.

[31] P.-J. Chiang, G. N. Ali, A. K. Mikkilineni, E. J. Delp, J. P. Allebach, and G. T.-C. Chiu, "Extrinsic Signatures Embedding Using Exposure Modulation for Information Hiding and Secure Printing in Electrophotography," in *NIP & Digital Fabrication Conference*, Society for Imaging Science and Technology, vol. 2004, 2004, pp. 295–300.

[32] P.-J. Chiang, T.-C. Chiu, A. K. Mikkilineni, O. Arslan, R. Moshe, G. Kumontoy, E. J. Delp, and J. P. Allebach, "Extrinsic Signature Embedding in Text Document Using Exposure Modulation for Information Hiding and Secure Printing in Electrophotography," in *NIP & Digital Fabrication Conference*, Society for Imaging Science and Technology, vol. 2005, 2005, pp. 231–234.

[33] P.-J. Chiang, A. K. Mikkilineni, E. J. Delp, J. P. Allebach, and G. T.-C. Chiu, "Extrinsic Signatures Embedding and Detection in Electrophotographic Halftone Images Through Laser Intensity Modulation," in *NIP & Digital Fabrication Conference*, Society for Imaging Science and Technology, vol. 2006, 2006, pp. 432–435.

[34] S. Lin and D. J. Costello, *Error Control Coding*. Prentice Hall, 2001, vol. 2.

[35] C. E. Shannon, "A Mathematical Theory of Communication," *Bell System Technical Journal*, vol. 27, no. 3, pp. 379–423, 1948.

[36] R. G. Gallager, *Information Theory and Reliable Communication*. Springer, 1968, vol. 2.

[37] R. W. Hamming, "Error Detecting and Error Correcting Codes," *The Bell System Technical Journal*, vol. 29, no. 2, pp. 147–160, 1950.

[38] S. B. Wicker and V. K. Bhargava, *Reed-Solomon Codes and Their Applications*. John Wiley & Sons, 1999.

[39] L. Rizzo, "Effective Erasure Codes for Reliable Computer Communication Protocols," *ACM SIGCOMM Computer Communication Review*, vol. 27, no. 2, pp. 24–36, 1997.

[40] M. Sathiamoorthy, M. Asteris, D. Papailiopoulos, A. G. Dimakis, R. Vadali, S. Chen, and D. Borthakur, "Xoring Elephants: Novel Erasure Codes for Big Data," in *Proceedings of the VLDB Endowment*, VLDB Endowment, vol. 6, 2013, pp. 325–336.

[41] M. N. Krohn, M. J. Freedman, and D. Mazieres, "On-the-fly Verification of Rateless Erasure Codes for Efficient Content Distribution," in *IEEE Symposium on Security and Privacy, 2004. Proceedings. 2004*, IEEE, 2004, pp. 226–240.

[42] D. J. MacKay, "Fountain Codes," *IEE Proceedings-Communications*, vol. 152, no. 6, pp. 1062–1068, 2005.

[43] J. W. Byers, M. Luby, M. Mitzenmacher, and A. Rege, "A Digital Fountain Approach to Reliable Distribution of Bulk Data," *ACM SIGCOMM Computer Communication Review*, vol. 28, no. 4, pp. 56–67, 1998.

[44] M. C. Bogino, P. Cataldi, M. Grangetto, E. Magli, and G. Olmo, "Sliding-window Digital Fountain Codes for Streaming of Multimedia Contents," in *2007 IEEE International Symposium on Circuits and Systems*, IEEE, 2007, pp. 3467–3470.

[45] M. Luby, A. Shokrollahi, M. Watson, T. Stockhammer, and L. Minder, "RaptorQ Forward Error Correction Scheme for Object Delivery," *RFC*, vol. 6330, pp. 1–69, 2011.

[46] M. Luby, "LT Codes," in *The 43rd Annual IEEE Symposium on Foundations of Computer Science, 2002. Proceedings.*, IEEE, 2002, pp. 271–280.

[47] R. Palanki and J. S. Yedidia, "Rateless Codes on Noisy Channels," in *ISIT*, Citeseer, 2004, p. 37.

[48] F. James, "A Review of Pseudorandom Number Generators," *Computer Physics Communications*, vol. 60, no. 3, pp. 329–344, 1990.

[49] A. Shokrollahi, "Raptor Codes," *IEEE/ACM Transactions on Networking (TON)*, vol. 14, no. SI, pp. 2551–2567, 2006.

[50] O. Etesami and A. Shokrollahi, "Raptor Codes on Binary Memoryless Symmetric Channels," *IEEE Transactions on Information Theory*, vol. 52, no. 5, pp. 2033–2051, 2006.

[51] N. Damera-Venkata, J. Yen, V. Monga, and B. L. Evans, "Hardcopy Image Barcodes via Block-Error Diffusion," *IEEE Transactions on Image Processing*, vol. 14, no. 12, pp. 1977–1989, 2005.

[52] D. Kacker, T. Camis, and J. P. Allebach, "Electrophotographic Process Embedded in Direct Binary Search," *IEEE Transactions on Image Processing*, vol. 11, no. 3, pp. 243–257, 2002.

[53] O. Bulan, V. Monga, G. Sharma, and B. Oztan, "Data Embedding in Hardcopy Images via Halftone-dot Orientation Modulation," in *Security, Forensics, Steganography, and Watermarking of Multimedia Contents*, 2008, p. 68190C.

[54] R. Ulichney, M. Gaubatz, and S. Simske, "Encoding Information in Clustered-dot Halftones," in *NIP & Digital Fabrication Conference*, Society for Imaging Science and Technology, vol. 2010, 2010, pp. 602–605.

[55] W. K. Gilliland, C. G. Midgley, A. M. Murphy, and W. T. Bowerman, *Printer Toner Usage Indicator with Image Weighted Calculation*, US Patent 5,349,377, Sep. 1994.

[56] Z. Zhao, R. Ulichney, M. Gaubatz, S. Pollard, and J. P. Allebach, "Advances in the Decoding of Data-Bearing Halftone Images," in *NIP & Digital Fabrication Conference*, Society for Imaging Science and Technology, vol. 2019, 2019, pp. 162–167.

[57] Z. Zhao, Y. Xu, R. Ulichney, M. Gaubatz, S. Pollard, and J. P. Allebach, "Data-bearing Halftone Image Alignment and Assessment on 3D Surface," *Electronic Imaging*, vol. 2020, no. 15, pp. 196–1, 2020.

[58] Y. Sun and J. P. Allebach, "Analyzing the Decoding Rate of Circular Coding in a Noisy Transmission Channel," *Proceedings of Media Watermarking, Security, and Forensics 2020*, pp. 26–20, 2020.

[59] Y. Sun, R. Ulichney, M. Gaubatz, S. Pollard, S. Simske, and J. P. Allebach, "Analysis of a Visually Significant Bar Code System Based on Circular Coding," *Color Imaging XXIII: Displaying, Processing, Hardcopy, and Application*, vol. 2018, no. 16, 29 Janurary - 2 February 2018.

[60] H. Santos-Villalobos, H. J. Park, C. Kim, P. Choe, R. Kumontoy, K. Low, K. Oldenburger, M. Ortiz, X. Lehto, M. Lehto, *et al.*, "A Web-based Self-diagnosis Tool to Solve Print Quality Issues," in *NIP & Digital Fabrication Conference*, Society for Imaging Science and Technology, vol. 2006, 2006, pp. 465–471.

[61] W. Jang, M.-C. Chen, J. P. Allebach, and G. T.-C. Chiu, "Print Quality Test Page," *Journal of Imaging Science and Technology*, vol. 48, no. 5, pp. 432–446, 2004.

[62] C. Kim, P. Choe, M. Lehto, and J. Allebach, "Development of a Web-based Interactive Self-help Troubleshooting Tool for Print Quality Problems," in *International Conference on Human-Computer Interaction, Las Vegas, Nevada USA*, 2005, pp. 22–27.

[63] P. Choe, C. Kim, M. R. Lehto, X. Lehto, and J. Allebach, "Evaluating and Improving a Self-help Technical Support Web Site: Use of Focus Group Interviews," *International Journal of Human-Computer Interaction*, vol. 21, no. 3, pp. 333–354, 2006.

[64] P. Choe, C. Kim, M. R. Lehto, and J. Allebach, "Experimental Comparison of Adaptive vs. Static Thumbnail Displays," in *International Conference on Human-Computer Interaction*, Springer, 2007, pp. 41–48.

[65] P. Choe, M. R. Lehto, and J. Allebach, "Self-help Troubleshooting by Q-KE-CLD Based on a Fuzzy Bayes Model," in *Symposium on Human Interface and the Management of Information*, Springer, 2007, pp. 391–400.

[66] H. J. Santos-Villalobos, V. Loewen, and J. P. Allebach, "Houston, We Have A Color Issue!" In *Color Imaging XIV: Displaying, Processing, Hardcopy, and Applications*, International Society for Optics and Photonics, vol. 7241, 2009, p. 72411D.

[67] R. Kumontoy, K. Low, M. Ortiz, C. Kim, P. Choe, S. Leman, K. Oldenburger, M. Lehto, X. Lehto, H. Santos-Villalobos, *et al.*, "Web-based Diagnosis Tool for Customers to Self-solve Print Quality Issues," *Journal of Imaging Science and Technology*, vol. 54, no. 4, pp. 40 503–1, 2010.

[68] W. Jang and J. P. Allebach, "Simulation of Print Quality Defects," *Journal of Imaging Science and Technology*, vol. 49, no. 1, pp. 1–18, 2005.

[69] H. J. Santos-Villalobos, V. Loewen, M. Lehto, and J. Allebach, "A Web-based Troubleshooting Tool to Help Customers Self-solve Color Issues With a Digital Printing Workflow," in *Imaging and Printing in a Web 2.0 World II*, International Society for Optics and Photonics, vol. 7879, 2011, p. 787 906.

[70] L. A. Fletcher and R. Kasturi, "A Robust Algorithm for Text String Separation From Mixed Text/Graphics Images," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 10, no. 6, pp. 910–918, 1988.

[71] L. Likforman-Sulem, A. Hanimyan, and C. Faure, "A Hough Based Algorithm for Extracting Text Lines in Handwritten Documents," in *Proceedings of 3rd International Conference on Document Analysis and Recognition*, IEEE, vol. 2, 1995, pp. 774–777.

[72] Y. Pu and Z. Shi, "A Natural Learning Algorithm Based on Hough Transform for Text Lines Extraction in Handwritten Documents," in *Advances In Handwriting Recognition*, World Scientific, 1999, pp. 141–150.

[73] Z. Shi and V. Govindaraju, "Line Separation for Complex Document Images Using Fuzzy Runlength," in *First International Workshop on Document Image Analysis for Libraries, 2004. Proceedings.*, IEEE, 2004, pp. 306–312.

[74] F. Le Bourgeois, H. Emptoz, E. Trinh, and J. Duong, "Networking Digital Document Images," in *Proceedings of Sixth International Conference on Document Analysis and Recognition*, IEEE, 2001, pp. 379–383.

[75] G. Louloudis, B. Gatos, I. Pratikakis, and C. Halatsis, "Text Line Detection in Handwritten Documents," *Pattern Recognition*, vol. 41, no. 12, pp. 3758–3772, 2008.

[76] G. Louloudis, B. Gatos, and C. Halatsis, "Text Line Detection in Unconstrained Handwritten Documents Using a Block-based Hough Transform Approach," in *Ninth International Conference on Document Analysis and Recognition (ICDAR 2007)*, IEEE, vol. 2, 2007, pp. 599–603.

[77] Y. Li, Y. Zheng, and D. Doermann, "Detecting Text Lines in Handwritten Documents," in *18th International Conference on Pattern Recognition (ICPR'06)*, IEEE, vol. 2, 2006, pp. 1030–1033.

[78] Y. Guo, Y. Sun, P. Bauer, J. P. Allebach, and C. A. Bouman, "Text Line Detection Based on Cost Optimized Local Text Line Direction Estimation," in *Color Imaging XX: Displaying, Processing, Hardcopy, and Applications*, International Society for Optics and Photonics, vol. 9395, 2015, p. 939 507.

VITA

Yufang Sun received her BS in Electrical Engineering from the University of Jilin from China (2004). She is currently a PhD student, working as image processing and data analysis research assistant with Prof. Jan Allebach, in the School of Electrical and Computer Engineering at Purdue University. Her research interests are in image information embedding, decoding error analysis, etc. She has been working on the projects of circular coding and stegaframe detection, both sponsored by HP Labs.