

# STARS

University of Central Florida  
**STARS**

---


Electronic Theses and Dissertations, 2004-2019

---

2017

## Prototype Development in General Purpose Representation and Association Machine Using Communication Theory

Huihui Li  
*University of Central Florida*

 Part of the [Electrical and Computer Engineering Commons](#)  
Find similar works at: <https://stars.library.ucf.edu/etd>  
University of Central Florida Libraries <http://library.ucf.edu>

This Doctoral Dissertation (Open Access) is brought to you for free and open access by STARS. It has been accepted for inclusion in Electronic Theses and Dissertations, 2004-2019 by an authorized administrator of STARS. For more information, please contact [STARS@ucf.edu](mailto:STARS@ucf.edu).

---

### STARS Citation

Li, Huihui, "Prototype Development in General Purpose Representation and Association Machine Using Communication Theory" (2017). *Electronic Theses and Dissertations, 2004-2019*. 5534.  
<https://stars.library.ucf.edu/etd/5534>



PROTOTYPE DEVELOPMENT IN GENERAL PURPOSE REPRESENTATION AND  
ASSOCIATION MACHINE USING COMMUNICATION THEORY

by

HUIHUI LI

B.S.E.E. Nanjing University of Posts and Telecommunications, 2011

M.S.E.E. University of Central Florida, 2013

A dissertation submitted in partial fulfilment of the requirements  
for the degree of Doctor of Philosophy  
in the Department of Electrical and Computer Engineering  
in the College of Engineering and Computer Science  
at the University of Central Florida  
Orlando, Florida

Summer Term  
2017

Major Professor: Lei Wei

© 2017 Huihui Li

## **ABSTRACT**

Biological system study has been an intense research area in neuroscience and cognitive science for decades of years. Biological human brain is created as an intelligent system that integrates various types of sensor information and processes them intelligently. Neurons, as activated brain cells help the brain to make instant and rough decisions. From the 1950s, researchers start attempting to understand the strategies the biological system employs, then eventually translate them into machine-based algorithms. Modern computers have been developed to meet our need to handle computational tasks which our brains are not capable of performing with precision and speed. While in these existing man-made intelligent systems, most of them are designed for specific purposes. The modern computers solve sophistic problems based on fixed representation and association formats, instead of employing versatile approaches to explore the unsolved problems.

Because of the above limitations of the conventional machines, General Purpose Representation and Association Machine (GPRAM) System is proposed to focus on using a versatile approach with hierarchical representation and association structures to do a quick and rough assessment on multitasks. Through lessons learned from neuroscience, error control coding and digital communications, a prototype of GPRAM system by employing (7,4) Hamming codes and short Low-Density Parity Check (LDPC) codes is implemented. Types of learning processes are presented, which prove the capability of GPRAM for handling multitasks.

Furthermore, a study of low resolution simple patterns and face images recognition using an Image Processing Unit (IPU) structure for GPRAM system is presented. IPU structure consists of a randomly constructed LDPC code, an iterative decoder, a switch and scaling, and decision devices. All the input images have been severely degraded to mimic human Visual Information Variability (VIV) experienced in human visual system. The numerical results show that 1) IPU can reliably

recognize simple pattern images in different shapes and sizes; 2) IPU demonstrates an excellent multi-class recognition performance for the face images with high degradation. Our results are comparable to popular machine learning recognition methods towards images without any quality degradation; 3) A bunch of methods have been discussed for improving IPU recognition performance, e.g. designing various detection and power scaling methods, constructing specific LDPC codes with large minimum girth, etc.

Finally, novel methods to optimize M-ary PSK, M-ary DPSK, and dual-ring QAM signaling with non-equal symbol probabilities over AWGN channels are presented. In digital communication systems, MPSK, MDPSK, and dual-ring QAM signaling with equiprobable symbols have been well analyzed and widely used in practice. Inspired by bio-systems, we suggest investigating signaling with non-equiprobable symbol probabilities, since in bio-systems it is highly-unlikely to follow the ideal setting and uniform construction of single type of system. The results show that the optimizing system has lower error probabilities than conventional systems and the improvements are dramatic. Even though the communication systems are used as the testing environment, clearly, our final goal is to extend current communication theory to accommodate or better understand bio-neural information processing systems.

To my beloved parents, Kehong Li and Jie Yan

## ACKNOWLEDGMENTS

I would like to thank my advisor Dr. Lei Wei, for spending enormous amount of time and efforts in advising my doctoral research. Without his inspiration, I could not complete my Ph.D. study and make this dissertation possible. His ambitions and self-motivations encourage me to push my limits and obtain fruitful research outcomes.

And also, I would like to thank my committee members, Dr. Nazanin Rahnavard, Dr. Azadeh Vosoughi from Department of Electrical and Computer Engineering, Dr. Niels da Vitoria Lobo from Department of Computer Science and Dr. Wei Wang from Department of Phycology, who have provided valuable suggestions in the preparation of my proposal and dissertation. Moreover, I appreciate the support from all the members from Signal and Communications Lab. Particularly I would like to acknowledge Bowen Dai, Yin Li, Alireza Sani, Mojtaba Shiraza, Toktam Amazade. Without their constructive suggestions I could not finish the dissertation that soon.

I would like to thank the Department of Electrical and Computer Engineering, University of Central Florida. Without the continuous Teaching Assistantship support throughout my Ph.D. period, I would not be able to take risky approach to this direction.

Finally, I would like to thank my parents, Kehong Li and Jie Yan. Your unconditional love always support me going through the tough periods in my doctoral study.

## TABLE OF CONTENTS

|                                                                                             |     |
|---------------------------------------------------------------------------------------------|-----|
| LIST OF FIGURES . . . . .                                                                   | xii |
| LIST OF TABLES . . . . .                                                                    | xv  |
| CHAPTER 1: INTRODUCTION . . . . .                                                           | 1   |
| Background and Motivation . . . . .                                                         | 1   |
| Dissertation Outline . . . . .                                                              | 3   |
| Contributions . . . . .                                                                     | 4   |
| Paper List . . . . .                                                                        | 5   |
| CHAPTER 2: LITERATURE REVIEW . . . . .                                                      | 7   |
| GPRAM and Low-resolution Human Face Recognition . . . . .                                   | 7   |
| Low-Density Parity Check Codes . . . . .                                                    | 10  |
| Digital Modulation Schemes . . . . .                                                        | 12  |
| CHAPTER 3: PROTOTYPE OF GENERAL PURPOSE REPRESENTATION AND ASSO-<br>CIATE MACHINE . . . . . | 15  |
| Overview GPRAM and Design Principle . . . . .                                               | 15  |



|                                                                                                                                              |    |
|----------------------------------------------------------------------------------------------------------------------------------------------|----|
| Implementation of GPRAM . . . . .                                                                                                            | 18 |
| Define Operations . . . . .                                                                                                                  | 18 |
| Define Connection Rules . . . . .                                                                                                            | 19 |
| Perform the Task Using Hamming (7,4) code . . . . .                                                                                          | 21 |
| Perform the Task Using Other LDPC Codes . . . . .                                                                                            | 22 |
| GPRAM System with Switch . . . . .                                                                                                           | 24 |
| Perform the Task Using Hamming (7,4) code . . . . .                                                                                          | 25 |
| Perform the Task Using Other Codes . . . . .                                                                                                 | 26 |
| Improvement in Connections and Operations . . . . .                                                                                          | 27 |
| Improvement in Learning Using Perfect Codeword . . . . .                                                                                     | 29 |
| Two-task Learning Using Perfect Codeword and Multiple Connection . . . . .                                                                   | 30 |
| Three-task Learning Using Perfect Codeword and Multiple Connection . . . . .                                                                 | 32 |
| Improvement in Multitask Learning Using Progressive Learning . . . . .                                                                       | 33 |
| Summary . . . . .                                                                                                                            | 33 |
| <br>                                                                                                                                         |    |
| CHAPTER 4: IMAGE PROCESSING UNIT FOR GPRAM FOR RECOGNIZING LOW-<br>RESOLUTION FACE IMAGES WITH VISUAL INFORMATION VARIABIL-<br>ITY . . . . . | 36 |
| Define Structure of IPU . . . . .                                                                                                            | 36 |

|                                                                                                    |    |
|----------------------------------------------------------------------------------------------------|----|
| Understanding IPU Using Simple Images . . . . .                                                    | 40 |
| Bit Convergence . . . . .                                                                          | 41 |
| Simple Examples for Differentiating Circle Images . . . . .                                        | 42 |
| Task 1: Differentiation Between Images {B, D} and Null Image . . . . .                             | 46 |
| Task 2: Differentiation Between Image B and D . . . . .                                            | 47 |
| Task 3: Differentiation Between Images {A, B, C, D} and Null Image . . . . .                       | 48 |
| Task 4: Differentiation Between Images {A, B, D} and Image C . . . . .                             | 48 |
| Simple Examples for Differentiating Patterns in Different Shapes and Sizes . . . . .               | 48 |
| Task 5: Differentiation Among Group I, II, III and IV . . . . .                                    | 49 |
| Task 6: Differentiation Between {C, G, K, O} and {A, B, D, E, F, H, I, J,<br>L, M, N, P} . . . . . | 50 |
| VIV and Power Scaling Method . . . . .                                                             | 51 |
| Visual Information Variability . . . . .                                                           | 52 |
| Point Spread (PS) . . . . .                                                                        | 52 |
| Fixation Motion (FM) . . . . .                                                                     | 52 |
| Orientation Movement (OM) . . . . .                                                                | 53 |
| Power Scaling Method . . . . .                                                                     | 53 |
| Results of Differentiating Different Patterns with VIV . . . . .                                   | 55 |

|                                                                                                                |    |
|----------------------------------------------------------------------------------------------------------------|----|
| Experimental Procedures and IPU Detections for Simple Images . . . . .                                         | 56 |
| Detector Design . . . . .                                                                                      | 57 |
| Results of Detectors IV and V in Tasks 2, 5, 6 . . . . .                                                       | 59 |
| Features Appeared in Task 5 . . . . .                                                                          | 59 |
| Recognition Low-Resolution Face Images with VIV . . . . .                                                      | 63 |
| Numerical Results . . . . .                                                                                    | 66 |
| Face Recognition: One Out of Ten Faces . . . . .                                                               | 66 |
| Impact of LDPC Codes with Large Minimum Girth to the Results of One Out of Ten . . . . .                       | 69 |
| Face Recognition: One Out of 20 and 40 . . . . .                                                               | 70 |
| Face Recognition in Multi-class Classification . . . . .                                                       | 70 |
| Summary . . . . .                                                                                              | 72 |
| <br>                                                                                                           |    |
| CHAPTER 5: OPTIMIZING MPSK, MDPSK AND DUAL-RING QAM SIGNALING WITH<br>NON-EQUAL SYMBOL PROBABILITIES . . . . . | 79 |
| Signals and Systems of MPSK and MDPSK . . . . .                                                                | 79 |
| MPSK . . . . .                                                                                                 | 79 |
| MDPSK . . . . .                                                                                                | 82 |
| Optimal Detection and Error Probability for MPSK and MDPSK Signaling . . . . .                                 | 83 |

|                                                                                     |         |
|-------------------------------------------------------------------------------------|---------|
| Method 1: Simplified Method . . . . .                                               | 83      |
| Method 2: Optimized Method . . . . .                                                | 86      |
| Extension to Optimizing Dual-ring QAM Signaling with Non-equal Symbol Probabilities | 88      |
| Signals and Systems of Dual-ring QAM . . . . .                                      | 88      |
| Computing Decision Region for Dual-ring QAM System . . . . .                        | 90      |
| Optimizing Symbol Error Rate for Dual-ring QAM . . . . .                            | 92      |
| Numerical Results . . . . .                                                         | 93      |
| Numerical Result of MPSK . . . . .                                                  | 94      |
| Numerical Result of MDPSK . . . . .                                                 | 96      |
| Numerical Result of Dual-ring QAM . . . . .                                         | 97      |
| Summary . . . . .                                                                   | 98      |
| <br>CHAPTER 6: CONCLUSION AND FUTURE WORK . . . . .                                 | <br>102 |
| Conclusion . . . . .                                                                | 102     |
| Future Work . . . . .                                                               | 103     |
| <br>LIST OF REFERENCES . . . . .                                                    | <br>105 |

## LIST OF FIGURES

|                                                                                                                                               |    |
|-----------------------------------------------------------------------------------------------------------------------------------------------|----|
| Figure 2.1: A Tanner Graph Representation for Example Code . . . . .                                                                          | 11 |
| Figure 3.1: Block Diagram of a Simple GPRAM System . . . . .                                                                                  | 16 |
| Figure 3.2: A Simple Example of GPRAM System . . . . .                                                                                        | 17 |
| Figure 3.3: Connections of One Individual GPRAM System . . . . .                                                                              | 20 |
| Figure 3.4: Parity Check for (8, 4) and (10, 5) LDPC codes . . . . .                                                                          | 23 |
| Figure 3.5: Training Result for Random Codes . . . . .                                                                                        | 24 |
| Figure 3.6: Training Results for Different Codes Considering Three-state . . . . .                                                            | 27 |
| Figure 3.7: Training Results for Different Codes Using RB . . . . .                                                                           | 30 |
| Figure 3.8: Training Results for Different Codes Using PCMC in Two Tasks Training . .                                                         | 31 |
| Figure 3.9: Training Results for Different Codes Using PCMC in Three Tasks Training .                                                         | 34 |
| Figure 3.10 Training Result for Different Codes Using PLMT in Three Tasks Training . .                                                        | 35 |
| Figure 4.1: Structure of IPU for GPRAM for One Frame . . . . .                                                                                | 37 |
| Figure 4.2: Illustration in the Circle Image and in Tanner Graph Structure for the First<br>Tier Connections of the 739-th Position . . . . . | 43 |
| Figure 4.3: Four Sets of 16 Testing Figures . . . . .                                                                                         | 44 |

|                                                                                                                                                                                                   |    |
|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----|
| Figure 4.4: Differentiation Error Rates in Independent Number of Noise Seeds for 10e5 Trials . . . . .                                                                                            | 62 |
| Figure 4.5: Error Rates of Different Noise Variance for Task 5 . . . . .                                                                                                                          | 63 |
| Figure 4.6: Ten Different Face Images of One Distinct Object . . . . .                                                                                                                            | 64 |
| Figure 4.7: Effects of Point Spread ( $\beta_1$ ), Drifting-like Motion( $\beta_2$ ), Head Orientation Rotation ( $\beta_3$ ) and Gaussian Noise ( $\sigma_{a1}$ ) on One Face Image . . . . .    | 65 |
| Figure 4.8: Error Rates of Two Types of Detectors at Different Noise Variance . . . . .                                                                                                           | 68 |
| Figure 4.9: Histogram for 10000 Trials in Independent Number of Noise Seeds at $\sigma_{a1} = 1.5$ , $\sigma_{a2} = 2.25$ for LDPC Codes (a) With $H$ Matrix and (b) Without $H$ Matrix . . . . . | 74 |
| Figure 4.10 Impact of LDPC codes with Different Large Minimum Girth in (a) Power Scaling Method 3 and (b) Power Scaling Method 4 . . . . .                                                        | 75 |
| Figure 4.11 Error Rates for 10e5 Trials in Face Images with the Variation of $\sigma_{a1}$ . . . . .                                                                                              | 76 |
| Figure 4.12 Error Rates for Testing on One Out of 20 and One Out of 40 Images Comparing Between With $H$ and Without $H$ matrix . . . . .                                                         | 77 |
| Figure 4.13 Recognition Accuracy Rate Comparing with Other Methods Tested On the Images with VIV . . . . .                                                                                        | 78 |
| Figure 5.1: (a) Signal Constellation for $M$ -PSK with Non-uniform Symbols; (b) Signal Constellation for $M$ -PSK in New Coordinates for Method 1 . . . . .                                       | 80 |
| Figure 5.2: Signal Constellation for Dual-ring QAM . . . . .                                                                                                                                      | 88 |

|                                                                                                                                                                                              |     |
|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----|
| Figure 5.3: P(e) of Optimized and Non-optimized 3PSK Signaling Per Information Bit<br>Energy . . . . .                                                                                       | 94  |
| Figure 5.4: P(e) of Optimized and Non-optimized 4PSK Signaling Per Information Bit<br>Energy . . . . .                                                                                       | 98  |
| Figure 5.5: P(e) of Optimized and Non-optimized 8PSK Signaling Per Information Bit<br>Energy . . . . .                                                                                       | 99  |
| Figure 5.6: P(e) of Optimized and Non-optimized 8DPSK Signaling Per Information Bit<br>Energy . . . . .                                                                                      | 99  |
| Figure 5.7: P(e) of Optimized and Non-optimized Dual-ring 8QAM Signaling Per Infor-<br>mation Bit Energy . . . . .                                                                           | 100 |
| Figure 5.8: Signal Constellations and Decision Regions for Optimal 8QAM When $P_m$<br>= $\{0.03, 0.01, 0.04, 0.02, 0.25, 0.15, 0.20, 0.30\}$ , $\varepsilon_b/N_0= 10.17\text{dB}$ . . . . . | 101 |

## LIST OF TABLES

|                                                                                                                                        |    |
|----------------------------------------------------------------------------------------------------------------------------------------|----|
| Table 3.1: Five Tasks in the Test . . . . .                                                                                            | 17 |
| Table 3.2: Examples for Three Typical Performance . . . . .                                                                            | 22 |
| Table 3.3: Three Typical Cases for Three-states . . . . .                                                                              | 26 |
| Table 3.4: Three Typical Cases for RB . . . . .                                                                                        | 29 |
| Table 3.5: Example of (20,10) LDPC Codes Using PCMC in Two Tasks . . . . .                                                             | 31 |
| Table 3.6: Example of (20, 10) LDPC Codes Using PCMC in Three Tasks . . . . .                                                          | 32 |
| Table 3.7: Example of (40,20) LDPC Codes Using PLMT . . . . .                                                                          | 35 |
| Table 4.1: Number of Survived Positions for Different Cases . . . . .                                                                  | 47 |
| Table 4.2: Performance of Detector III in Differentiating Group I, II, III and IV . . . . .                                            | 50 |
| Table 4.3: Performance Comparison in Detectors for Images With and Without VIV . . . . .                                               | 56 |
| Table 4.4: Detector IV and Detector V . . . . .                                                                                        | 58 |
| Table 4.5: Performance Comparison in Detectors IV, V for Images With and Without<br>VIV . . . . .                                      | 60 |
| Table 4.6: Differentiation Error Rate for Recognizing One Out of 16 Images with $\sigma_{a1} =$<br>$1.3, \sigma_{a2} = 1.95$ . . . . . | 62 |
| Table 4.7: Four Types of Error Rates for Task 5 . . . . .                                                                              | 63 |



|                                                                                                                                                                  |    |
|------------------------------------------------------------------------------------------------------------------------------------------------------------------|----|
| Table 4.8: Error Rates for Recognizing One Out of 10 Faces . . . . .                                                                                             | 67 |
| Table 4.9: Recognition Accuracy Rate on ORL Database (Mean $\pm$ Std-dev) . . . . .                                                                              | 71 |
| Table 4.10: Recognition Rate for 10 Cases of Training and Test Sets in 40 Classes with<br><i>scale</i> = 0.5, $\sigma_{a1}$ = 0.8, $\sigma_{a2}$ = 1.0 . . . . . | 71 |
| Table 5.1: $P(e)$ , $\phi_m$ and $\alpha_m$ for 3PSK at $\varepsilon_b/N_0 = 7$ dB . . . . .                                                                     | 96 |
| Table 5.2: $P(e)$ , $\phi_m$ and $\alpha_m$ for 4PSK at $\varepsilon_b/N_0 = 7$ dB . . . . .                                                                     | 96 |

# CHAPTER 1: INTRODUCTION

## Background and Motivation

The modern computer was developed in the 1940s to meet our need to handle computational tasks which our brains are not capable of performing with precision and speed. Visionary thoughts from Turing [1] [2] and von Neumann [3] laid the foundation in terms of mathematical principles and engineering structures. During the same period, Shannon laid the foundation of the mathematical measurement of information [4]. Over the last sixty years, telecom engineers have learned how to achieve the Shannon limit using coding and detection theory. During the process, we have accumulated a wealth of knowledge on how to preserve, transmit, and detect information.

Apparently, given the current computer architecture designed by scientists and engineers, the modern computers are created for solving mathematical problems with scientific models involved. However, human mind could play key roles in many aspects [5], including making rough estimations and decisions, and guiding the general plan. This encourages the researchers to develop a machine to mimic part functions of human brain. Therefore General Purpose Representation and Association Machine(GPRAM) system is proposed in [5][6]. This machine could make quick estimates at certain confidence with vague computation and approximation.

Meanwhile, over the past 20 years, huge amount of progress in understanding how vision works in computational neuroscience has been made[7]. In [8], researchers summarized nineteen results of face recognition by humans that all computer vision researchers should know about. In [8], authors endeavored to make readers understand the strategies bio-logical system employs, as a first step towards eventually translating them into machine-based algorithms. From the results human visual scientists discovered, increased knowledge about the ways people recognize each other plays a role

of stimulus guiding us to develop practical automatic face-recognition system in IPU. For example, researchers figured out (1) Human can recognize familiar faces in very low-resolution[9][10][11]; (2) Facial features are processed holistically[12][13][14][15].

It is definitely amazing how far we have achieved on Medicine [16] and Neuroscience [17]. Researchers have also tried to reconstruct the visual view from animals through non-intruding technology [18] or directly recording neural activities [19]. But researchers found the animal view they directly obtained was only recognizable and it was still impossible to form the sharp and clear picture. In [9], the author concluded blurringly sampled and quantized images could actually improve the recognition. In [20], the importance of self-noises in a system has been well described, which indicates the possibility of applying LDPC codes as a representative layer in such system. In [21] and [22], authors are inspired by the biological system, presenting the feasibility of implementing matched filters for the purpose of achieving statistically optimal performance. In [23], a device achieving hyper-acuity vision is proposed. In [24], authors figured out how the Image Processing Unit structure reliably recognize digits images despite the image quality being poor, as well as how it provides a hyper-acuity capability comparable to human visual systems. In [23] and [24], it is easy to see the realistic Visual Information Variability (VIV) including point spread, fixation movement, orientation rotation and Gaussian noise are experienced by bio-visual systems. The results in [24] indicates IPU could reliably recognize digits despite the image quality being poor, which further demonstrates VIV may be unanticipatedly beneficial for developing a multitask visual system.

In summary, all the aforementioned results interest us to build up the prototype of GPRAM system. Inspired by the bio-visual system, we are curious if there exists a versatile system that could holistically learn and recognize low-resolution images with serious image quality degradation, which is closely experienced in the human visual system.

## Dissertation Outline

In this dissertation, the concentration is on studying the prototype development in GPRAM system using communication theory. The dissertation is organized as follows:

The literature review is presented in Chapter 2. GPRAM concept and low resolution facial image recognition are reviewed at first. Error control coding theory including LDPC codes and (7,4) Hamming codes are introduced next. At last, the digital modulation techniques involved in communication theory is presented.

In Chapter 3, we go from bio-system to observe whether simple LDPC codes could be used in telecommunication systems. It helps us build a testable platform to evaluate the theory proposed in [5] [6]. We first overview the concept of GPRAM, then construct a prototype GPRAM system using (7,4) Hamming and simple LDPC codes. Through the experimental results we demonstrate GPRAM principles. Furthermore, types of learning processes are proposed for comparison, which prove the capability of GPRAM for handling multitasks. After highlighting the future challenges, we conclude this chapter.

Chapter 4 presents a study on low resolution simple patterns and face images recognition using Image Processing Unit (IPU) structure for General Purpose Representation and Association Machine system. IPU structure consists of a randomly constructed LDPC code, an iterative decoder, a switch and scaling, and decision devices. All the input images have been severely degraded to mimic human Visual Information Variability (VIV) experienced in human visual system. The numerical results show that 1) IPU can reliably recognize simple pattern images in different shapes and sizes; 2) IPU proves an excellent multi-class recognition performance for the face images with high degradation. Our results are comparable to popular recognition methods towards images without any quality degradation; 3) A bunch of methods have been discussed for improving IPU

recognition performance, e.g. designing various detection and power scaling methods, constructing specific LDPC codes with large minimum girth, etc.

Chapter 5 derives formulas to optimize signal constellations, decision regions, and symbol error rates for  $M$ -ary PSK,  $M$ -ary DPSK, and dual-ring QAM signaling with non-equal symbol probabilities over AWGN channel. In each modulation scheme, the performance is evaluated and compared in symbol error rate at the same information bit energy. The results show that the optimizing system has lower error probabilities than two conventional systems, 1) a system with non-equiprobable symbols using source coding; 2) a system with non-equiprobable symbols using equally spaced constellation. Several examples presented in this chapter elaborate that the improvements are dramatic.

Chapter 6 concludes the dissertation and discusses the future work.

## Contributions

The major contributions in this dissertation are listed below.

(1) Construct a prototype of General Purpose Representation and Association Machine(GPRAM) using (7,4) Hamming code and simple LDPC codes. Through the simulations, we have demonstrated GPRAM concept, illustrated the differences between conventional system design and our new versatile approach, and also have highlighted challenges faced in implementing GPRAM systems. (Chapter 3)

(2) Develop several essential functions in GPRAM prototype, improve learning processes for three-state systems and derive progressive learning methods for GPRAM in multitask learning. (Chapter 3)

(3) Construct an Image Processing Unit(IPU) structure for GPRAM system to recognize low-resolution images with Visual Information Variability(VIV). Different tasks have been selected to test the performance ability of IPU/GPRAM system, which include (a) simple pattern images in different types and sizes; (b) human face images with varying lighting, facial expressions and facial details. The results we found are (a) IPU could recognize simple pattern images and human face image in poor quality with very high recognition rate; (b) IPU could handle multi-class classifications in human face recognition, where we could achieve  $93.45\% \pm 1.44$  recognition rate for 5/5 training/testing, comparable to the state-of-the-art algorithms on images with VIV; (c) Demonstrate the impact of LDPC codes with large minimum girth to the recognition results. All of them indicate a potential new research direction for future understanding of both GPRAM systems and human visual systems. (Chapter 4)

(4) Propose a method and derive formulas to optimize signal constellations, decision regions, and symbol error rates for M-ary PSK, M-ary DPSK, and dual-ring QAM signaling with non-equal symbol probabilities over AWGN channels. Specifically, it is shown numerically the approximately optimal dual-ring QAM system with non-equal symbol probabilities leads to performance gain around 2.8 dB comparing conventional systems. Even though the communication systems are used as the testing environment, clearly, it demonstrates the possibility of extending current communication theory to accommodate or better understand bio-neural information processing systems. (Chapter 5)

## Paper List

### **Journal Papers:**

J1 B. Dai, H. Li and L. Wei, "Image Processing Unit for General-Purpose Representation and

Association System for Recognizing Low-Resolution Digits with Visual Information Variability”, *IEEE Transactions on Systems, Man and Cybernetics: Systems*, vol. pp, no. 99, pp 1-12, Sep 2016

J2 H. Li, B. Dai and L. Wei, ”Image Processing Unit for General Purpose and Association System for Recognizing Low-Resolution Facial Images with Visual Information Variability”, submitted to *IET Computer Vision*

J3 H. Li and L. Wei, ”Optimizing MPSK, MDPSK and Dual-ring QAM Signaling with Non-equal Symbol Probabilities”, ready for submission

J4 B. Chen, L. Wei, H. Li, ”Teaching complicated conceptual knowledge with simulation videos in foundational electrical engineering courses”, *Journal of Technology and Science Education*, Vol. 6, no. 3, 2016

**Conference Papers:**

C1 H. Li, B. Dai, S. Schultz and L. Wei, ”General Purpose Representation and Association Machine Part 3: Prototype Study using LPDC codes”, *SoutheastCon, 2013 Proceedings of IEEE*, 10.1109/SECON.2013.6567484 (Oral Presentation)

C2 H. Li and L. Wei, ”General Purpose Representation and Association Machine Part 4: Improve Learning for Three-states and Multi-tasks”, *SoutheastCon, 2013 Proceedings of IEEE*, 10.1109/SECON.2013.6567485 (Oral Presentation)

## CHAPTER 2: LITERATURE REVIEW

In this chapter, we review the related work in the literature for GPRAM, low-resolution facial image recognition, error control coding regarding LDPC codes, and digital modulation schemes in communication theory.

### GPRAM and Low-resolution Human Face Recognition

Human brain is created as an intelligent system that integrates various types of sensor information and processes them intelligently. Human brain is capable of executing functions such as intelligence, reasoning and abstract thought in general purpose [25][26][27]. According to one theory, intelligence begins to emerge [28] [29]. From the 1940s, researchers start attempting to understand the strategies the biological system employs, then eventually translate them into machine-based algorithms. Modern computer is developed to handle computational tasks which human brains are not capable of performing with precision and speed [30]. Visionary thoughts from Turing [1] and von Neumann [3] laid the foundation in terms of mathematical principle and engineering structure. During the same period, Shannon laid the foundation of the mathematical measurement of information [4]. Over the last sixty years, telecom engineers have learned how to achieve Shannon limit using coding and detection theory. During the process, researchers have accumulated a wealth of knowledge on how to preserve, transmit, and detect information. Recent publication in [5] [6] generated seven key lessons the author learned in the error control codings, which are listed below

1. There exists some good randomly constructed long codes, which could approach the Shannon limit [31].
2. As long as the Tanner graphs have few small loops, the derived randomly constructed codes



are near optimal [32].

3. Information could be collected and passed around between sub-graphs at low complexity [32][33].
4. Decoding process could be implemented iteratively and could be robust in the iterations to noises and errors [33].
5. In [34], researchers discovered the the connections between iterative decoding and Pearl's belief algorithm [35], as the key tool to process information for Bayesian networks.
6. Neurons and iterative decoding share a similar way of functioning: repetition, random permutation, and non-linear operation [36].
7. Low resolution graph representation and iterative decoding could unify some algorithms and signal processing schemes such as Fourier transform and Kalman filtering problems [37].

Could the best known information transmission/reception mechanism help us to understand intelligent information representation and processing in a biological brain? To answer this question, researchers have spent a considerable amount of time to understand long codes [38], codes on graphs [39] [40] [41] [42] [43] and large scale random ad-hoc networks [44]. They also learned a few lessons from life science, which were summarized in Part 2 [6]. Lei Wei *et.al* have tried many systems [20][21][22], and finally established GPRAM system in [5] [6].

Face recognition algorithms normally contains two major parts, face detection and face identification [45]. Several major human face recognition techniques are reviewed as follows. In the method of eigenfaces in [46] [47] [48] , authors introduced principal component analysis to efficiently represent pictures of faces. And motivated by the technique of Kirby and Sirovich, [49] employed eigenfaces. [49] [50] proposed new methods working on eigenface to eigenfeatures corresponding to face components, such as eyes, noses and mouth. Substantial related work in multimodle

biometrics are recorded in [51] and [52]. The neural network could also help us recognize face images due to its non linearity. Multilayer perceptron [53] and convolutional neural network [54] have been proposed for face detection. [55] described a multi-resolution pyramid structure for face verification. Other major face recognition methods such as Hidden Markov Models(HMMs) [56] [57], geometrical feature matching [58] [59], template matching [60] [61] could also provide a certain high degree of accuracy within a considerable computational time.

Specifically, with the development of mug shots searching, security monitoring and surveillance system, low-resolution face recognition has become a popular topic. Low-resolution face recognition aims to recognize faces from small sizes or poor quality images with varying pose, illumination, expression, etc [62]. Many machine-learning basis recognition system have been developed. For instance, Zhao *et al.* in [64] presented their system could recognize face images with a resolution of  $24 \times 21$  pixels and even get better results on  $19 \times 17$  pixels by using a combination of PCA and LDA (linear discriminant Analysis). Lemieux *et al.* in [65] studied PCA system with AR database, and observed that the minimal resolution of face size is about  $21 \times 16$  pixels. Also, Boom *et al.* in [66] investigated PCA and LDA system with FRGC database, and found the minimal resolution of face size to be  $32 \times 32$  pixels. It is only recently that a handful of authors have started to address the problem of low-resolution face recognition with unconventional forms of super-resolution (SR) [67]. SR algorithms have been investigated to increase the resolution of face images so that higher resolution image will be better for recognition. This approach aims to find the most similar face image from the face image subspace learned by the training face images. These algorithms achieve recognition performance by using super-resolution information and methods without the burden of producing a visually appealing super-resolved result[68] [69] [70] [71]. It is worth mentioning that VIV is not considered nor applied in the studies above.

## Low-Density Parity Check Codes

Low-density parity-check (LDPC) codes as a class of linear block codes, were first introduced by Gallager in his thesis [72] [73] and was rarely considered in the following 20 years. Until 1981, Tanner proposed an extraordinary work [32] which generalized LDPC codes and introduced a graphical representation of LDPC codes, which is currently called Tanner graphs. MacKay and Neal [74] [75] continued Gallager's work in the mid-1990's, discovered the advantages of linear block codes which could process low density parity-check matrices. In this section we will provide the fundamental representations of LDPC codes via parity-check matrices and Tanner graph.

The background of linear block codes is illustrated as follows. A block code is a code within  $k$  bits input,  $n$  bits output. The code is designated as an  $(n, k)$  code. If the input is  $k$  bits, then there are  $2^k$  distinct messages over the field  $GF(2)$ . The  $n$  symbols of each message associated with each input block is called a codeword. For the linear  $(n, k)$  block code, "linear" means any linear combination of any selection of codewords is also a codeword. In linear block codes, each group of codes is associated with two matrices, named Generator Matrix ( $G$  matrix) and Parity Check Matrix ( $H$  matrix).

There are several widely used examples of block codes. One of them is Hamming code. Hamming Code was proposed in 1947 by Dr. Hamming, who first introduced the idea of error-correcting codes [76]. As a family of linear error-correcting codes, Hamming codes could detect up to two-bit errors or correct one-bit errors without detection of uncorrected errors.

A low-density parity-check code is considered as a linear block code [36] for which the parity-check matrix  $H$  has a low density of 1's. Fig. 2.1 indicates a Tanner graph representation of LDPC

codes, representing a  $5 \times 10$  parity check matrix  $H$ :

$$H = \begin{pmatrix} 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 1 \end{pmatrix} \quad (2.1)$$

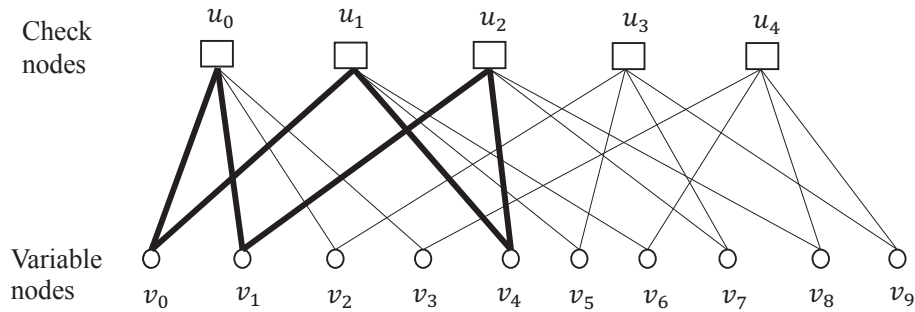


Figure 2.1: A Tanner Graph Representation for Example Code

There are two types of nodes in a Tanner graph which are variable nodes and check nodes. Check node  $u$  is connected to variable node  $v$  whenever element  $h_{vu}$  in  $H$  is 1. So there are  $m = n - k$  check nodes, one for each check equation, and  $n$  variable nodes, one for each code bit  $c_i$ . If a variable node is constrained by a check node, there is an edge connecting these two nodes.

The codeword  $c$  is a binary column vector with length of  $n$  which satisfies the parity check equation as:

$$Hx = 0 \quad (2.2)$$

We call the graph a regular LDPC code if the parity-check matrix contains exactly  $d_c$  1's in each column, exactly  $d_s$  1's in each row, otherwise it is an irregular LDPC code. The low density means that the number of 1's in each row and column of  $H$  is small compared to the block length  $n$  [77].

Cycles refer to a finite set of connected edges which start and end at the same node, with the restrictions that no node appears more than once [78]. In Fig. 2.1, there is a length of 6 cycles which are specified by the bold edges. The girth represents the length of a smallest cycle in the graph. We are interested in the short cycles, since they degrade the performance of the iterative decoding algorithm used for LDPC codes [79]. Sum-product decoding, as known as belief propagation in the AI community [77], will introduced in details in this dissertation.

## Digital Modulation Schemes

In recent years, communication systems are widely used in the daily life. In video, audio or any information in the form of electrical signal is turned as data then transferred between two or more points [80]. Depending on the nature of the communication channel, data can suffer from one or more channel impairments such as noise, attenuation, distortion, fading and interference. And our goal is to reliably transmit these data to the destination by using the given channel. Therefore, researchers are considering generating a signal which represents the binary data as well as matches the characteristics of the channel. The process of communication systems could deal with the development of new encoding techniques, modulation techniques, possibilities for newer transmission channels and the demodulation and decoding techniques. The process of mapping a digital sequence to signals for transmission over a communication channel is called digital modulation.

In digital communications, digital data is usually transmitted in the form of a stream of binary data. There are various digital modulation schemes. The basic types of digital modulation scheme

are Amplitude Shift Keying (ASK), Frequency Shift Keying (FSK) and Phase Shift Keying (PSK) respectively [81][82][83]. In this dissertation, we focus on the PSK modulation, which derive other modulation schemes as DPSK, QPSK and MPSK. In many literatures, digital communication systems for band-limited channel such as optimized  $M$ -ary PSK,  $M$ -ary DPSK, dual-ring QAM signaling with equiprobable symbols, and white, Gaussian noise channel have been well analyzed [80][84]. The optimal signal, optimal constellation, and optimal receiver have been used widely in practice.

Recently, researchers began to think of accommodating the practical needs of non-equiprobable symbols [85] [86]. If the symbols produced by the source are not equiprobable, source coding is applied to convert the source into equiprobable symbols first [80] [84], then applied to channel coding to protect data. We shall call this conventional system. In the past, various issues related to joint source and channel coding (JSCC) have been studied [87]. In practices, due to the sub-optimality of the compression scheme, the bit stream often exhibits a certain amount of redundancy [88] [89]. Furthermore, in wireless sensor network, sensor device may not be able to perform complex source coding computation due to limitation in hardware and power consumption [90]. With these redundancies, the symbol probabilities are not equal [91]. On the other side in the biological systems, the information encompasses non-equal probabilities of occurrence for different symbols (events) [92]. The training and learning behavior of human beings and animals in the nature would be influenced by these non-equiprobable events, which attracts us focusing on the system with non-equal probabilities.

Instead of using source coding, Korn *et.al* have considered to optimize the systems with non-equiprobable symbols [93] [94], which have not been investigated until very recently largely due to the difficulty of the problem and its solution. In [95] [96] several authors have investigated non-uniform sources for various signaling format. Wei and Korn have extended the work of [93] [94] to ASK/QAM [97] [98] and orthogonal/ bi-orthogonal signals [99]. Recently, exact evalua-

tion for arbitrary 2-D modulation and non-uniform signaling has been analyzed in [100], in which signal constellation is given. However, for MPSK, MDPSK, and dual-ring QAM signals, it is still unknown how to optimize signaling for non-equal probable symbol.

## **CHAPTER 3: PROTOTYPE OF GENERAL PURPOSE REPRESENTATION AND ASSOCIATE MACHINE**

In this chapter, a prototype of GPRAM system is implemented by utilizing Hamming code and short LDPC codes. Based on numerous experiments, the design principles of GPRAM system are well demonstrated. Furthermore, types of learning processes are proposed to improve the performance, which further exhibit the capability of GPRAM for handling multitasks. From the results it is amazing to see many interesting features begin to merge and some of them may have bio-implications which could be explored in the future.

### **Overview GPRAM and Design Principle**

GPRAM, proposed in [5] tries to mimic certain functions in biological brains to represent the outside world and make associations among these representations to solve general purpose problems. GPRAM system focuses on using a versatile approach with hierarchical representation and association structures to do a quick and rough assessment on multitasks.

The whole information-processing system is divided into two parts: the first part is for rough and quick estimates, which will decide what to do and how to do, and the second part is for completing the task with precision and accuracy. GPRAM machine will mimic part of human brain and only focus on quick and rough estimates and decisions, i.e., to carry out the first part of work, and the conventional man-made machine is largely designed to carry out the second part of work, in which scientific and mathematical tools play essential and important roles.

The design procedures for these two parts are very different. For building the conventional machine, each part of the machine should be precisely defined; then we assemble the parts into a



machine that performs special tasks. We call this precision design approach. GPRAM uses a versatile approach, which split into two stages: group design and individual specification. In group designs, we focus on preserving common features in the group. In individual specification, systems will be narrowed down to focus more on some specific tasks. This dissertation only focuses on the group design.

The definition of GPRAM system is illustrated as follows. Fig. 3.1 plots a block diagram of a simple GPRAM example system using LDPC. This system has some sensor inputs (i.e.,  $s_1, s_2$ ), some input/output actions (i.e.,  $a_1, a_2$ ), and some output actions (i.e.,  $a_3$ ). In the conventional system design, we wish to find a precise structure which is optimized for a specific task. For example, how to design a system which achieves logic output of  $a_3$ , given inputs  $s_1, s_2, a_1$ , and  $a_2$ , specified in Table 3.1. Even for reconfigurable systems, we would like use the same hardware, but reconfigure to different precise structure for different tasks.

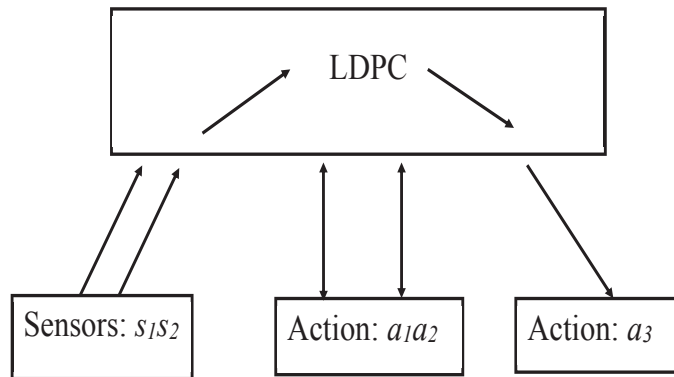


Figure 3.1: Block Diagram of a Simple GPRAM System

In GPRAM, we are more interested in how to design rules and structures which will lead to a group of "good" systems, not one perfect system. This is because (a) GPRAM is a general purpose system, which may need to perform other different tasks in future; (b) each system could land in different structures and different representation format, which may also be good for other tasks; (c)

it may need to perform many other internal/external functions throughout the process, for example, updating or smoothing its structure.

Table 3.1: Five Tasks in the Test

| Tasks | Input |       |       |       | Output | Special Function |
|-------|-------|-------|-------|-------|--------|------------------|
|       | $s_1$ | $s_2$ | $a_1$ | $a_2$ | $a_3$  | Switch           |
| 1     | 1     | 1     | 0     | 1     | 0      | ON               |
| 2     | 1     | 0     | 0     | 1     | 1      | ON               |
| 3     | 1     | 1     | 1     | 0     | 0      | OFF              |
| 4     | 0     | 0     | 0     | 1     | 0      | ON               |
| 5     | 0     | 0     | 1     | 0     | 0      | OFF              |

Therefore the problem is narrowed down to a simple example using (7,4) Hamming code, which is shown in Fig. 3.2. A Tanner graph is used here to represent the code, in which  $v_s$  are variable nodes and  $c_s$  are check nodes. There is a special function unit, called the Switch, which will be further introduced in the later sections.

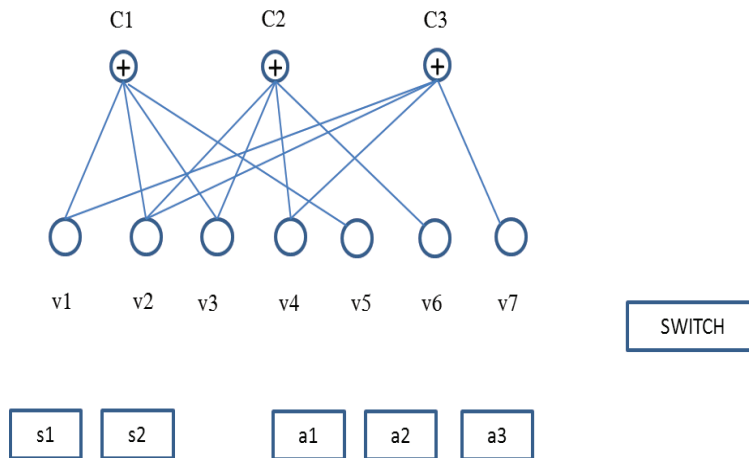


Figure 3.2: A Simple Example of GPRAM System

Now the problem is specified as how to find good rules and structures to approximately implement

the logics in Table 3.1.

### Implementation of GPRAM

The entire process is divided into two stages: (a) to establish connections between  $v_s$  and  $\{s_1, s_2, a_1, a_2, a_3\}$ ; (b) to perform the task. In the following subsections, we will define rules to make connections and operations. Each individual implementation of GPRAM is specified by its connection pattern. Its behavior will also be determined by the connection pattern. We do expect behavior differences among each individual.

#### *Define Operations*

Assume the smallest operation time unit as the iteration number, for example, each iteration requires 1 *ms*.

Before any connections established between  $v_s$  and  $\{s_1, s_2, a_1, a_2, a_3\}$ ,  $u_k^l = 2b_k^l/\sigma_2^2$  are generated as log-likelihood ratio (LLR) of variable node  $k$ , where  $b_k^l = n_k^l$ ,  $n_k^l$  is an independent Gaussian noise of zero mean and variance of  $\sigma_2^2 = 1$ , and  $l$  is iteration number. The noises at iterations are random and independent. Sum-product algorithm is implemented to update the message as conventional iterative decoding, i.e.,

$$v_{km}^{(l)} = \begin{cases} u_k^{(l)} & l = 0 \\ u_k^{(l)} + \sum_{\substack{q=1 \\ q \neq m}}^{d_v} u_{qk}^{(l)} & l > 0 \end{cases} \quad (3.1)$$

$$\tanh \frac{u_{mk}^{(l)}}{2} = \prod_{\substack{q=1 \\ q \neq k}}^{d_c} \tanh \frac{v_{mq}^{(l-1)}}{2} \quad (3.2)$$

where  $v_{km}^{(l)}$  is the LLR from the variable node  $k$  to the check node  $m$ ,  $u_{mk}^{(l)}$  is the LLR from check node  $m$  to the variable node  $k$ .  $d_v$  and  $d_c$  are the degrees of variable and check nodes, respectively. At the end of each of iteration, we make the decision of each variable node as

$$c_k^{(l)} = \begin{cases} 1 & v_k^{(l)} > 0 \\ 0 & v_k^{(l)} < 0 \end{cases} \quad (3.3)$$

where  $v_k^{(l)} = u_k^{(l)} + \sum_{q=1}^{d_v} u_{qk}^{(l-1)}$ . After a number of iterations, (say  $I_a$ ), we make a further process based on comparison among  $c_k^{(l)}$ s and values of  $\{s_1, s_2, \dots\}$  and a certain set of rules.

After the connections between  $\{s_1, s_2, a_1, a_2, a_3\}$  and  $v_s$  are established, for those nodes connected to input signals  $\{s_1, s_2, a_1, a_2\}$ ,  $u_k^l = 2b_k^l/\sigma_2^2$  are generated as log-likelihood ratio (LLR) of variable node  $k$ , where  $b_k^l = a_k^l + n_k^l$  and  $a_k^l = (2z - 1)B$ ,  $z$  denotes the input logic, and finally  $B$  is a constant (e.g., 5). For example, if  $s_1$  is connected to  $v_5$  and  $s_1^l = 0$ , then  $b_5^l = -B + n_5^l$ . The remaining operation is identical to those in (1) and (2). The output value of  $a_3$  at iteration  $l$  is equal to  $c_k^l$ , where  $a_3$  is connected to  $v_k$ .

### *Define Connection Rules*

Types of rules, each with different characteristics, can be used to make connections. Here, we present one of them. We split the connection stage into three steps. At the first step, we make connections of  $s_1, s_2$ ; then  $a_1, a_2$ ; finally  $a_3$ .

**Step 1:** Initialize the system by setting all  $v_{km}$  and  $u_{mk}$  to zero. Pick up Task 1, so  $s_1 s_2 = 11$ . Since no connection has been established, we have  $b_k^l = n_k^l$  for  $k=1, \dots, 7$ . We update  $b_k^l$  according to 4.1 and 4.2 for each iteration. After 30 iterations, we get a 30 by 7 table of  $c_k^l$ . Each column is

corresponding to one variable node. We count the numbers of "1" in each column, and connect  $s_1 s_2$  to two columns with largest numbers of "1" in the table, i.e., two locations of variable nodes where  $s_1, s_2$  connect. In one individual system,  $s_1 s_2$  has connected to  $v_1 v_2$ , which is illustrated on the blue broken line in Fig. 3.3.

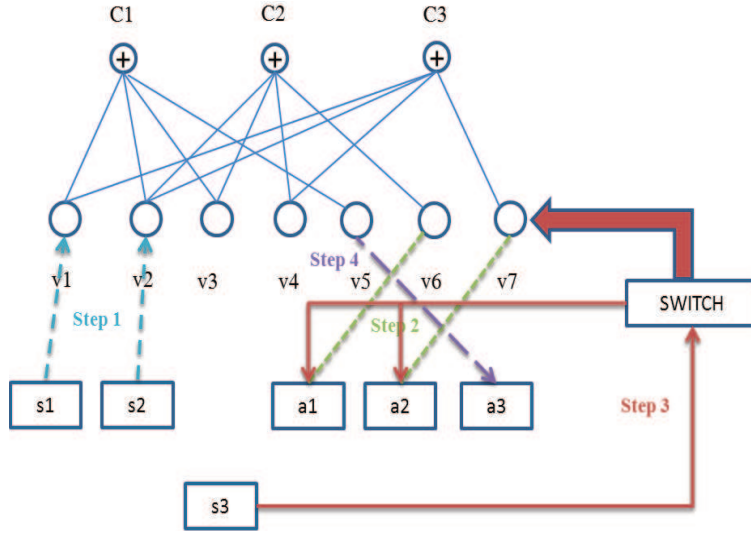


Figure 3.3: Connections of One Individual GPRAM System

**Step 2:** In this step,  $a_1 a_2$  will connect to another two variable nodes. Firstly, we initialize the system by setting all  $v_{km}$  and  $u_{mk}$  to zero, but maintain the connections established in **Step 1**. In Table 3.1, when Task 1 is selected,  $s_1 s_2 a_1 a_2 = 1101$ . Now, we have  $b_k^l = B + n_k^l$  for two variable nodes connected to  $s_1 s_2$ , and  $b_k^l = n_k^l$  for the remaining variable nodes. Again, run for 30 iterations and generate a 30 by 7 table of  $c_k^l$ . Within the remaining 5 unconnected variable nodes, we connect  $a_1$  to the column which has the most of "0"; and  $a_2$  to the column which has the most of "1". Again, one GPRAM connects  $a_1$  to  $v_6$  and  $a_2$  to  $v_7$  as illustrated in Fig. 3.3.

**Step 3:** In this step,  $a_3$  will connect to one variable node selected from the remaining unconnected nodes. Firstly, we initialize the system by setting all  $v_{km}$  and  $u_{mk}$  to zero, but maintain the connec-

tions established in *Step 1* and *Step 2*. This time, instead of inputting one task (in *Step 1* and *Step 2*), we input two tasks. That is, during the first 10 iterations, we have  $\{s_1, s_2, a_1, a_2, a_3\} = \{1, 1, 0, 1, 0\}$  (Task 1), and then switch to  $\{s_1, s_2, a_1, a_2, a_3\} = \{1, 0, 0, 1, 1\}$  (Task 2) for the next 20 iterations. We have  $b_k^l = (2z_k^l - 1)B + n_k^l$  for four variable nodes connected to  $s_1, s_2, a_1, a_2$ ,  $b_k^l = n_k^l$  for the remaining three variable nodes. After the overall 30 iterations, we generate a 30 by 7 table of  $c_k^l$ . Within the remaining 3 unconnected variable nodes, we connect  $a_3$  to the column which has the largest number of iterations that  $a_3$  is equal to  $c_k^l$ . Again, one GPRAM connects  $a_3$  to  $v_5$  as illustrated in Fig. 3.3.

Therefore,  $\{s_1, s_2, a_1, a_2, a_3\}$  has all successfully connected to the variable nodes. So during each trial, messages (i.e., some forms of a posterior probability) are passing between variable nodes and check nodes many times. Furthermore, due to the noise,  $\{s_1, s_2, a_1, a_2\}$  will connect to different locations. For those unused variable nodes, they are free to connect other sensors or action nodes in future at this stage.

#### *Perform the Task Using Hamming (7,4) code*

In the last part, GPRAM has made connections based on Tasks 1 and 2, which are trained tasks and others are untrained tasks. In this part we want to verify that for the trained tasks, GPRAM has a high probability playing the same action agreed with Table 3.1. After connections,  $\{s_1, s_2, a_1, a_2\}$  will be set to different values separately according to tasks in Table I. Firstly, we initialize the system by setting all  $v_{km}$  and  $u_{mk}$  to zero. Pick up Task 1, where  $\{s_1, s_2, a_1, a_2\} = \{1, 1, 0, 1\}$ . Now we have  $b_k^l = (2z_k^l - 1)B + n_k^l$  for four variable nodes connected to  $s_1, s_2, a_1, a_2$ , and for the remaining variable nodes. Again, in each trial, we run 30 iterations and generate a 30 by 7 table of and we record the value of  $a_3$  in the last iteration. Then, we perform 100 trials and summarize the probability of  $a_3 = 0$  or not in Task1. We run the same process for Tasks 2 to 5 and count the

numbers of  $a_3 = 1$  or  $a_3 = 0$ .

We do the same test for 50 different GPRAM connections. Based on the trained Tasks, we classify all the connections into three typical kinds: good connection, middle connection and bad connection. Table 3.2 shows the typical examples of the three kinds. Here, we define a good connection as 90% or more agreeing with the output of  $a_3$ , a middle connection is 70%-90% and a bad connection is 70% or below respectively (comparing rows 1 to 4 in Table 3.2). Regarding rows 5 to 10, since there is no training for these tasks, it is not surprised that  $a_3$  will be 0 or 1 randomly. For some tasks, the sum of number of  $\{s_1, s_2, a_1, a_2, 1\}$  and  $\{s_1, s_2, a_1, a_2, 0\}$  is not equal to 100, because during some trials, iterative decoding ( $c_k^{30}$  s) does not converge to  $\{s_1, s_2, a_1, a_2\}$ . We discard these trials.

Table 3.2: Examples for Three Typical Performance

| $s_1 s_2 a_1 a_2$ | $a_3$ | Good (>90%) | Middle(70% - 90%) | Bad (<70%) |
|-------------------|-------|-------------|-------------------|------------|
| 1101              | 1     | 6           | 23                | 50         |
|                   | 0     | 93          | 76                | 50         |
| 1001              | 1     | 98          | 70                | 37         |
|                   | 0     | 2           | 27                | 60         |
| 1110              | 1     | 72          | 56                | 50         |
|                   | 0     | 26          | 43                | 48         |
| 0001              | 1     | 4           | 54                | 52         |
|                   | 0     | 95          | 45                | 44         |
| 0010              | 1     | 65          | 60                | 56         |
|                   | 0     | 31          | 39                | 41         |

*Perform the Task Using Other LDPC Codes*

Now, let us use different LDPC codes, mainly by expanding the length of the code word. We use two codes of rate equal to 1/2 with lengths of (8,4) and (10,5). The parity matrices are shown in

Fig. 3.4. We also use randomly generated regular rate 1/2 (20,10) LDPC codes with (6,3) degrees of variable and check nodes. We repeat those processes described in the last section for (8,4) and (10,5) codes. For (20,10) codes, in each trail, we will randomly construct a code, but within the trial, the code is fixed. Now, we focus on how many trials are in the ranges of good, middle, and bad connections, illustrated in Figure 5 and compared with the (7,4) Hamming code.

$$H_{(8,4)LDPC} = \begin{bmatrix} 1 & 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

$$H_{(10,5)LDPC} = \begin{bmatrix} 1 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

Figure 3.4: Parity Check for (8, 4) and (10, 5) LDPC codes

In Fig. 3.5, the results show that for (7,4) Hamming code, it has 82% probability in the good range, 12% in the middle range, and 6% in the bad range. When increasing the length of codes, it becomes harder and harder to get a good code and a good connection. The probability of getting good connections drops from 72% for (8,4) codes, to 34% for (10,5) codes, and to 0 for (20,10) codes. Thus, the longer the code is, the more difficult to train the system to get good connections.



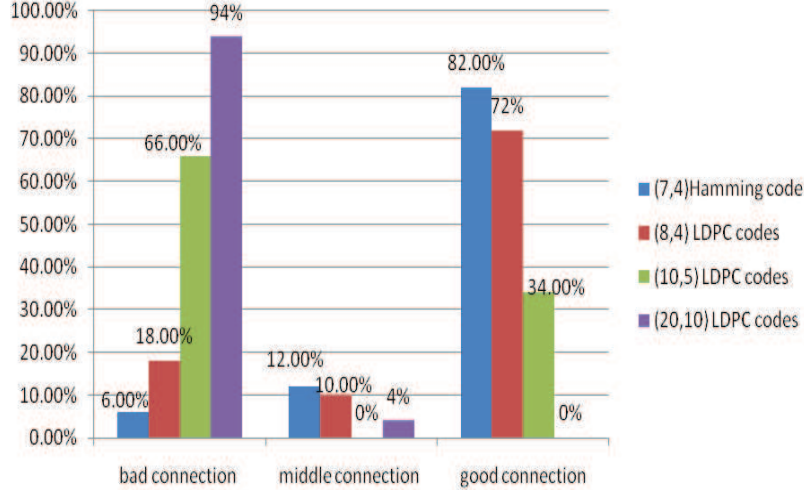


Figure 3.5: Training Result for Random Codes

### GPRAM System with Switch

As we can see from previous section, if we fixed the code word structure, those GPRAMs with bad connections would not be able to improve its behavior. In bio-systems, it has been widely believed that during the sleeping mode, the brain disconnects the sensor inputs and performs revision of internal structures. Could we implement similar function in our prototype? In this section, we present a solution to disconnect the sensor inputs. In our solution defined in the previous section, we make hard decision for the decoding part (see (4.3)). Let us use three states to describe, 0 or 1, or  $X$ , where  $X$  denotes uncertain. We select parameters as follow

$$c_k^{(l)} = \begin{cases} 1 & v_k^{(l)} \geq 3 \\ X & -3 < v_k^{(l)} < 3 \\ 0 & v_k^{(l)} \leq -3 \end{cases} \quad (3.4)$$

where three-states to describe  $c_k^{(l)}$ , are 0 or 1, or  $X$ , and  $X$  denotes uncertain. After a number of iterations, (say  $I_a$ ), we make connections based on comparison among  $c_k^{(l)}$ s and values of  $\{s_1, s_2, a_1, a_2, a_3\}$  and a certain set of rules.

There are many reasons that the uncertain state is needed. Here we only focus on how to block the sensor inputs. That is, no matter the values of  $[s_1, s_2, a_1, a_2]$ , the system should converge to uncertain states for most of  $c_k^{(l)}$ . In error control coding, particularly iterative decoding, we can increase the noise density  $\sigma_2^2$  to achieve this switch function. That is, during the OFF state, we increase noise power to bring the system into this uncertain state, so the sensor input would be blocked at this layer.

#### *Perform the Task Using Hamming (7,4) code*

We follow the same procedures described in the previous section but without Switch Unit to construct the processing system. After adding the switch and making connection,  $\{s_1, s_2, a_1, a_2\}$  will be set to different value separately according to tasks in Table 3.1. First, we initialize the system by setting all  $v_{km}$  and  $u_{mk}$  to zero. Pick up Task 1, where  $\{s_1, s_2, a_1, a_2\}=\{1,1,0,1\}$ . Again, in each trial, we run 30 iterations and generate a  $30 \times 7$  table of and we record the value of  $a_3$  in the last iteration. Then, we perform 100 trials and summarize the probability of  $a_3=0$  or not in Task 1. We run the same process for Tasks 2 to 5 and count the numbers of  $a_3 = 1$  or  $a_3 = 0$ . When  $a_1 a_2=01$  (machine in ON state), we can inject the light noises ( $\sigma_{on}^2 = 1$ ) as  $n_k^l$ ; when  $a_1 a_2=10$  (machine in OFF state) heavy noises ( $\sigma_{off}^2 = 4$ ) could push the machine into the uncertain state.

We do the same test for 50 different GPRAM connections. Based on the trained tasks, we classify all the connections into three typical kinds, good connection (defined as 90% or more agreeing with the output of  $a_3$ ), middle connection (70% to 90%), bad connection (70% or less). Among the 50 different GPRAM connection trials, many of them fail to make valid connection due to

state  $X$ . We leave these sets in the bad connection category. Even though it established a valid connection, during operation iterative decoding ( $c_k^{30}$ s) may not converge to  $\{s_1, s_2, a_1, a_2\}$  and we have to discard these trials. For some tasks, the sum of numbers of  $\{s_1, s_2, a_1, a_2, 1\}$ ,  $\{s_1, s_2, a_1, a_2, 0\}$  and  $\{s_1, s_2, a_1, a_2, X\}$  is not equal to 100. Table 3.3 shows three typical examples. For  $\{s_1, s_2, a_1, a_2\}=\{1, 1, 1, 0\}$ ,  $\{0, 0, 1, 0\}$ , we can see most of 100 trials have been discarded. For  $\{s_1, s_2, a_1, a_2\}=\{0, 0, 0, 1\}$ , since there is no training for this task, it is not surprising that  $a_3$  will be 0 or  $X$  or 1 randomly (case by case).

Table 3.3: Three Typical Cases for Three-states

| $s_1s_2a_1a_2$ | $a_3$ | Good (>90%) | Middle(70% - 90%) | Bad (<70%) |
|----------------|-------|-------------|-------------------|------------|
| 1101           | 1     | 0           | 0                 | 12         |
|                | X     | 5           | 11                | 23         |
|                | 0     | 93          | 76                | 50         |
| 1001           | 1     | 98          | 70                | 37         |
|                | X     | 2           | 5                 | 23         |
|                | 0     | 2           | 27                | 60         |
| 1110           | 1     | 72          | 56                | 50         |
|                | X     | 1           | 8                 | 1          |
|                | 0     | 26          | 43                | 48         |
| 0001           | 1     | 4           | 54                | 52         |
|                | X     | 3           | 14                | 32         |
|                | 0     | 95          | 45                | 44         |
| 0010           | 1     | 65          | 60                | 56         |
|                | X     | 7           | 12                | 2          |
|                | 0     | 31          | 39                | 41         |

*Perform the Task Using Other Codes*

Now, let us use different LDPC codes, mainly by expanding the length of the code word. We use two codes of rate equal to 1/2 with lengths of (8,4) and (10,5) (see [8]). We also use randomly generated regular rate 1/2 (20,10) LDPC codes with (6,3) degrees of variable and check nodes. We

repeat those processes for (8,4) and (10,5) codes. For (20,10) codes, in each trial, we will randomly construct a code, but within the trial, the code is fixed. In Fig. 3.6, we focus on how many trials are in the good, middle, and bad connections, comparing with the results using (7,4) Hamming code. Comparing Fig. 3.6 with Fig. 3.5, we can see the results for long codes are far worse. The rest of chapter is devoted to how to improve good connections and how to learn multitasks.

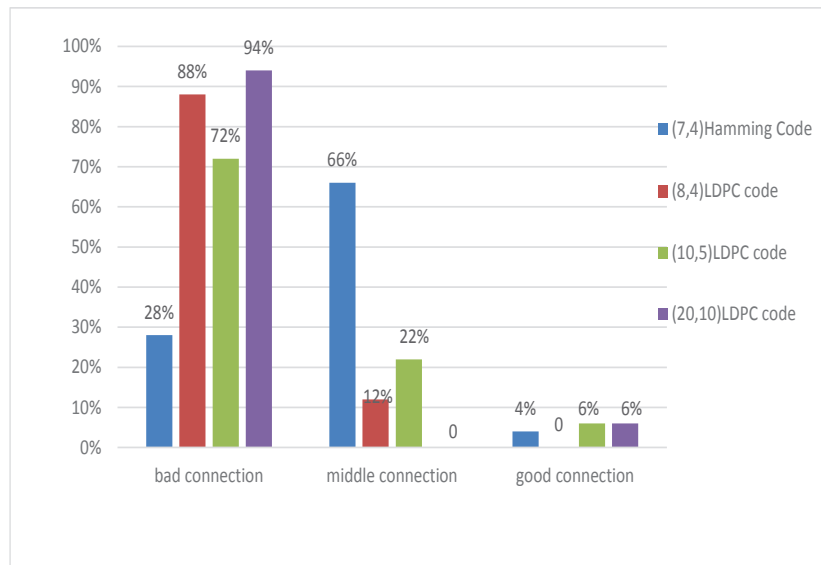


Figure 3.6: Training Results for Different Codes Considering Three-state

### Improvement in Connections and Operations

During the learning steps in the previous section, since only input of signal is the noise (no connection in Step 1 and two connections in Step 2), the values of  $c_k^l$  are almost always at state  $X$ . So, the modification in connection **Steps 1-3** is (a) to generate 2000 by 7 table of  $x_k^l$  (b) to connect to the column which has the largest difference between the numbers of "1" and "0". For example,  $s_1=1$ , so it computes the difference between the numbers of "1" and "0" for each column and connects

to the one with the largest difference.

Up to now, we only focused on the result at 30th iteration. With the uncertain state, the decoder could switch between 0 and  $X$  (often), or 1 and  $X$  (often), or 0 and 1 (rarely). In order to handle these uncertainties, the modification in operation uses the following averaging methods. Let us define an intermediate variable  $h_k^l$  by replacing  $c_k^l$  in (3), then redefine  $c_k^l$  as follows.

$$c_k^l = \begin{cases} 1 & \text{if } (h_k^{l-K+1}, h_k^{l-K+2}, \dots, h_k^l) \text{ meets Cond1} \\ 0 & \text{if } (h_k^{l-K+1}, h_k^{l-K+2}, \dots, h_k^l) \text{ meets Cond0} \\ X & \text{Otherwise} \end{cases} \quad (3.5)$$

where

$$h_k^l = \begin{cases} 1 & v_k^l \geq 3 \\ X & -3 < v_k^l < 3 \\ 0 & v_k^l \leq -3 \end{cases} \quad (3.6)$$

We can see the decision of 0 or 1 depend on a rule book and a window size of  $K$   $\{h_k^l\}$  s. Now, let us define a rule books (**RB**). **RB**:  $K=30$ , we will make a decision of 0 or 1, if in the window of  $\{h_k^l\}$ s there are more than T of 0 or 1, respectively. For example, if within 30 times of  $\{h_k^l\}$  s there are 18 of 0, 2 of 1, and 10 of  $X$ , then we will make a decision of 0 (say  $T=15$ ); if within 30 times of  $\{h_k^l\}$ s there are 14 of 0, 6 of 1, and 10 of  $X$ , then we will make a decision of  $X$  (say  $T=15$ ). Now, we can repeat what we did in the last Section.

From the results in Table 3.4 and Fig. 3.7, we can see that (a) switch off-state has been completely shut off; (b) dramatically improves the learning using short code, but for long codes, the effectiveness is still not very good.

This is due to two problems: (i) for long codes, it is hard to converge to a valid codeword when it

starts; (ii) if one input connects to one variable node, 4 inputs can only control 4 out of 20 variable nodes. In the next section, we will improve this using multiple connection and perfect codewords.

Table 3.4: Three Typical Cases for RB

| $s_1s_2a_1a_2$ | $a_3$ | Good (>90%) | Middle(70% - 90%) | Bad (<70%) |
|----------------|-------|-------------|-------------------|------------|
| 1101           | 1     | 0           | 0                 | 0          |
|                | X     | 0           | 19                | 32         |
|                | 0     | 100         | 81                | 68         |
| 1001           | 1     | 100         | 60                | 67         |
|                | X     | 0           | 40                | 33         |
|                | 0     | 0           | 0                 | 0          |
| 1110           | 1     | 0           | 0                 | 0          |
|                | X     | 0           | 0                 | 0          |
|                | 0     | 0           | 0                 | 0          |
| 0001           | 1     | 100         | 0                 | 0          |
|                | X     | 0           | 31                | 50         |
|                | 0     | 0           | 69                | 50         |
| 0010           | 1     | 0           | 0                 | 0          |
|                | X     | 0           | 0                 | 0          |
|                | 0     | 0           | 0                 | 0          |

### Improvement in Learning Using Perfect Codeword

From Fig. 3.7, we could summarize for long codes, such as (20,10) LDPC codes, it becomes more difficult to get good connection. This is largely due to two limitations, (a) during the initial step (i.e., Step 1), decoded  $c_k^l$  may not form a valid codeword; (b) after connection, since we only allow one input to connect one variable node, only small fractional variable nodes are controlled by inputs. So, we can introduce mechanisms to overcome these limitations. We propose learning procedures using Perfect Codewords and with Multiple Connection (PCMC) for two tasks first, then for three tasks.

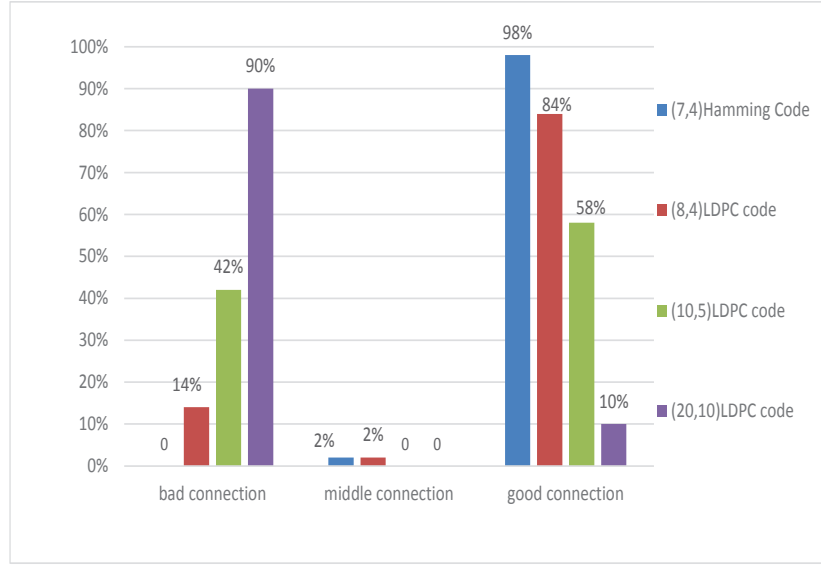


Figure 3.7: Training Results for Different Codes Using RB

### *Two-task Learning Using Perfect Codeword and Multiple Connection*

In Step 1, instead of generating  $c_k^l$ s, we randomly pick up two codewords generated from LDPC codes. For example, two codewords of a (20,10) code are  $u_1=[00000100110100111100]$ ;  $u_2=[00000101010111000011]$ . Then according to Task1  $s_1s_2a_1a_2a_3=[11010]$ , Task2  $s_1s_2a_1a_2a_3=[10011]$  in , we make  $s_1s_2a_1a_2a_3$  separately connect to multiple variable nodes based on each column of these two codeword. So in our example  $s_1(11)$  will connect to  $[v_6, v_{10}]$ ,  $s_2(10)$  will connect to  $[v_9, v_{15}, v_{16}, v_{17}, v_{18}]$ ,  $a_1(00)$  will connect to  $[v_1, v_2, v_3, v_4, v_5, v_7, v_{11}]$ ,  $a_2(11)$  will connect to  $[v_{12}]$ , and  $a_3(01)$  will connect to  $[v_8, v_{13}, v_{14}, v_{19}, v_{20}]$ .

After connection, we repeat using **RB**. Then, we implement the whole process for 50 times by randomly choosing two different codewords every time. The results are presented in Table 3.5 and Fig. 3.8 , which show dramatically improvement in learning for (20,10) and (40,20) LDPC codes.

Table 3.5: Example of (20,10) LDPC Codes Using PCMC in Two Tasks

| $s_1s_2a_1a_2$ | $a_3$ | Good (>90%) | Middle(70% - 90%) | Bad (<70%) |
|----------------|-------|-------------|-------------------|------------|
| 1101           | 1     | 0           |                   | 55         |
|                | X     | 0           |                   | 0          |
|                | 0     | 100         |                   | 45         |
| 1001           | 1     | 100         |                   | 54         |
|                | X     | 0           |                   | 0          |
|                | 0     | 0           |                   | 46         |
| 1110           | 1     | 0           |                   | 0          |
|                | X     | 0           |                   | 0          |
|                | 0     | 0           |                   | 0          |
| 0001           | 1     | 0           |                   | 0          |
|                | X     | 68          |                   | 0          |
|                | 0     | 14          |                   | 0          |
| 0010           | 1     | 31          |                   | 14         |
|                | X     | 6           |                   | 46         |
|                | 0     | 0           |                   | 26         |

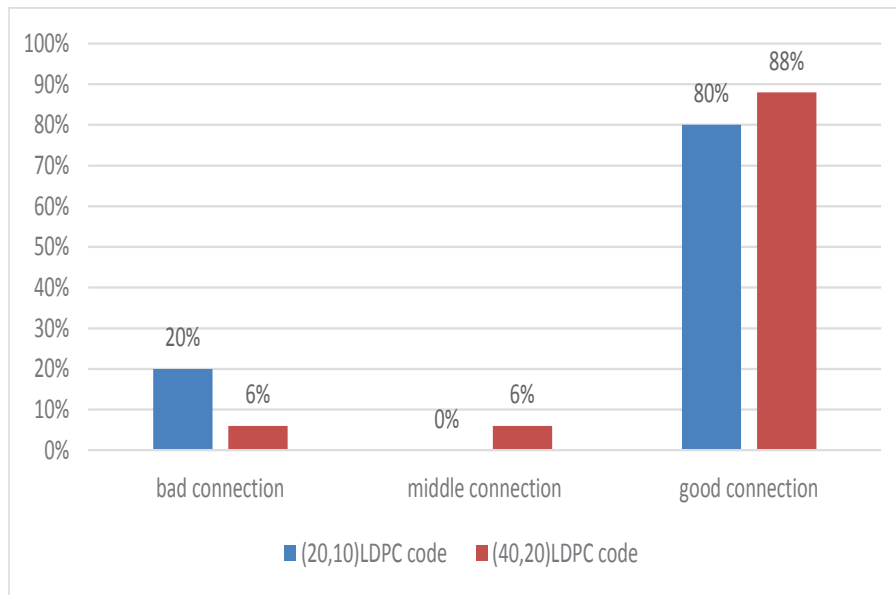


Figure 3.8: Training Results for Different Codes Using PCMC in Two Tasks Training



Table 3.6: Example of (20, 10) LDPC Codes Using PCMC in Three Tasks

| s1s2a1a2 | a3 | Good (>90%) | Middle(70% - 90%) | Bad (<70%) |
|----------|----|-------------|-------------------|------------|
| 1101     | 1  | 0           | 0                 | 0          |
|          | X  | 0           | 0                 | 27         |
|          | 0  | 100         | 91                | 33         |
| 1001     | 1  | 100         | 87                | 41         |
|          | X  | 0           | 0                 | 32         |
|          | 0  | 0           | 0                 | 0          |
| 1110     | 1  | 2           | 0                 | 0          |
|          | X  | 1           | 0                 | 0          |
|          | 0  | 0           | 0                 | 0          |
| 0001     | 1  | 0           | 0                 | 0          |
|          | X  | 0           | 0                 | 100        |
|          | 0  | 100         | 0                 | 0          |
| 0010     | 1  | 3           | 81                | 0          |
|          | X  | 0           | 0                 | 0          |
|          | 0  | 0           | 0                 | 0          |

*Three-task Learning Using Perfect Codeword and Multiple Connection*

Our ultimate goal for GPRAM is to learn multi-tasks, not limited to two tasks. In our consideration, one way to implement it is to randomly choose three different codewords, and make multiple connections based on each column of these three codewords (five patterns,  $s_1(110)$ ,  $s_2(100)$ ,  $a_1(000)$ ,  $a_2(111)$ ,  $a_3(010)$ ). In fact, due to the limitation of multi-tasks, variable nodes are not totally connected. Then we repeat what we did in the previous subsection to obtain. From Table 3.6 and Fig. 3.9, it is clear to see more than 90% of system can fall back to bad connection, because very few variable nodes satisfy the above five patterns.

## Improvement in Multitask Learning Using Progressive Learning

In this section, a progressive learning method is introduced to improve multitask learning results (PLMT).

Since progressive learning requires large code space, here only (40,20) LDPC codes are evaluated. For (40,20) codes, the first 20 bits are systematic (or message) bits. Once these 20 bits are defined, then we can generate the rest of 20 parity bits.

The progressive learning method is given as follow. At first, two codewords are randomly picked and connections are made in the message bits for the first two tasks (task 1 and task 2). Then, the message bits of the third codeword is generated based on the established connections and the new task (task 4). Finally, we generate parity bits for all three codewords and search for variable nodes which satisfy five patterns to make connection. Table 3.7 and Fig. 3.10 presents the dramatically improvement for multitasks.

### Summary

In this chapter, we build GPRAM prototype by utilizing Hamming code and simple LDPC codes. First of all, we briefly overview GPRAM system and its design principles. Based on our numerous experiments, the design principles of GPRAM system has been demonstrated. Moreover, types of learning processes have been proposed, which address the challenge from the perspective of long codes. Their excellent improvements further prove the capability of GPRAM for handling multitasks.

From the results, it is amazing to see many interesting features begin to merge and some of them may have bio-implications which need to be explored further. Our ultimate goal is to build a robot-

like system which can display behaviors similar to some small animals. Once we reach this stage, then the system can be used as a test dummy to develop, evaluate, and furthermore improve our research tools to study the identical type of systems.

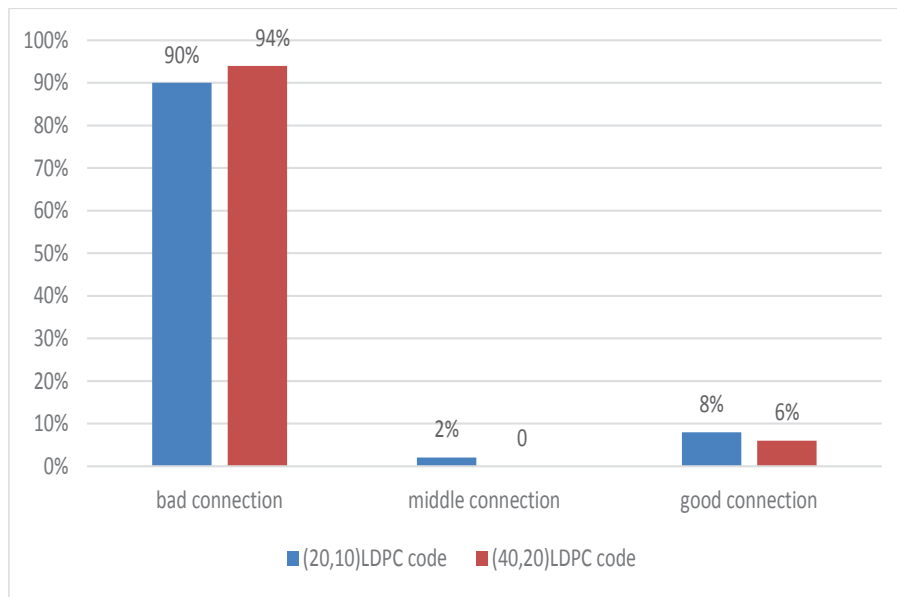


Figure 3.9: Training Results for Different Codes Using PCMC in Three Tasks Training

Table 3.7: Example of (40,20) LDPC Codes Using PLMT

| s1s2a1a2 | a3 | Good (>90%) | Middle(70% - 90%) | Bad (<70%) |
|----------|----|-------------|-------------------|------------|
| 1101     | 1  | 0           |                   | 0          |
|          | X  | 0           |                   | 35         |
|          | 0  | 100         |                   | 19         |
| 1001     | 1  | 100         |                   | 17         |
|          | X  | 0           |                   | 32         |
|          | 0  | 0           |                   | 0          |
| 1110     | 1  | 0           |                   | 3          |
|          | X  | 0           |                   | 2          |
|          | 0  | 0           |                   | 0          |
| 0001     | 1  | 0           |                   | 0          |
|          | X  | 0           |                   | 24         |
|          | 0  | 100         |                   | 19         |
| 0010     | 1  | 3           |                   | 5          |
|          | X  | 0           |                   | 4          |
|          | 0  | 0           |                   | 0          |

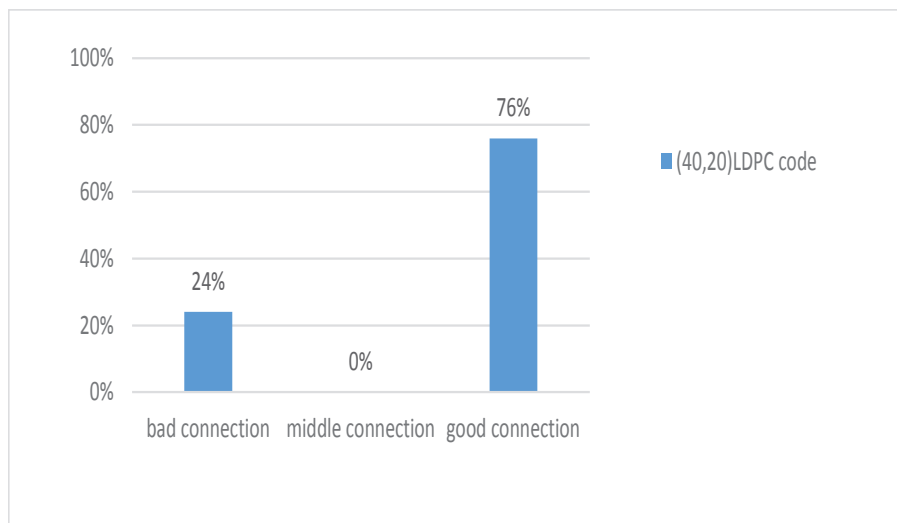


Figure 3.10: Training Result for Different Codes Using PLMT in Three Tasks Training

## **CHAPTER 4: IMAGE PROCESSING UNIT FOR GPRAM FOR RECOGNIZING LOW-RESOLUTION FACE IMAGES WITH VISUAL INFORMATION VARIABILITY**

In the last chapter, we introduced GPRAM concept and implemented GPRAM using simple (7,4) Hamming code and LDPC codes. Meanwhile in [24], we demonstrated that IPU/GPRAM can recognize low-resolution digits (28-by-28 pixels) with visual information variability (VIV) such as point spread, fixation movement, orientation rotation and Gaussian noises. Immediately our attention has shifted toward the first significant milestone, i.e., to see whether our IPU unit can recognize low-resolution facial images with VIV. Again, if not, we then do not need to progress further. This chapter is going to present our IPU can deliver results very close to the best known machine recognizing low-resolution facial images capability, despite of realistic VIV. Keep in mind, the state-of-the-art methods recorded in literature are largely absence of VIV. Compared with the outstanding face recognition of human visual system, there are still notifiable difference. Our work even though may inspire others to continue constructing the hierarchical IPU for GPRAM, ultimately building GPRAM.

### Define Structure of IPU

The original idea employed in this section can be tracked back to our previous work in 2005 [22]. In this section, we articulate IPU/GPRAM structure and establish a testable platform used in the application of face recognition.

In Fig. 4.1, we illustrate a block diagram of the image processing unit for GPRAM prototype. As illustrated in Fig. 4.1, each pixel in any given picture directly connects to each photo-sensor

unit(say,  $s_1, s_2, \dots$ ). Therefore, an image with  $K$ -pixels could be represented by  $K$  sensor units. In general, each pixel is firstly specified by an integer from 0 to 255 then further scaled down and normalized into a range of  $[0, 1]$  by dividing by 255. For simplicity, in our illustration we start elaborating binary images only in black and white color. In later sections, general case will be further discussed.

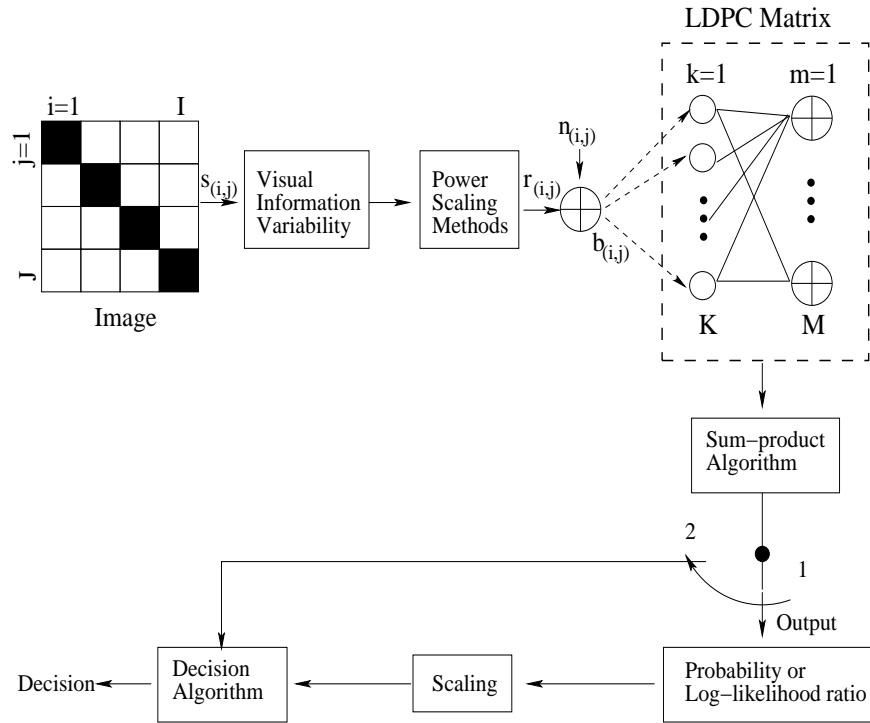


Figure 4.1: Structure of IPU for GPRAM for One Frame

It is shown in Fig. 4.1, each pixel in an image with dimensions of  $I$  by  $J$  could be represented by  $s_{(i,j)}$  or  $s_k$  which locates at the  $i$ -th row  $j$ -th column and  $k = (i - 1) * I + j$ . Furthermore, after going through VIV and power scaling methods (will be discussed in Section 5),  $K$  sensor units will enter into the following unit blocks as: (1) a LDPC matrix model, (2) an iterative sum-product decoding unit, (3) a switch (4) a scaling unit and (5) a comparison decision unit. Specifically, as shown in Fig .4.1, a switch is firstly operated at position 1 to compute a template for  $K$  sensors as

either measured probabilities of variable nodes or log-likelihood ratio (LLR). Then the switch will be operated at position 2, using templates and decoding results to make comparison computation. The dashed box in Fig. 4.1 indicates we are able to remove LDPC codes or not by comparison.

The method to converting sensor inputs to variable node inputs is defined as follows. When without variabilities like VIV and power scaling methods, i.e.,  $\gamma_k = s_k$ , for binary images, we map black to 0 and white to 1 at pixel  $k$ , i.e.,  $s_k = \{0, 1\}$ . Then we map them into  $\{-1, 1\}$ , respectively, and add AWGN noise to generate inputs to variable nodes of LDPC code. We have  $b_k^{(l)} = n_k^{(l)}$  if the variable node  $k$  does not connect to any sensor input.  $b_k^{(l)} = 1 + n_k^{(l)}$  or  $b_k^{(l)} = -1 + n_k^{(l)}$  if the node  $k$  connects to a sensor input (say  $s$ ) and the input value is 1 or 0 (i.e.,  $s = 1$  or 0) respectively.  $n_k^{(l)}$  is a Gaussian random variable with zero mean and variance of  $\sigma_{a1}^2$ , and  $(l)$  is the iteration number. Each iteration uses one frame of information, so  $l$ -th iteration experiences the  $l$ -th frame. Define  $u_k^{(l)} = 2b_k^{(l)}/\sigma_{a2}^2$  as log-likelihood ratio (LLR) of the variable node  $k$ . And for the purposes of robustness [101],  $\sigma_{a2} > \sigma_{a1}$ . The noises at each iteration are random and independent. More importantly, when we extend to images with VIV, energies need to be scaled, which will be shown in details in Section IV. Sum-product algorithm is used to update the message as in conventional iterative decoding, i.e.

$$v_{km}^{(l)} = \begin{cases} u_k^{(l)} & l = 0 \\ u_k^{(l)} + \sum_{\substack{q=1 \\ q \neq m}}^{d_v} u_{qk}(l) & l > 0 \end{cases} \quad (4.1)$$

$$\tanh \frac{u_{mk}^{(l)}}{2} = \prod_{\substack{q=1 \\ q \neq k}}^{d_c} \tanh \frac{v_{mq}^{(l-1)}}{2} \quad (4.2)$$

where  $v_{km}^{(l)}$  is the LLR from the variable node  $k$  to the check node  $m$ ,  $u_{mk}^{(l)}$  is the LLR from check node  $m$  to the variable node  $k$ .  $d_v$  and  $d_c$  are the degrees of variable and check nodes, respectively.

At the end of each of iteration, we make the decision of each variable node as

$$c_k^{(l)} = \begin{cases} 1 & v_k^{(l)} > 0 \\ 0 & v_k^{(l)} < 0 \end{cases} \quad (4.3)$$

where  $v_k^{(l)} = u_k^{(l)} + \sum_{q=1}^{d_v} u_{qk}^{(l-1)}$ . After a number of iterations, (say  $I_a$ ), we make a further process based on comparison among  $c_k^{(l)}$ s and values of  $\{s_1, s_2, \dots\}$  and a certain set of rules.

The above process follows GPRAM principles. Here we outline several aspects.

(1) *Why do we insert noises into clear pictures?* From the point view of GPRAM design, GPRAM system does not view clear picture as a clear picture, but as a noise corrupted sample of internal represented known picture (or previous experienced picture). As soon as GPRAM system converges to previous experienced internal represented picture, it can make association with many previous experiences as well. For example, as we will see in the next section, after we present 1000 noise corrupted samples of a circle patch to this IPU/GPRAM system, it builds its association between the patch and  $c_{739}^{(30)} = 1$ . This association link is established for all possible noisy samples which produce  $c_{739}^{(30)} = 1$  in iterative decoding, not just 1000 experienced samples. Therefore, in future, if the image has the property of  $c_{739}^{(30)} = 1$  after convergence, it is known that IPU system has seen one of more than thousands of noise corrupted samples associated to the patch.

(2) *Why do we use independent Gaussian noise for each iteration?* In traditional communication system, the noise for each variable input is fixed, since decision statistics is fixed. When IPU processes video information, we continuously take input from each pixel which is corrupted by noise processes. Furthermore, in bio-visual systems, the internal noise exists in many visual processing neural layers. For simplicity, we assume the noise is white and independent in this dissertation. In



future, we will relax it to dependant noise.

(3) *Why do we use two different variances in adding the noise (i.e.,  $\sigma_{a1}^2$ ) and in LLR computation (i.e.,  $\sigma_{a2}^2$ )?* In conventional communication systems, noise variance can be estimated at the receiver and then be used to perform iterative decoding. However, in real operation of IPU, the variance could vary at a wide range and estimation may be complicated. So in this dissertation, we would like to find a fixed value of  $\sigma_{a2}^2$  which provides near optimal decoding for a wide range of  $\sigma_{a1}^2$ . In the next section, we will show how to select these two noise variance values.

(4) *Why do we limit maximum value of LLR to  $1e5$ ? That is, the probabilities of the node in any state are no more accurate than 99.999%.* In GPRAM, we should never compare its accuracy, but possible variation capability. During normal operation, each mode may only work at about 60% - 90% accuracy, not 99.9% accuracy as we see in communication system. But in this dissertation, we still need to compare with existing objects and face recognition algorithms. Therefore, we select,  $1e5$ , which allows us to compare up to about 0.1% accuracy.

Even though, the design principles of GPRAM are very different from the machine people build today. Current LDPC codes/decoding, electronic computation and components are not the best choice to build GPRAM system. Our simple setting in this dissertation enables us to seek what this theory can lead us to achieve.

### Understanding IPU Using Simple Images

In this section, we will introduce bit convergence and its functions in pattern recognitions. We start our journey from studying a few simple binary images. At first, we consider the function

of bit convergence in one image. Next, differentiating a group of images in the same shapes but in different sizes is presented. Finally, we differentiate images in different shapes and different sizes. In Fig. 4.1, simple images illustrated in this section are processed following the first two unit blocks of IPU , i.e., (1) a LDPC code, (2) an iterative sum-product decoding unit without implementing VIV and power scaling methods. For the remaining three unit blocks in Fig. 4.1, we will further discuss after introducing VIV and power scaling methods in the following sections.

### *Bit Convergence*

A straightforward example for understanding IPU is illustrated as follows. A black/white circle patch with 32-by-32 pixels is selected. Each black pixel takes value of 0 and white for 1. We randomly generate a (6,3) LDPC codes with 1024 ( $32 \times 32$ ) variable nodes and 512 check nodes, i.e., the parity check matrix  $\mathbf{H}$  with 512 rows and 1024 columns. Each row contains six "1" and each column contains three "1", while the rest elements are all "0".

Using Gaussian elimination method, we can obtain a generator matrix  $\mathbf{G}=[\mathbf{P}, \mathbf{I}]$  and  $\mathbf{GH}^T=0$  using modulo two addition, where  $\mathbf{I}$  is 512 by 512 identity matrix,  $\mathbf{P}$  is 512 by 512 parity check portion of matrix, and superposition  $\mathbf{T}$  denotes matrix transpose. Therefore the first 512 variable bits are parity check bits and the last 512 variable bits are systematic (or called message in error control coding) bits. If we get  $\mathbf{G}_I=[\mathbf{I}, \mathbf{P}_I]$ , then the first 512 variable bits become systematic bits and the last variable bits become parity check bits. If we permute columns, we can make any 512 variable bits as systematic bits and the rest as parity check bits. Given a vector ( $\mathbf{m}$ ) of 1 by 512 message bits, we obtain the code-word of 1024 bits as  $\mathbf{C}=\mathbf{mG}$ .

Now, each variable node of  $\mathbf{H}$  links to a pixel of the circle patch, from top left to bottom right, mapping pixel  $\{(1,1)... (32,32)\}$  to variable nodes  $\{1, ..., 1024\}$ . We input the original perfect image in black and white color, convert black pixels to -1 and white pixels to 1, and then add

AWGN  $n_k^{(l)}$ s of zero mean and variance  $\sigma_{a1}$  of 0.6 to each pixel. Then sum-product algorithm is applied for decoding within 30 iterations each trial implemented. After decoding, it is clear to see some connections between each bit. For instance, in Fig. 4.2, we illustrate a variable node (i.e., #739 bit) and its first tier 15 connected variable nodes (i.e., #36, #364, #469, #564, #809; #124, #416, #507, #724, #994; #65, #358, #457, #665, #787) via 3 check nodes. And from the Tanner graph presented in Fig. 4.2, we can see the code structure of #739 bit, which links to 3 check nodes, then 15 variable nodes (called the first tier), then to 45 check nodes, then to 225 variable nodes (called the second tier). As we carefully check each set of nodes after 30 iterations, all the three sets in the first tier satisfy the parity check constraints. However, in the second tier, about half of 45 sets satisfy constraints and another half do not.

Keep in mind, at initial stage, IPU has no experience at all on visual inputs. So, we choose a randomly constructed (6,3) LDPC code. In the code, the constraints are all XOR logic operations and each is connected to 6 variable nodes. In future, we will include many other logic operations since the constraints imposed in variable nodes may simply follow other logics such as AND, OR, *etc.*. One of the key functions in IPU at this stage is to quickly identify pattern which IPU has experienced. In other words, as soon as a pattern is input into the system, it has some mechanism to indicate the presence of pattern. One of the simplest ways to perform identification is to have an indication bit, just like the 739-th bit.

Now let us describe how to find these essential bits.

### *Simple Examples for Differentiating Circle Images*

In this subsection, we first introduce the concept of survived positions and then evaluate how these positions contribute to pattern recognition.

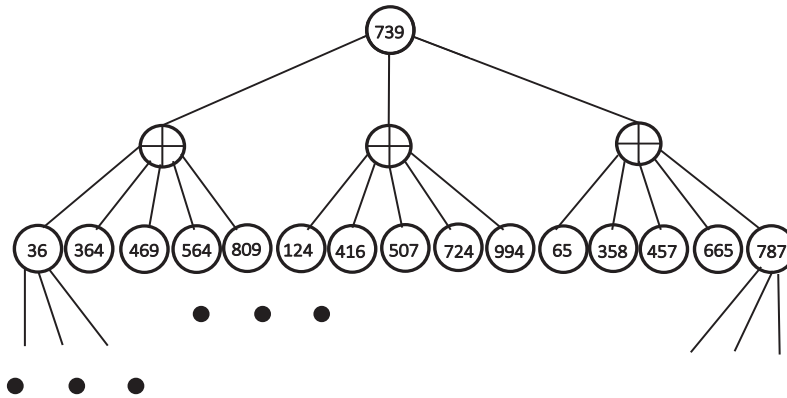
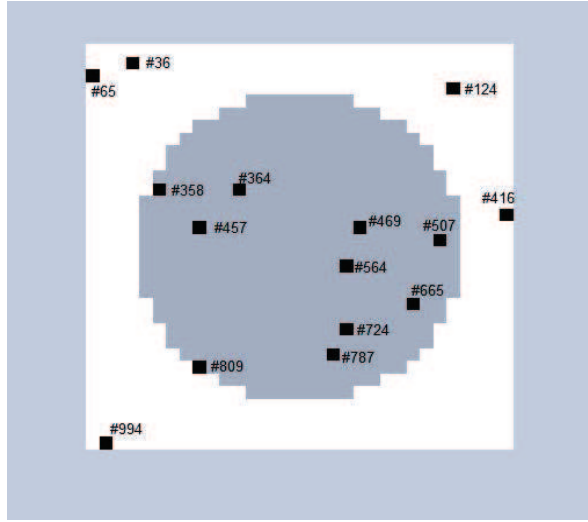


Figure 4.2: Illustration in the Circle Image and in Tanner Graph Structure for the First Tier Connections of the 739-th Position

In each trial, we load one of 16 images (illustrated in Fig. 4.3) into IPU and at the first iteration we reset the likelihood of each node to zero, i.e.,  $v_{km}^{(l)} = 0$  in (4.1). We then run the sum-product algorithm for 30 iterations. In general, after a number of iterations, the output values of  $c_k^{(l)}$  in (4.3) become stable. Then we count how many times of  $c_k^{(l)}$  are "1" or "0" in the last ten iterations. A threshold  $Q_1$  is set up here. If  $\sum_{l=21}^{30} c_k^{(l)} \geq Q_1$ , we then know IPU has  $Q_1$  times of decision "1" and  $1 - Q_1$  times of decision "0". If  $Q_1$  is selected as a number close to 10, then we know

whether the decision at this position is stable or not. For example, let  $Q_1$  be 9, then the decision position converges to "1" (or "0") 9 times if  $\sum_{l=21}^{30} c_k^{(l)} \geq Q_1$  (or  $\sum_{l=21}^{30} c_k^l \leq 10 - Q_1$ ). So we can group these stable positions in the set  $\{d_n\}$  where  $\sum_{l=21}^{30} c_{k=n}^{(l)} \geq Q_1$  or  $\sum_{l=21}^{30} c_{k=n}^{(l)} \leq 10 - Q_1$ . And the corresponding decision value "1"s and "0"s can be labeled as  $\{V_{d_n}\}$ , where  $d_n$  lists the stable positions.

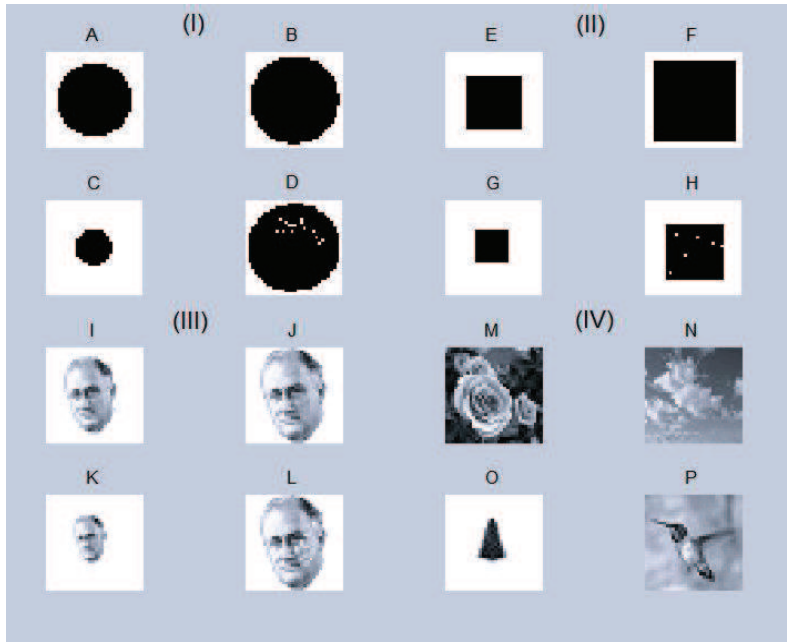


Figure 4.3: Four Sets of 16 Testing Figures

In either way,  $\{d_n\}$  and  $\{V_{d_n}\}$  are stable for this trial. Then a large scale experiments are executed. 100 trials are repeated for one specific image and a few survived positions which are stable  $Q_2$  times out of 100 trials have been figured out.

In Table 4.1(a), we evaluate number of survived positions for Image A with various values of  $\sigma_{a_1}$  and  $\sigma_{a_2}$  at  $Q_1 = 9$  and  $Q_2 = 90$ .  $Q_1 = 9$  and  $Q_2 = 90$  has been chosen because we focus on 70% accuracy of pattern recognition and for the worst case, we still have 810 correct decisions out of a

total of 1000 decisions (100 trials, each with 10 decisions). The result shows that the number of survived positions could vary from none to almost all 1024 variable nodes. At one end, we need to avoid too few survived positions, since without those positions, we would not be able to build a pattern recognition device. On the other end, we also need to avoid too many survived positions due to complexity to sort it out. However, we should be aware that this is caused by the hard decision with the threshold value of  $Q_1$  and  $Q_2$ . In the real systems, we can describe survived condition as soft probabilities. That is, all positions can contribute to recognition, each based on a weight associated to likelihood of stability. In Section 5, we will work on this extension. Now we continue our journey with hard threshold of  $Q_1$  and  $Q_2$ .

We repeat the aforementioned procedure for Images A, B, C, D at  $Q_1 = 9$ ,  $Q_2 = 90$ ,  $\sigma_{a_1} = 0.6$ ,  $\sigma_{a_2} = 1$ . The numbers of survived positions are 306, 379, 477 and 365 for Images A, B, C and D, respectively (For Image A, in this run we get 306 which is slightly different from 304 in Table 4.1(a) because of simulation is an estimate of true value). Out of these positions, we identify 8 common positions, which are  $\{\text{positions } k \text{ (} c_k \text{ value)}\} = \{36(1), 155(1), 459(0), 502(0), 739(1), 899(1), 966(1), 1017(1)\}$ . These common survived positions could be the base to build variation invariance pattern recognition capability. That is, we could recognize the circle pattern regardless of size (Images A, B, C) and distortion (Images B, D). Table 4.1(b) presents the common survived positions for Image A to D when adjusting the value of  $\sigma_{a_1}$  and  $\sigma_{a_2}$ . Our interest will be in those common survived positions of which the amount are not too few, nor too many common.

Of course, we are not only interested in the common survived positions, but also in those survived positions which are different from each other; not only interested in the total four images in one group, but also in any subset of four images. For example, given Images B and D, at  $\sigma_{a_1} = 0.6$ ,  $\sigma_{a_2} = 1.0$ , there are 191 common survived positions, and the rest positions are survived positions in difference(i.e., for B we have  $379 - 191 = 188$ ; and for D we have  $365 - 191 = 174$ ). With these different positions, IPU could construct detectors to differentiate B and D. Given Images

A, B and D, there are 49 common positions. Compare these common positions of A, B and D, with the survived position of C, 8 of them are identified in common, and the rest of 469 in C are different. With these common and different survived positions, detector could be built to differentiate whether the input image is C or one from the subset  $\{A, B, D\}$ .

There are infinite ways to expand the possible combinations of subset of images. The key task of IPU is to have these ready before the task is specified, so it will be ready for any possible variation of specified tasks. Furthermore, when noise power in IPU is high, it can establish solution for easy task first (for example differentiate B and D may carry out at high noise power environment since it does not need 191 common positions). Later on, when the noise power is reduced, IPU could work on more challenging tasks.

Specifically, several tasks regarding differentiating different circle images are set up as follows.

*Task 1: Differentiation Between Images  $\{B, D\}$  and Null Image*

Before differentiating Images B and D, first of all, we need to clarify if the test image is the image with pattern or null image. Here null image is defined as no image but all noise inputs. According to our result, there are none survived positions for null image, while 24 concentrated common survived positions with 24 particular bit values exist for Images  $\{B, D\}$  when  $\sigma_{a1} = 0.7$ ,  $\sigma_{a2} = 1.0$  (variance is increased since it is necessary to avoid redundant common survived positions). Under this case, our Detector I is designed as follows: At beginning, one image from  $\{B, D, \text{null image}\}$  is randomly picked up then it is passed to IPU to process 30 iterations. As mentioned before, for each candidate Image  $X$ , we are able to obtain  $\{d_{nX}\}$  and  $\{V_{d_{nX}}\}$ . If more than  $Q_3$  out of 24 survived positions fall in  $\{d_{nX}\}$  and also match the values in  $\{V_{d_{nX}}\}$ , Detector I will recognize Image  $X$  belongs to the subset  $\{B, D\}$ , otherwise it will be identified as null image. If the final decision matches with the input, then the system makes a right decision, otherwise error

counts. Here  $Q_3$  is set up as 70%. After applying Detector I for a total of 3000 trials with equally distributed for Images B, D and null image, we obtain an error rate as 0.067%.

Table 4.1: Number of Survived Positions for Different Cases

(a) Number of Survived Positions for Images A with the Variations of  $\sigma_{a_1}$  and  $\sigma_{a_2}$

| $\sigma_{a_2} \backslash \sigma_{a_1}$ | 0.9 | 0.8 | 0.7 | 0.6 | 0.5 | 0.4 | 0.3 | 0.2 | 0.1  |
|----------------------------------------|-----|-----|-----|-----|-----|-----|-----|-----|------|
| 1.0                                    | 0   | 2   | 58  | 304 | 566 | 798 | 904 | 978 | 1018 |
| 0.9                                    | 0   | 1   | 55  | 234 | 460 | 657 | 801 | 897 | 972  |
| 0.8                                    | 0   | 0   | 21  | 152 | 363 | 556 | 702 | 800 | 913  |
| 0.7                                    | 0   | 0   | 9   | 101 | 255 | 437 | 597 | 720 | 839  |
| 0.6                                    | 0   | 0   | 1   | 58  | 178 | 348 | 492 | 619 | 737  |
| 0.5                                    | 0   | 0   | 0   | 42  | 134 | 278 | 424 | 540 | 640  |
| 0.4                                    | 0   | 0   | 0   | 22  | 111 | 232 | 377 | 478 | 579  |
| 0.3                                    | 0   | 0   | 0   | 18  | 107 | 207 | 339 | 447 | 537  |
| 0.2                                    | 0   | 0   | 0   | 0   | 0   | 0   | 0   | 0   | 0    |
| 0.1                                    | 0   | 0   | 0   | 0   | 0   | 0   | 0   | 0   | 0    |

(b) Number of Common Survived Positions for Images A to D with the Variations of  $\sigma_{a_1}$  and  $\sigma_{a_2}$

| $\sigma_{a_2} \backslash \sigma_{a_1}$ | 0.7 | 0.6 | 0.5 | 0.4 | 0.3 | 0.2 | 0.1 |
|----------------------------------------|-----|-----|-----|-----|-----|-----|-----|
| 1                                      | 0   | 8   | 70  | 167 | 255 | 322 | 379 |
| 0.9                                    | 0   | 7   | 38  | 100 | 171 | 244 | 314 |
| 0.8                                    | 0   | 3   | 12  | 56  | 106 | 174 | 254 |
| 0.7                                    | 0   | 0   | 6   | 24  | 71  | 121 | 172 |
| 0.6                                    | 0   | 0   | 3   | 13  | 39  | 78  | 119 |
| 0.5                                    | 0   | 0   | 2   | 6   | 18  | 53  | 86  |
| 0.4                                    | 0   | 0   | 0   | 5   | 12  | 41  | 57  |
| 0.3                                    | 0   | 0   | 0   | 3   | 10  | 25  | 50  |

### Task 2: Differentiation Between Image B and D

At high noise  $\sigma_{a_1} = 0.7$ ,  $\sigma_{a_2} = 1.0$ , there are overall 22 common positions survived for Images B and D. Image B has 68 and Image D has 65 different individual survived positions respectively.



Therefore, Detector II is designed as follows: Like Detector I, once a candidate image  $X$  is chosen from B or D then it will be passed into IPU to process 30 iterations. After generating  $\{d_{nX}\}$  and  $\{V_{d_{nX}}\}$ , we are able to calculate ratio  $R_B$  and  $R_D$  as the percentage of 68 positions from Image B and the percentage of 65 positions from Image D fall in  $\{d_{nX}\}$  and also match the values in  $\{V_{d_{nX}}\}$ . If  $R_B > R_D$ , Image  $X$  will be detected as Image B, otherwise Image D. The result displays the error differentiation rate is 0.15% based on 1000 trials of B and 1000 trials of D respectively.

*Task 3: Differentiation Between Images  $\{A, B, C, D\}$  and Null Image*

In the test, the number of survived positions for  $\{A, B, C, D\}$  is 14 and there are none survived positions for images with all noise. Here Detector I is employed, and  $Q_3 = 70\%$  is initialized. Finally we obtain the error differentiation rate 0.28%.

*Task 4: Differentiation Between Images  $\{A, B, D\}$  and Image C*

We employ Detector II, repeat the procedures described in Task 2 and finally get the error differentiation rate 0.0%.

*Simple Examples for Differentiating Patterns in Different Shapes and Sizes*

Now let us increase the number of images from one group of 4 images, to four groups of a total 16 images. Besides the circles, the rest three groups of shapes are squares, faces, and randomly selected patterns. The corresponding detectors are able to be constructed based on survived positions to indicate which group the input image belongs to, or even further, which image it is. Of course, when we sort out these bit positions, we are caring more about how our IPU system handle the task which is not what we have considered during the design phase, for example, could our IPU quickly

differentiate small size images (Image C, G, K, O) from the rest. In other words, how quickly IPU could construct a detector to recognize a group with hidden features such as the input size here.

*Task 5: Differentiation Among Group I, II, III and IV*

Our experimental results indicate at  $\sigma_{a1} = 0.5$  and  $\sigma_{a2} = 1.0$ , there are 72, 76, 141, 25 individual survived positions (already deduct the positions all groups share) for Group I, II, III, IV respectively. Based on these common positions, Detector III is designed as follows to decide which group the input image belongs to. 16 images from 4 distinct patterns in Fig. 4.3 are selected. Each image is tested for 1000 trials. For each image, we try to distinguish it from other 15 images, and finally make a decision as which shape the test image belongs to. When one candidate image  $X$  chosen from Group I - IV comes into IPU for processing 30 iterations' decoding,  $\{d_{nX}\}$  and  $\{V_{d_{nX}}\}$  are able to be achieved. Then we separately calculate the ratio  $R_I, R_{II}, R_{III}$  and  $R_{IV}$  as how much percentage of 72 positions from Group I, 76 positions from Group II, 141 positions group III, and 25 positions from Group IV falling in  $\{d_{nX}\}$  as well as matching the values of  $\{V_{d_{nX}}\}$ . Then Image  $X$  will be detected as the image with the highest maximum ratio from  $\{R_I, R_{II}, R_{III}, R_{IV}\}$ . After applying Detector III the experimental results show when  $\sigma_{a1} = 0.5$  and  $\sigma_{a2} = 1.0$ , the average error differentiation rate for 16 images in Group I- IV is equal to 2.2% whereas detector errors for each images are listed in Table 4.2. In Table 4.2, the small images  $\{C, G, K, O\}$  achieve higher error rates as 12.8%, 6.9%, 2.7%, 2.9% comparing to other regular sized images. One possible but reasonable reason is since the  $H$  matrix of LDPC codes is randomly generated, a few survived positions mapped in small images may carry non-effective information, which mislead small size images making wrong decisions.

Table 4.2: Performance of Detector III in Differentiating Group I, II, III and IV

| Group | Image | Error Rate | Detected As Group I | Detected As Group II | Detected As Group III | Detected As Group IV |
|-------|-------|------------|---------------------|----------------------|-----------------------|----------------------|
| I     | A     | 1.8%       | 98.2%               | 1.8%                 | 0                     | 0                    |
|       | B     | 1.1%       | 98.9%               | 1.1%                 | 0                     | 0                    |
|       | C     | 12.8%      | 87.2%               | 10.1%                | 0.3%                  | 2.4%                 |
|       | D     | 0          | 100%                | 0                    | 0                     | 0                    |
| II    | E     | 0.4%       | 0.4%                | 99.6%                | 0                     | 0                    |
|       | F     | 0          | 0                   | 100%                 | 0                     | 0                    |
|       | G     | 6.9%       | 5.6%                | 93.1%                | 0.1%                  | 1.2%                 |
|       | H     | 0          | 0                   | 100%                 | 0                     | 0                    |
| III   | I     | 6.6%       | 0.1%                | 6.5%                 | 93.4%                 | 0                    |
|       | J     | 0          | 0                   | 0                    | 100%                  | 0                    |
|       | K     | 2.7%       | 2.6%                | 0.1%                 | 97.3%                 | 0                    |
|       | L     | 0          | 0                   | 0                    | 100%                  | 0                    |
| IV    | M     | 0          | 0                   | 0                    | 0                     | 100%                 |
|       | N     | 0          | 0                   | 0                    | 0                     | 100%                 |
|       | O     | 2.9%       | 0.1%                | 2.1%                 | 0.7%                  | 97.1%                |
|       | P     | 0          | 0                   | 0                    | 0                     | 100%                 |

*Task 6: Differentiation Between  $\{C, G, K, O\}$  and  $\{A, B, D, E, F, H, I, J, L, M, N, P\}$*

Now, let us consider the new task: differentiate small images with the rest. There are 42 common survived positions for Images C, G, K, O at  $\sigma_{a1} = 0.7$   $\sigma_{a2} = 1.0$ . In contrast, the common survived positions for the rest 12 images are none. So Detector I described in Section 4 is employed here and each image will be tested for 1000 trials. Finally, we are able to recognize the input image as a small image or not with an average error rate 2.03%, that is, Image A: 4.8% mistakenly detected as small image, Image D: 3.5% small image, Image E: 6.4% small image, Image K: 7.9% non-small image, Image M: 2.8% small image, Image P: 7.1% small image, and for the remaining images they can be 100% correctly detected.

## VIV and Power Scaling Method

Up to now, we have considered simple shapes with still images and the only variation is the size of shape. Now, we will discuss realistic VIV, typically experienced by bio-visual systems. We select parameters similar to what have been used in [102], namely, point spread and fixational motion. We also consider orientation movement introduced in [24].

According to [24], suppose  $N_t$  frames of images are input into IPU. In the  $l$ -th frame,  $s_k^{(l)}$  as the value of  $k$ -th pixel in the visual frame used for the  $l$ -th iteration are generated and then mapped into certain voltage. After that, pixel values of images are convolved with the point spread function in two dimensions. Then images will be further processed with drift-like fixational motion and vertical orientation. After these three operations, Gaussian noise  $n_k^l$  with zero mean and variance  $\sigma_{a1}^2$  and add this noise term to it to get  $b_k^{(l)}$ , which is denoted as the value used in the  $k$ -th variable node for the  $l$ -th iteration in the decoder. Without VIV, we have  $r_k^{(l)} = N_1$  if  $s_k^l = 1$ , and  $r_k^{(l)} = -N_1$  if  $s_k^l = 0$ , which is the mapping process.

Moreover, due to functions of VIV, the pixel values can change dramatically from frame to frame. Therefore, it is necessary to make the images scaled to a compatible range. Four scaling methods are introduced later in this section.

## Visual Information Variability

### Point Spread (PS)

According to [102], line-spread functions could be derived as

$$h(x, y) = \frac{a_1}{2\pi a_3} \exp\left[-\frac{0.5(x^2 + y^2)}{a_3^2}\right] + \frac{a_2}{2\pi a_4} \exp\left[-\frac{0.5(x^2 + y^2)}{a_4^2}\right] \quad (4.4)$$

where the coefficients  $a_1 = 0.417$ ,  $a_2 = 0.583$ , variances  $a_3 = 0.443\beta_1$  and  $a_4 = 2.04\beta_1$ . We can tune the variance by varying the scale factor,  $\beta_1$ .

### Fixation Motion (FM)

The second aspect is a drift-like motion, which we model as a Gaussian random-walk function with a Gabor-like impulse response [102]

$$g(t) = \left( \frac{40}{\sqrt{2\pi a_5}} \exp\left[-\frac{t^2}{2a_5^2}\right] - 0.0117 \right) \cos(4\pi t/1000) \quad (4.5)$$

where  $a_5 = 223.61\beta_2$  ms, and  $\beta_2$  is the scale factor.

### *Orientation Movement (OM)*

The third aspect is orientation movement, which results in rotation along the clockwise and counterclockwise directions. A simple filter is used as,

$$\theta(t) = 0.95\theta(t-1) + 0.05n_0(t)\frac{\pi}{9}\beta_3 \quad (4.6)$$

where  $n_0(t) \sim \mathcal{N}(0, \sigma_1^2)$  and the initial value of  $\theta(t)$ ,  $\theta(0)$ , is a uniformly distributed random variable in the interval of  $[-20^\circ, 20^\circ]$ .

So far, three VIV aspects have been fully introduced. In our design, there are three scale factors  $\beta_1, \beta_2, \beta_3$  used to control the degree of VIV aspects. For example, if we set all of them zero, then there are no motion, point spread, and orientation movement; if we set all of them as 2, then the VIV aspects become much severer than the case which we set all of them as 1.

### *Power Scaling Method*

In this subsection, we list four methods to present how we do power scaling for pixels values in each imperfect frame. We define  $f_{(i,j)}$  as  $s_{(i,j)}$  after adding VIV. Method 1 has no-mean or no-energy normalization, i.e., we directly feed  $r_{(i,j)}^{(l)} = f_{(i,j)}^{(l)}$  as its value to decoder.

Method 2 has mean normalization, i.e.,

$$m^{(l)} = \frac{\sum_{i=1}^I \sum_{j=1}^J f_{(i,j)}^{(l)}}{I * J} \quad (4.7)$$

$$r_{(i,j)}^{(l)} = f_{(i,j)}^{(l)} - m^{(l)} \quad (4.8)$$

Then pixels have been grouped into two groups, i.e., those above the mean, i.e.,  $r_{(i,j)}^{(l)} > 0$  and those below the mean. Later, mean value of each group is computed as

$$\begin{aligned}
m_p^{(l)} &= \frac{\sum_{i=1}^I \sum_{j=1}^J r_{(i,j)}^{(l)}}{\sum_{i=1}^I \sum_{j=1}^J Z\left(r_{(i,j)}^{(l)} > 0\right)} \text{ if } r_{(i,j)}^{(l)} > 0 \\
m_n^{(l)} &= \frac{\sum_{i=1}^I \sum_{j=1}^J r_{(i,j)}^{(l)}}{\sum_{i=1}^I \sum_{j=1}^J Z\left(r_{(i,j)}^{(l)} \leq 0\right)} \text{ if } r_{(i,j)}^{(l)} \leq 0
\end{aligned} \tag{4.9}$$

where

$$Z(x) = \begin{cases} 1 & x \text{ is true} \\ 0 & x \text{ is false} \end{cases} \tag{4.10}$$

Finally,  $r_{(i,j)}^{(l)}$  is updated as  $\frac{r_{(i,j)}^{(l)}}{\min(m_p^{(l)}, |m_n^{(l)}|)}$ ,  $b_{(i,j)}^{(l)} = r_{(i,j)}^{(l)} + n_{(i,j)}^{(l)}$  where  $n_{(i,j)}^{(l)}$  is a Gaussian distributed noise with zero-mean, variance  $\sigma_{a1}^2$ . We do so to ensure the group with smaller mean, i.e.,  $\min(m_p^{(l)}, |m_n^{(l)}|)$  is not too small to drive the decoder to all zero or all one cases.

In Method 3, following the mean-normalization described in Method 2, the total energy of both positive and negative groups have been computed as follows.

$$\begin{aligned}
W_p^{(l)} &= \sum_{i=1}^I \sum_{j=1}^J \left[ r_{(i,j)}^{(l)} \right]^2 \text{ if } r_{(i,j)}^{(l)} > 0 \\
W_n^{(l)} &= \sum_{i=1}^I \sum_{j=1}^J \left[ r_{(i,j)}^{(l)} \right]^2 \text{ if } r_{(i,j)}^{(l)} \leq 0
\end{aligned} \tag{4.11}$$

Define  $S_p^{(l)}$  and  $S_n^{(l)}$  as scaling factors to the positive and negative groups.

$$\begin{aligned} S_p^{(l)} &= \min \left( 1, \sqrt{\frac{W_n^{(l)}}{W_p^{(l)}}} \right) \\ S_n^{(l)} &= \min \left( 1, \sqrt{\frac{W_p^{(l)}}{W_n^{(l)}}} \right) \end{aligned} \quad (4.12)$$

And also, each  $r_{(i,j)}^{(l)}$  is multiplied by one of two scaling factors,  $S_p^{(l)}$  or  $S_n^{(l)}$ , if its value is positive or negative. By doing so, we ensure the energy in both groups are balanced and normalized, therefore for given noise power densities the decoder would not fail to converge.

In Method 4, due to the total energy of each corrupted digit images (after method 2) varying drastically,  $S_p^{(l)}$  and  $S_n^{(l)}$  could be set as

$$\begin{aligned} S_p^{(l)} &= \sqrt{\frac{1500}{W_p^{(l)}}} \\ S_n^{(l)} &= \sqrt{\frac{1500}{W_n^{(l)}}} \end{aligned} \quad (4.13)$$

so the  $W_p^{(l)}$  and  $W_n^{(l)}$  for each image under any conditions will be the same after scaling.

In summary, Method 1 is the method without scaling which is for comparison only, and Method 2 is the prerequisite for both Method 3 and Method 4.

### *Results of Differentiating Different Patterns with VIV*

So far, all the VIV aspects and power scaling methods have been discussed in details. Then we begin to process the low-resolution pattern images shown in Fig. 4.3 by considering all VIV aspects.



We list all the testing tasks in Table 4.3. Each image is tested for 1000 trials with VIV scale factors  $\beta_1 = 1.0, \beta_2 = 1.0, \beta_3 = 1.0$  using power scaling Method 4.

In order to obtain fit number of common survived positions (neither too many nor too few),  $\sigma_{a_1}$  and  $\sigma_{a_2}$  are appropriately selected for each task. Based on the common positions generated from each combination, the corresponding results for all six tasks are listed in Table 4.3. For comparison, results for images without any VIV factor are also presented there.

When comparing Column 4 and Column 5 in Table 4.3, it is clear to see the recognition error rates for images with VIV dramatically increase, especially for Task 2, 5 and 6. The reason is obvious. Each VIV factor disturbs IPU from obtaining effective common survived bits in decoding, which causes the detector difficult to make correct decisions. Hence, based on these results, we will continue studying other detectors to improve the performance, and more importantly, to pursue effective detectors that could handle more complex recognition tasks.

Table 4.3: Performance Comparison in Detectors for Images With and Without VIV

| Task No. | Differentiation Task                                  | Detector | Error rate for images with VIV | Error rate for images without VIV |
|----------|-------------------------------------------------------|----------|--------------------------------|-----------------------------------|
| 1        | {B, D} and null image                                 | I        | 0.20%                          | 0.067%                            |
| 2        | B and D                                               | II       | 28.30%                         | 0.15%                             |
| 3        | {A, B, C, D} and null image                           | I        | 0.80%                          | 0.28%                             |
| 4        | {A, B, C} and D                                       | II       | 0,06 %                         | 0.00%                             |
| 5        | Group I, II, III and IV                               | III      | 32.56%                         | 2.2%                              |
| 6        | {C, G, K, O} and {A, B, D, E, F, H, I, J, L, M, N, P} | I        | 11.25%                         | 2.03%                             |

#### Experimental Procedures and IPU Detections for Simple Images

So far, we have figured out the important role the bit convergence played during decoding for both still images and images with VIV. Apparently, when meeting images with VIV, the current detec-

tors which make full use of survived converge bits do not perform well. Therefore in this section, we will design other detectors which will dramatically decrease the error rates, and furthermore could be widely applied in all kinds of low-resolution images, not only simple shape images, but also face images.

### *Detector Design*

In Section 4, we have mentioned every position in an image could contribute to recognition, based on a weight associated to likelihood of stable. Our new detectors are hence built based on this idea.

*Training:* In IPU structure depicted in Fig. 1, after completing sum-product algorithm in decoding, IPU could generate two kinds of outputs: (1) averaged LLR (i.e., soft decision, given in (4.14) and multiplicity (i.e., hard decision, given in (4.15)) for each image in Fig. 4.3.

$$v_k(z) = \frac{\sum_{l=21}^{I_a} v_k^{(l)}}{I_a - 20} \quad (4.14)$$

In averaged LLR output (soft decision), averaged likelihood for each bit has been calculated. In (4.14),  $z$  stands for any training images (e.g., Image B and D for Task 2, all 16 images for Task 5 and 6) in Fig. 4.3, and  $v_k(z)$  denotes the averaged likelihood of  $k$  variables in  $z$ .  $v_k(z)$  is the mean of  $v_k^l$  over last  $I_a - 20$  iterations. In order to avoid error propagations, we only count the last 300 iterations, i.e.  $I_a = 320$ .

$$\begin{aligned} p(c_k = 0|z) &= \frac{\sum_{l=21}^{I_a} Z(c_k^{(l)} = 0)}{I_a - 20} \\ p(c_k = 1|z) &= \frac{\sum_{l=21}^{I_a} Z(c_k^{(l)} = 1)}{I_a - 20} \end{aligned} \quad (4.15)$$

On the other hand, in multiplicity output (hard decision), at each iteration the status of one bit will be decided. In (4.15), we count the occurrences of 0 and 1 of hard decision making (denoted as  $c_k^l(z)$  in (4.15)) of each bit  $k$  during last  $I_a - 20$  iterations. After dividing by total  $I_a - 20$  iterations, the probabilities of 0 and 1 for each  $k$  variable node of image will be obtained.

Therefore, two new types of detectors: Detector IV and Detector V are defined in Table 4.4. In Table 4.4, Detectors IV utilizes soft decision and Detector V for hard decision. In both detectors, full scale detector (32 \*32 nodes) and reduced scale detectors (part of whole image nodes) are both considered.

Table 4.4: Detector IV and Detector V

| Detector | Detector Input | $\sigma_{a2}$    | number of nodes      |
|----------|----------------|------------------|----------------------|
| IV       | $v_k(z)$       | $1.5\sigma_{a1}$ | 5, 10, 20, 40, 32*32 |
| V        | $p(c_k = 0 z)$ | $1.5\sigma_{a1}$ | 5, 10, 20, 40, 32*32 |

*Test:* In the test procedure, once a candidate image  $X$  of Fig. 4.3 has been selected, it will be decoded  $I_l$  iterations. For each iteration, distance between LLR of  $v_k^{(l)}$  and  $\ln \frac{p(c_k=1|z)}{p(c_k=0|z)}$ , or  $v_k(z)$  will be both obtained. Define  $L_k(z) = v_k(z)$  in Detector IV, whereas  $L_k(z) = \ln \frac{p(c_k=1|z)}{p(c_k=0|z)}$  in Detector V. Then after scaling, the distance between  $v^{(l)}$  of Image  $X$  and  $L(z)$  of image  $z$  during  $l$ -th iteration could be calculated in (4.16)

$$\xi^{(l)}(z) = \sum_{k=1}^{I*J} (v_k^l - L_k(z))^2 \quad (4.16)$$

And test decision is the one with minimum sum of distances over  $I_l$  (30 in our program) iterations.

$$Decision = arg \min_{z \in \mathbf{Z}} \sum_{l=1}^{I_l} \xi^l(z) \quad (4.17)$$

If decision matches the input then the system makes a right decision, otherwise error counts. Here,  $\mathbf{Z}$  stands for the training images pool in each task.

### *Results of Detectors IV and V in Tasks 2, 5, 6*

In this subsection, we display the performance of Detectors IV and V regarding Tasks 2, 5, and 6 in Table 4.5. Each image in each task is executed for 1000 trials, at  $\sigma_{a1} = 1.0$ ,  $\sigma_{a2} = 1.5$  with power scaling Method 4. Our goal is to verify if our new Detectors IV and V are able to improve the detectors performance for images with VIV. And also, for comparison, performance of Detectors IV and V for still images are also listed.

Table 4.5 presents the evidence that Detectors IV and V are able to differentiate imperfect images in low error rates. For instance, for images with VIV in Task 5, the resulted error rates are 0.32% for full scale (all 32\*32 bits are used) Detector IV (soft decision), whereas 0.18% for full scale Detector V (hard decision), improving much than using Detector III shown in Table 4.3. For Task 2 and Task 6, Detectors IV and V could drop down the error rate to 0.0%, comparing with 28.30% and 11.25% listed in Table 4.3 for Detector II in Task 2 and Detector I in Task 6. On the other side, for the images without any VIV, error rate for Tasks 2, 5, 6 for Detectors IV and V are all close to 0%.

### *Features Appeared in Task 5*

So far, we have thoroughly presented how our IPU system trains and tests, especially for images with VIV. But it is still not clear how noise variance, power scaling methods specifically contribute to recognition and how greatly they impact the recognition results. Therefore, this subsection will include how the value of noise variance, how the option of power scaling methods, and even more,

how  $\mathbf{H}$  matrix involved in LDPC codes influence the recognition results. Task 5 for images with VIV will be taken as an example.

Table 4.5: Performance Comparison in Detectors IV, V for Images With and Without VIV

| Task No. | Differentiation Task                                  | Detector | Error rate for images with VIV | Error rate for images without VIV |
|----------|-------------------------------------------------------|----------|--------------------------------|-----------------------------------|
| 2        | B and D                                               | IV       | 0%                             | 0%                                |
| 2        | B and D                                               | V        | 0%                             | 0%                                |
| 5        | Group I, II, III and IV                               | IV       | 0.32%                          | 0.0063%                           |
| 5        | Group I, II, III and IV                               | V        | 0.18%                          | 0.0125%                           |
| 6        | {C, G, K, O} and {A, B, D, E, F, H, I, J, L, M, N, P} | IV       | 0%                             | 0%                                |
| 6        | {C, G, K, O} and {A, B, D, E, F, H, I, J, L, M, N, P} | V        | 0.0063%                        | 0%                                |

Table 4.7 indicates four types of error rates for Task 5 (detecting which group the test image comes from). In order to observe the constraints of LDPC codes in decoding, one comparison group for IPU system without LDPC  $\mathbf{H}$  matrix is set up. And also, power scaling methods 3 and 4 are fully compared as well. As a result, it can be observed that (1) the performance of our IPU detection is improved by LDPC code constraint based on the comparison between Row 1 and Row 2, Row 3 and Row 4 respectively in Table 4.7; (2) Power scaling Method 4 which normalizes energy for all frames across every frame remarkably reduces the error rate, firmly indicating Method 4 contributes more to the optimal detector than Method 3 does.

Furthermore, we focus on researching the value of  $\sigma_{a1}$  and  $\sigma_{a2}$  in IPU system. In Fig. 4.5, the curves of error rates with different value of  $\sigma_{a1}$  and  $\sigma_{a2}$  have been plotted. Here, 10 times a total of 10000 trials for all tested images with independent noise seeds are executed. Meanwhile, power scaling Method 3 and Method 4, system with or without LDPC  $\mathbf{H}$  matrix involved are also considered for comparison. In Fig. 4.5, it is clear to see SNR controls the system performance and around  $\sigma_{a1} = 1.3$ ,  $\sigma_{a2} = 1.5\sigma_{a1}$ , the system with  $\mathbf{H}$  using Method 4 reaches the lowest error rate.

In Task 5, when differentiating Group I, II, III and IV, 16 images from 4 distinct shapes in Fig. 4.3 have been selected for training and test. For each image, we try to distinguish it from other 15 images, and finally make a decision as which shape the test image belongs to. According to our design, all the training and test images are fully considered the influence of all VIV aspects - point spread, drift-like motion, orientation movement and Gaussian noise demonstrated above. The error rate of system in 10000 trials is listed in Table 4.6 and 4.7. Table 4.6 shows the individual differential error rate as if the test image can be recognized correctly from 16 images. On the other hand, Table 4.7 illustrates the group differential error rate as if the test image can be detected as the same shape group in IPU. More specifically, different power scaling method -method 3 and method 4 have been chosen for comparison. And also, error rate of system without  $H$  is presented. In this instance,  $\sigma_1$  and  $\sigma_2$  is identical to 1.3, 1.95 respectively.

From Table 4.6 and 4.7 it is apparent to see 1) the power scaling method 4 leads significant improvement in recognition performance comparing with method 3, which verifies our assumption in advantage of method 4 in previous section. 2)  $H$  matrix plays a significant role in decoding in IPU. When comparing row 1 with row 2, row 3 with row 4 in Table 4.6 as well as those in Table 4.7,  $H$  matrix decreases the system error rate therefore improve the system performance more than 50%. 3) group differential rate is much lower than individual one this is because a few images in each shape shows similar to each other. For instance, circle Image B versus Image D, square image E versus image H, which dramatically gains the difficulty in differentiation especially from those located in the same shape.

Furthermore, we focus on researching the function of  $\sigma_{a1}$  and  $\sigma_{a2}$  in system. In Fig. 4.4 and Fig. 4.5, the curves of individual and group differential error rates with different value of  $\sigma_{a1}$  and  $\sigma_{a2}$  have been plotted. Here, 10 times 10000 trials with independent noise seeds are implemented. Meanwhile, power scaling method 3 and method 4 are also considered for comparison. From the plots it is clear to see at the beginning, the error rate curve shows relevantly fluctuating and then

with noise becoming heavy, system performance increase steadily after  $\sigma_{a1} = 1.0$ ,  $\sigma_{a2} = 1.5$ .

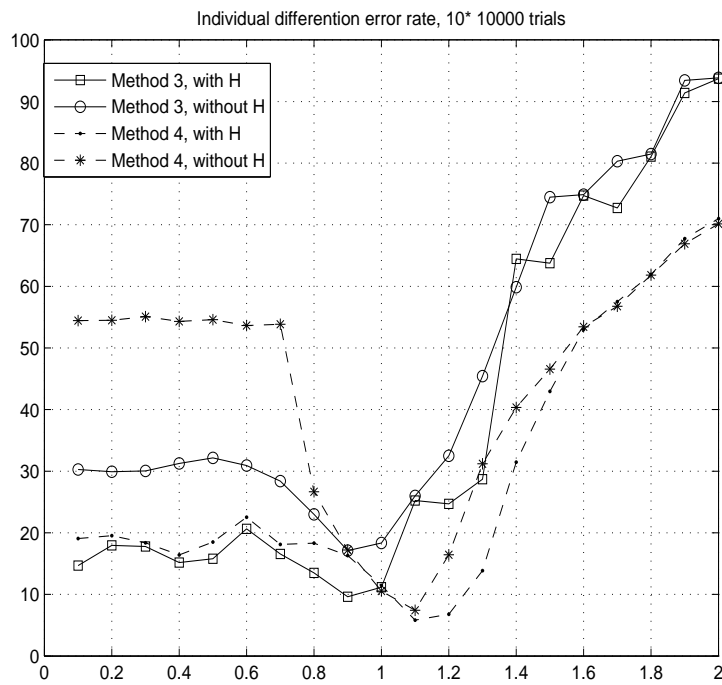


Figure 4.4: Differentiation Error Rates in Independent Number of Noise Seeds for 10e5 Trials

Table 4.6: Differentiation Error Rate for Recognizing One Out of 16 Images with  $\sigma_{a1} = 1.3$ ,  $\sigma_{a2} = 1.95$

| Power scaling method | With or without H matrix | Error rate |
|----------------------|--------------------------|------------|
| Method 3             | with H                   | 28.71%     |
| Method 3             | without H                | 45.44%     |
| Method 4             | with H                   | 13.85%     |
| Method 4             | without H                | 31.30%     |

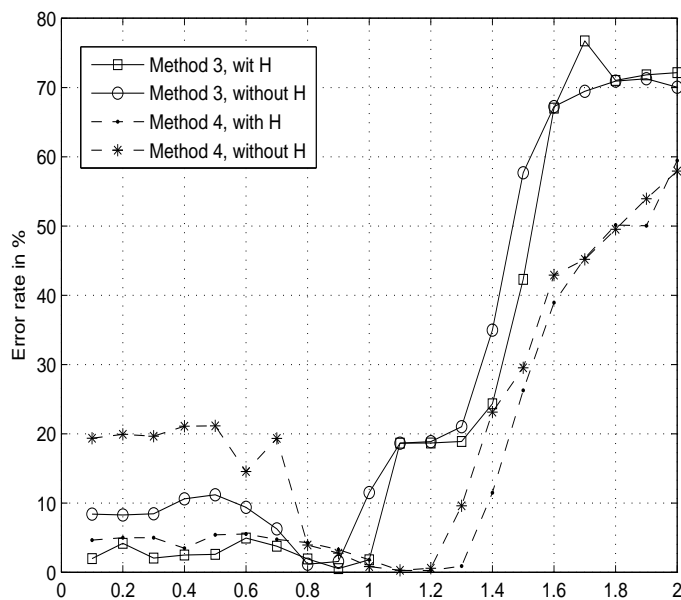


Figure 4.5: Error Rates of Different Noise Variance for Task 5

Table 4.7: Four Types of Error Rates for Task 5

| Power scaling method | With or without H Matrix | Error rate |
|----------------------|--------------------------|------------|
| Method 3             | with H                   | 11.86%     |
| Method 3             | without H                | 25.62%     |
| Method 4             | with H                   | 0.91%      |
| Method 4             | without H                | 9.59%      |

### Recognition Low-Resolution Face Images with VIV

In this subsection, we will focus on applying IPU to the low-resolution face recognition. As introduced in our previous section, IPU could identify and classify various simple images. So whether IPU could be employed in the area of bio-visual low-resolution face recognition also rises our curiosity.



We follow the training and test procedures described in Section 5 and use Detector IV and V to process low-resolution face images with VIV.

The database of face we used in our experiment is from AT&T Laboratories Cambridge [103]. In this data set, 10 different images of each of 40 distinct objects are presented. Each image was taken at different times, varying the lighting, facial expressions (open/closed eyes, smiling/not smiling) and facial details (glasses/no glasses). All the images were taken against a dark homogeneous background with the subjects in an upright, frontal position (with tolerance for some side movement). The size of each image is 92\*112 pixels, with 256 grey levels.

In our system, all the face images are transformed to 32 pixels \* 32 pixels but with the same grey levels as the original images, so each pixel is specified by an integer from 0 to 255. Fig. 4.6 shows ten different lighting and expressions image of one person in grey color.

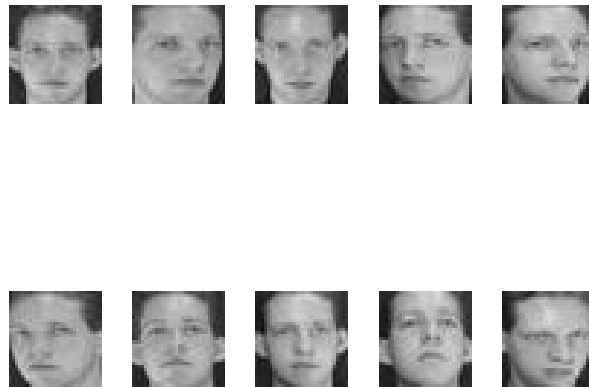


Figure 4.6: Ten Different Face Images of One Distinct Object

As inputs to each photo-sensor units in Fig. 4.1, each pixel is scaled to double precision -real number from 0 to 1, i.e.  $s_k \in (0,1)$ ,  $k=1, 2, \dots, 32*32$  instead of integer 0 for black and 1 for white image.

To solve face recognition problem, we consider all VIV aspects as well as power scaling Methods 3 and 4 illustrated in Section 5. We also utilize Detectors IV and V defined in Section 5. Fig. 4.7 shows the effects of all VIV on one face image.

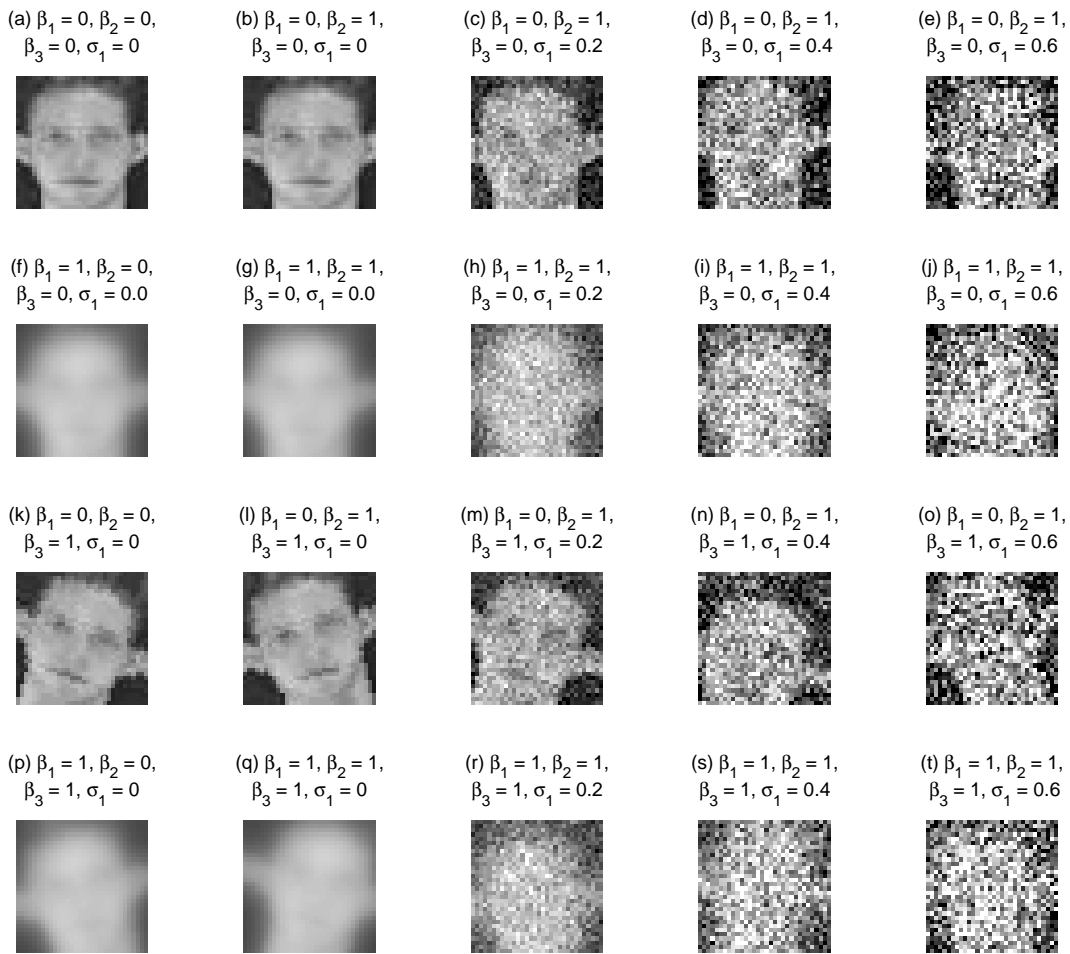


Figure 4.7: Effects of Point Spread ( $\beta_1$ ), Drifting-like Motion( $\beta_2$ ), Head Orientation Rotation ( $\beta_3$ ) and Gaussian Noise ( $\sigma_{a1}$ ) on One Face Image

## Numerical Results

Various experiments and results are presented in this section. Each experiment was ran 10000 times. And the result we show here may not be the best possible performance as it just indicates the influence of some parameters.

### *Face Recognition: One Out of Ten Faces*

In this subsection, 10 face images from 10 distinct objects have been picked up for training and test, that is, for 10 classes, each class only has one image. For each image, we try to distinguish it from other 9 images in different circumstances. All the images are considered the influence by all VIV aspects -point spread, drift-like motion, head orientation movement and Gaussian noise.

The error rates for 10 cases of Detector IV and Detector V are presented in the Table 4.8(a) with Gaussian noise  $\{\sigma_{a1}, \sigma_{a2}\} = \{0.8, 1.0\}$  and Table 4.8(b) with  $\{\sigma_{a1}, \sigma_{a2}\} = \{1.5, 2.25\}$ . In order to clearly identify how the different number of nodes influences the recognition performance in Detector IV and V, we reduced the number of nodes from 1024 to certain number of nodes, i.e., 40, 20, 10, 5 in reduced scale Detector IV and V. More specifically, we make plots in Fig. 4.8(a) and Fig. 4.8(b) as bit error rate versus number of nodes. It is clear to see in both figures, with the number of nodes for detector increasing, the recognition bit error rates dramatically decrease. What's more, by comparing different detector inputs under the same number of nodes at 5, 10, 20 and 40, we can see the performance of Detector IV(soft decision) shows much better than Detector V(hard decision) especially for low percentage of nodes used. When all the  $32*32$  nodes are employed, according to Case 5 and 10 in Table 4.8(a) and Table 4.8(b) the error rates of Detector IV and V are both under 0.1% and show extremely close.

For further studying IPU performance when employing all the nodes, a histogram of 10000 times

testing in independent noise seeds for Case 10 is shown as an example in Fig 4.9(a). As a contrast, a histogram of 10000 trials without LCPD  $\mathbf{H}$  matrix involved is also presented in Fig. 4.9(b). We can see in Fig. 4.9(a), the errors of more than 65% trials is less than 10 out of 10000. While in Fig. 4.9(b), errors for system without  $\mathbf{H}$  matrix are spread between the range of 0-1600. By comparing the distribution of bit error rates in Fig. 4.9(a) and Fig. 4.9(b), we are able to firmly conclude the important role  $\mathbf{H}$  matrix plays in decoding especially for the image information with kinds of heavy noise.

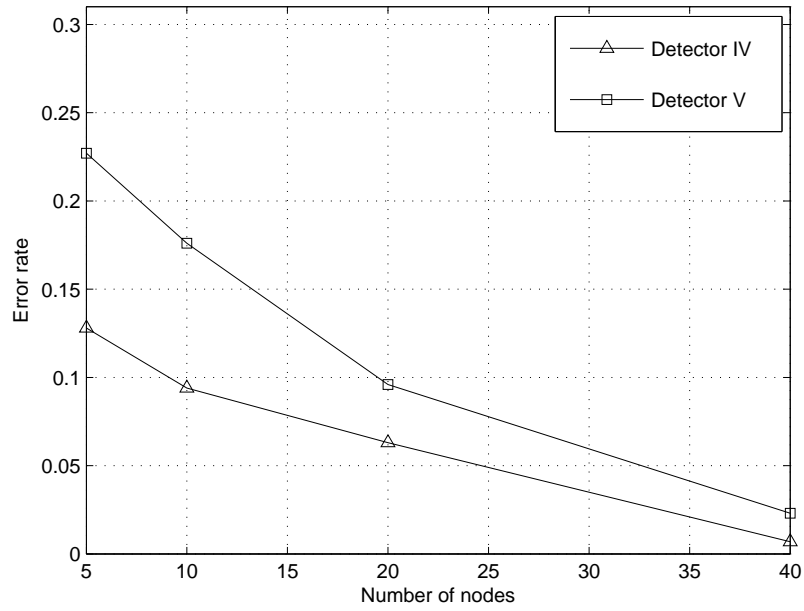
Table 4.8: Error Rates for Recognizing One Out of 10 Faces

(a) Error Rates for Recognizing One Out of 10 Faces with  $\sigma_{a1} = 0.8, \sigma_{a2} = 1.0$

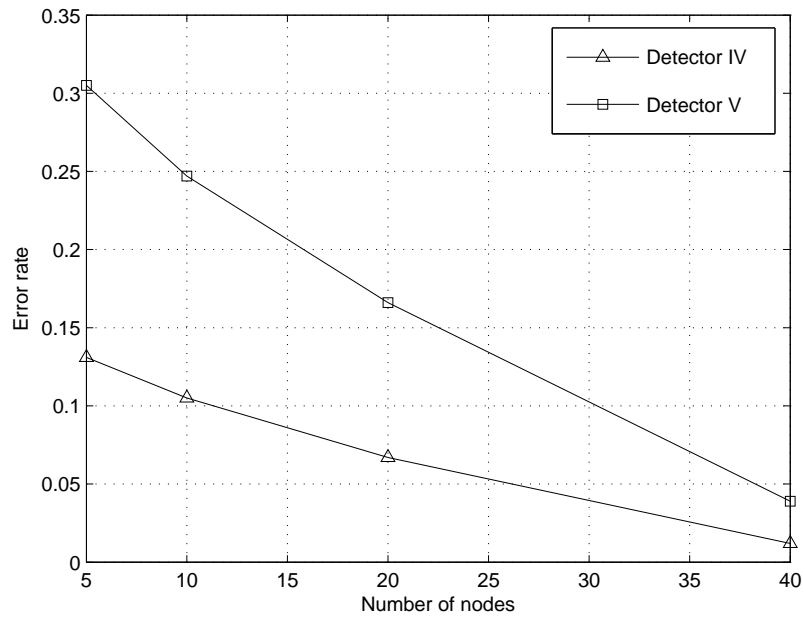
| Case | Detector | Number of nodes | Error rate |
|------|----------|-----------------|------------|
| 1    | IV       | 5               | 12.88%     |
| 2    | IV       | 10              | 9.47%      |
| 3    | IV       | 20              | 6.32%      |
| 4    | IV       | 40              | 0.69%      |
| 5    | IV       | 1024            | 0.07%      |
| 6    | V        | 5               | 22.74%     |
| 7    | V        | 10              | 17.61%     |
| 8    | V        | 20              | 9.63%      |
| 9    | V        | 40              | 2.32%      |
| 10   | V        | 1024            | 0.04%      |

(b) Error Rates for Recognizing One Out of 10 Faces with  $\sigma_{a1} = 1.5, \sigma_{a2} = 2.25$

| Case | Detector | Number of nodes | Error rate |
|------|----------|-----------------|------------|
| 1    | IV       | 5               | 13.14%     |
| 2    | IV       | 10              | 10.53%     |
| 3    | IV       | 20              | 6.76%      |
| 4    | IV       | 40              | 1.25%      |
| 5    | IV       | 1024            | 0.03%      |
| 6    | V        | 5               | 30.58%     |
| 7    | V        | 10              | 24.79%     |
| 8    | V        | 20              | 16.67%     |
| 9    | V        | 40              | 3.95%      |
| 10   | V        | 1024            | 0.08%      |



(a) Error Rates for Two Types of Detectors at  $\sigma_{a1} = 0.8, \sigma_{a2} = 1.0$



(b) Error rates of Two Types of Detectors at  $\sigma_{a1} = 1.5, \sigma_{a2} = 2.25$

Figure 4.8: Error Rates of Two Types of Detectors at Different Noise Variance

### *Impact of LDPC Codes with Large Minimum Girth to the Results of One Out of Ten*

In the last subsection, we presented the result of our face recognition system, but only focused on random generated LDPC codes. Although the current results show the system is well-performed, another option would also be fully considered, which is to improve LDPC codes. Indeed, excellent performance results for LDPC codes are reported for the short block lengths regime in [42] and [104]. In [42], the author designed short regular LDPC codes with large minimum girth ( $G_{min}$ ) and indicated those could beat the average performance of regular ensembles of the LDPC codes over binary symmetric channels. In this subsection, we use the methods illustrated in [42] to construct specific LDPC codes with larger  $G_{min}$ , and compare their performances in IPU with those using general LDPC codes.

We now construct two types of LDPC codes, one is (3,6) LDPC codes with length of 1024 and  $G_{min} = 6$  ( $H_1$  matrix), the other is (2, 4) LDPC codes with length of 1024 and  $G_{min} = 8$  ( $H_2$  matrix). In Fig. 4.10(a) and Fig. 4.10(b), cases of  $H_1$ ,  $H_2$  matrix as well as cases of random  $H$  matrix and IPU without  $H$  matrix in power scaling method 3 and 4 have been separately presented, which show the recognition error rates as a function of  $\sigma_{a1}$  for the test of one out of ten faces. Here, we execute the tests in 10 times 10000 trials within independent noise seeds.

From the results, we can see no matter for method 3 and method 4, it shows LDPC codes with larger  $G_{min}$  outperforms the generals. Moreover, (2, 4) codes with  $G_{min} = 8$  achieves the most excellent performance in both figures in Fig. 7. Meanwhile as we indicated before, the power scaling method 4 normalized the energy for all frames across every frame and its performance more substantially contribute to the optimal detector than Method 3.

### *Face Recognition: One Out of 20 and 40*

In this subsection, we extend one out of 10 faces to one out of 20 and 40 faces. We select the first image of each distinct object in the AT&T ORL database [103], and consider 20 and 40 groups of images.

The performance of one out of 20 and 40 faces are listed in the Fig. 4.12 by comparing between systems with  $H$  matrix and without at  $\sigma_{a1} = 1.5$ ,  $\sigma_{a2} = 2.25$ . Detector V and power scaling method 3 are exploited for comparison. Fig. 4.12 shows the error rate for IPU with LDPC codes are only 3.45% and 4.47% for the case of one out of 20 and 40, comparing to 11.68% and 12.74% respectively for IPU without any functions of LDPC codes.

### *Face Recognition in Multi-class Classification*

In this subsection, we document multiple classes low resolution face recognition under VIV conditions.

In AT&T ORL standard face database [103], each of the 40 distinct objects has 10 different images. For our IPU system, we divide these images into training and test set with different numbers, and we observe the recognition rate of IPU. All of these images will have full VIV aspects.

For ease of representation,  $Train_m/Test_n$  means  $m$  images per person are randomly selected for training and the rest  $n$  for test. As an example, when  $m = 5$ ,  $n = 5$ , 5 training images and 5 test images per object have been randomly selected and tested in 10000 trials in order to verify if the test image is from the correct class. Table 4.10 gives a table for ten kinds of training and test set in 40 classes, where  $scale = 0.5$ ,  $\{\sigma_{a1}, \sigma_{a2}\} = \{0.8, 1.0\}$ . Next, we adjust the number of classes for training and test, then compare their different performances. Table 4.9 summarizes the recognition

error rate when  $Train_m/Test_m = \{2/8, 3/7, 4/6, 5/5, 6/4, 7/3, 8/2\}$  in 50 randomly splits, within Detector V,  $scale = 0.5$ ,  $\{\sigma_{a1}, \sigma_{a2}\} = \{0.8, 1.0\}$ .

Table 4.9: Recognition Accuracy Rate on ORL Database (Mean  $\pm$  Std-dev)

| $Training_m/Test_n$ | 2/8                | 3/7                | 4/6                | 5/5                | 6/4               | 7/3                | 8/2                |
|---------------------|--------------------|--------------------|--------------------|--------------------|-------------------|--------------------|--------------------|
|                     | 79.62% $\pm$ 3.45% | 87.36% $\pm$ 2.93% | 91.21% $\pm$ 2.77% | 93.45% $\pm$ 1.44% | 94.94% $\pm$ 1.7% | 95.67% $\pm$ 1.41% | 96.75% $\pm$ 1.34% |

Table 4.10: Recognition Rate for 10 Cases of Training and Test Sets in 40 Classes with  $scale = 0.5$ ,  $\sigma_{a1} = 0.8$ ,  $\sigma_{a2} = 1.0$

| Case No. | Training set     | Test set         | Recognition error rate |
|----------|------------------|------------------|------------------------|
| 1        | {1, 2, 3, 4, 5}  | {6, 7, 8, 9, 10} | 7%                     |
| 2        | {6, 7, 8, 9, 10} | {1, 2, 3, 4, 5}  | 9.5%                   |
| 3        | {1, 3, 5, 7, 9}  | {2, 4, 6, 8, 10} | 5.5%                   |
| 4        | {2, 4, 6, 8, 10} | {1, 3, 5, 7, 9}  | 6.5%                   |
| 5        | {2, 3, 5, 8, 9}  | {1, 4, 6, 7, 10} | 4.5%                   |
| 6        | {1, 2, 3, 6, 8}  | {4, 5, 7, 9, 10} | 6.5%                   |
| 7        | {2, 3, 6, 7, 10} | {1, 4, 5, 8, 9}  | 6.5%                   |
| 8        | {1, 2, 5, 9, 10} | {3, 4, 6, 7, 8}  | 8%                     |
| 9        | {1, 3, 4, 5, 7}  | {2, 6, 8, 9, 10} | 5%                     |
| 10       | {3, 4, 7, 8, 10} | {1, 2, 5, 6, 9}  | 6.5%                   |

Every trial in 10000 total is independent. From Table 4.9, we can see for five training and five test, the average recognition accuracy rate for 50 sets is 93.45%, standard variance is 1.44%.

The experimental procedures besides VIV and power scaling methods are exactly followed those described in [105] using the same AT&T ORL database [103]. For comparison, for instance, according to Table 3 illustrated in Section 4.2 in [105], for 5 training and 5 testing tasks, the SSSL approaches proposed by Cai *et al.* [105] could obtain up to  $97.4 \pm 1.2\%$  recognition accuracy. Cai *et al.* [105] have also reported the performance of other popular methods such as Eigenface ( $87.9 \pm 2.5\%$ ), TensorPCA ( $88.1 \pm 2.5\%$ ), Fisherface ( $94.3 \pm 1.4\%$ ), 2DLDA ( $95.8 \pm 1.2\%$ ), Laplacianface ( $93.0 \pm 1.9\%$ ), and NPE ( $93.4 \pm 1.8\%$ ). All these do not have VIV. For 5 training



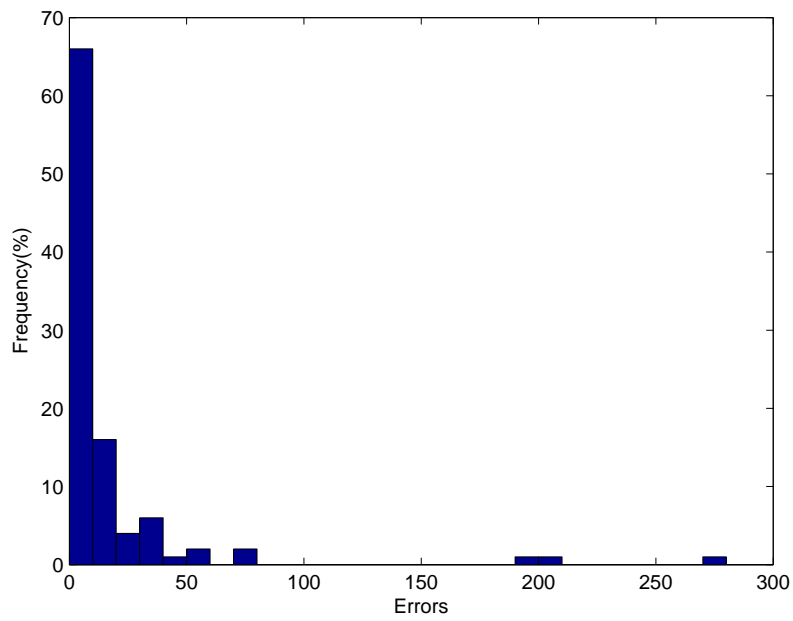
and 5 testing tasks shown in Table 4.9, when adding up VIV we are able to obtain  $93.45 \pm 1.44\%$  recognition accuracy.

On the other hand, we compare the recognition performance in different methods based on the same training and testing samples using VIV, where  $scale = 0.5$ ,  $\{\sigma_{a1}, \sigma_{a2}\} = \{0.8, 1.0\}$ . The number of training and test sample splits are 50. The methods provided for comparison with our IPU are PCA and LDA with PCARatio = 1.0. The recognition is carried out by using nearest neighbor classifier(NN). Fig. 4.13 presents the recognition accuracy rate under different number of training samples, where our IPU extremely beats other methods, which are tested on the images with all VIV. Apparently, a few traditional methods can't achieve good recognition performances when images experience realistic VIV through bio-visual systems.

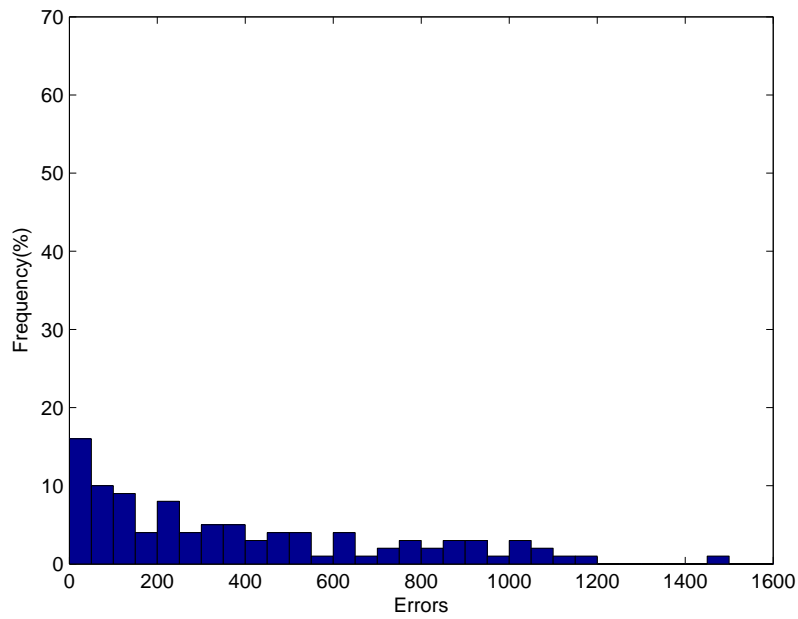
## Summary

In this chapter, we dedicate in exploring how to develop and utilize IPU to recognize simple patterns and face images for a general purpose and representation association machine. Inspired by bio-visual systems, IPU is designed as a gateway for information processing in recognition system. Not limited to recognizing low resolution digits and hyper-acuity tasks [24], in this chapter, we put efforts in designing a more integrated system solving the problem of recognizing simple pattern and face images. The quality of the patterns and face images have been severely degraded, which mimic the VIV experienced by human visual systems. The results show that 1) our IPU can recognize simple pattern images in different shapes and sizes reliably despite the image quality being very poor; 2) our IPU presents an excellent multi-class recognition performance in AT&T ORL database, comparable to the regular recognition methods for images without any quality degradation. 3) A bunch of methods have been proposed and compared for improving the performance of IPU, e.g. constructing specific LDPC codes with large minimum girth, designing various detecting

and power scaling method, etc. Our results demonstrate that our GPRAM Image Processing Unit performs the similar behavior to human visual systems which could recognize heavily distorted low-resolution human faces and its performance in terms of recognition successful rate and false alarm rate is almost as good as those achieved from state-of-the-art machine learning algorithms on images without severe distortion. This indicates a potential new research direction for future understanding of both GPRAM systems and human visual systems.

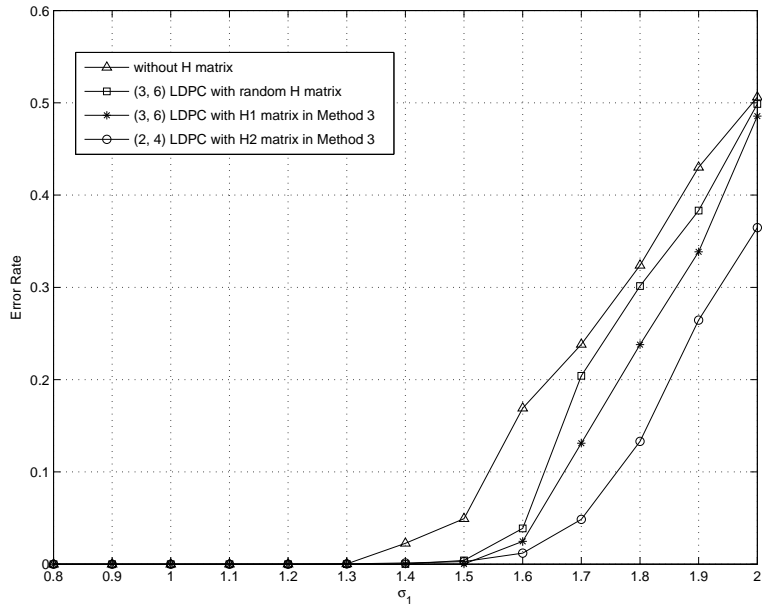


(a)

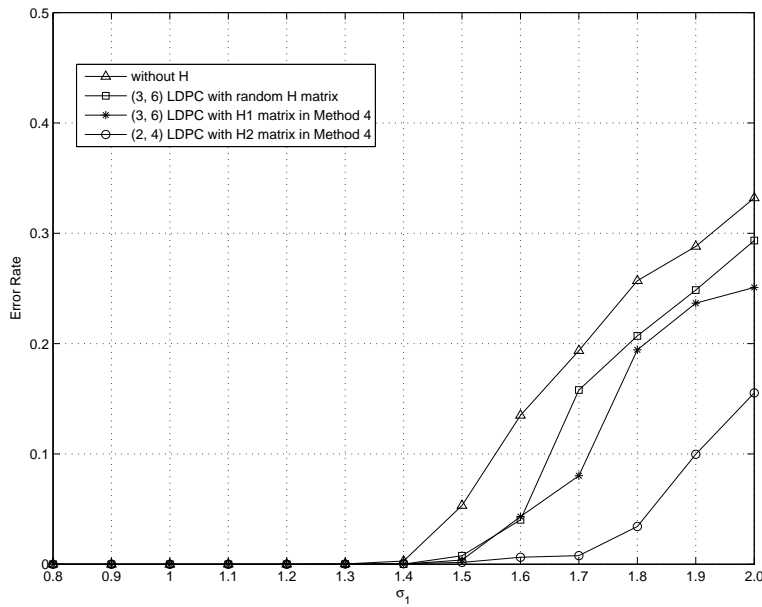


(b)

Figure 4.9: Histogram for 10000 Trials in Independent Number of Noise Seeds at  $\sigma_{a1} = 1.5$ ,  $\sigma_{a2} = 2.25$  for LDPC Codes (a) With  $H$  Matrix and (b) Without  $H$  Matrix



(a)



(b)

Figure 4.10: Impact of LDPC codes with Different Large Minimum Girth in (a) Power Scaling Method 3 and (b) Power Scaling Method 4

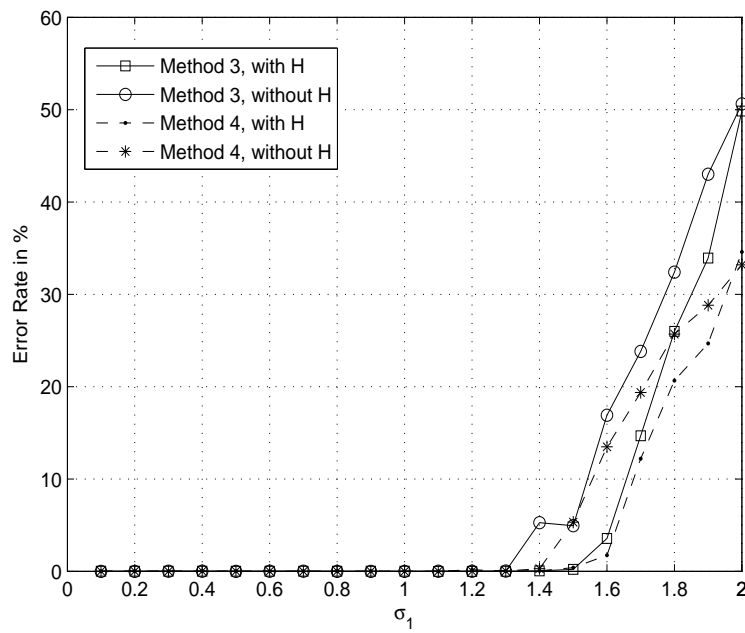


Figure 4.11: Error Rates for  $10^5$  Trials in Face Images with the Variation of  $\sigma_{a1}$

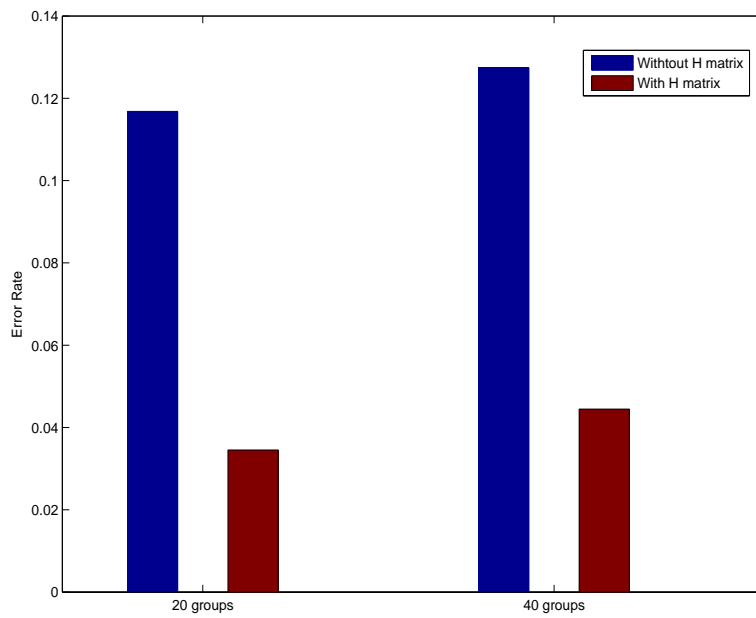


Figure 4.12: Error Rates for Testing on One Out of 20 and One Out of 40 Images Comparing Between With  $H$  and Without  $H$  matrix

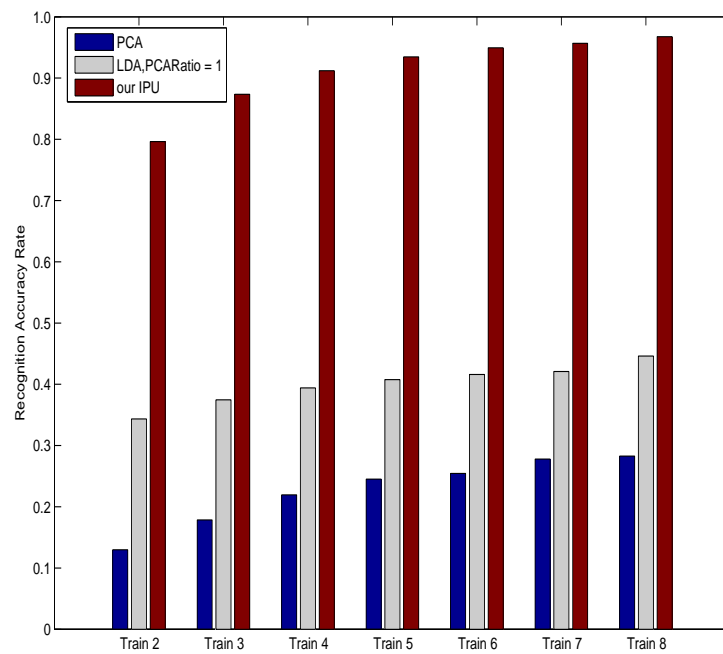


Figure 4.13: Recognition Accuracy Rate Comparing with Other Methods Tested On the Images with VIV

## CHAPTER 5: OPTIMIZING MPSK, MDPSK AND DUAL-RING QAM SIGNALING WITH NON-EQUAL SYMBOL PROBABILITIES

In this chapter we derive formulas to optimize signal constellations, decision regions, and symbol error rates for  $M$ -ary PSK,  $M$ -ary DPSK, and dual-ring QAM signaling with non-equal symbol probabilities over AWGN channels. Several methods have been introduced and developed. One is an optimizing method in which  $2M$  parameters ( $M$  decision constellations and  $M$  decision threshold) have been optimized, the other one is a simplified method in which only  $M$  decision constellations have been optimized. In each modulation scheme, the performance is evaluated and compared in symbol error rate at the same information bit energy. The results show that the optimizing system has lower error probabilities than two conventional systems, 1) a system with non-equiprobable symbols using source coding; 2) a system with non-equiprobable symbols using equally spaced constellation. Several examples presented in this chapter elaborate that the improvements are dramatic. Specially, it is shown numerically the approximately optimal dual-ring QAM system with non-equal symbol probabilities leads to performance gain around 2.8 dB comparing conventional systems.

### Signals and Systems of MPSK and MDPSK

#### *MPSK*

The constellation for an  $M$ -ary PSK signaling is shown in Fig.5.1(a). In this communication system, symbol  $\mathbf{s}_m = (\sqrt{\varepsilon} \cos \phi_m, \sqrt{\varepsilon} \sin \phi_m)$  is transmitted every  $T$  seconds from the set  $\{1, 2, \dots, M\}$  with probability  $0 \leq P_m \leq 1$ ,  $\sum_{m=1}^M P_m = 1$ , where  $\varepsilon$  is energy per symbol,  $\phi_m$  denotes the phase of each symbol where  $\phi_1 = 0$ . When symbols are transmitted with equal symbol probabilities,



each constellation point is generated by an evenly distributed array of points on the unit circle with phase  $\phi_m = 2\pi(m - 1)/M$ , where  $m = 1, 2, \dots, M$ . When symbol probabilities  $\mathbf{P} = \{P_m\}$  are non-equiprobable,  $\{\phi_m\}$  and the corresponding decision thresholds of  $\{s_m\}$  should be optimized to achieve the minimal symbol error rate of the whole system. This is the main focus of this chapter.

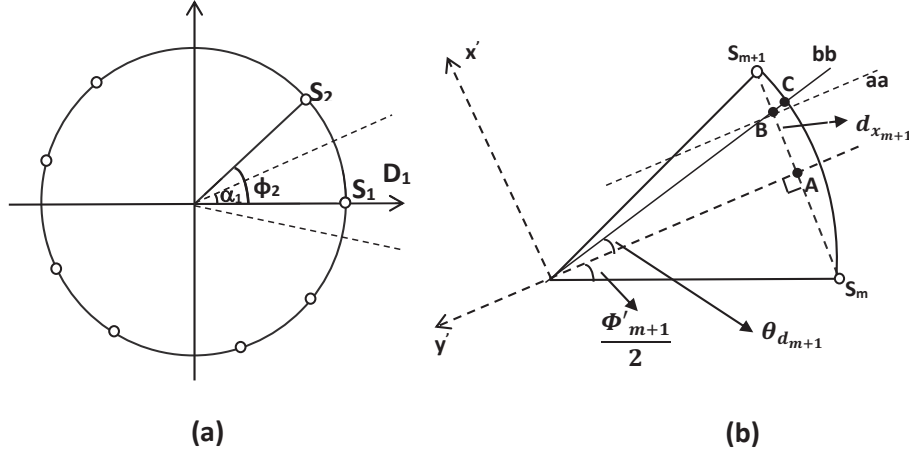


Figure 5.1: (a) Signal Constellation for  $M$ -PSK with Non-uniform Symbols; (b) Signal Constellation for  $M$ -PSK in New Coordinates for Method 1

Assuming the channel is an additive white Gaussian noise (AWGN) channel and a matched filter is used in receiver. When  $s_1 = (\sqrt{\varepsilon}, 0)$  is transmitted, the received vector is given by

$$\mathbf{r} = (r_1, r_2) = (\sqrt{\varepsilon} + n_1, n_2) \quad (5.1)$$

in which  $r_1$  and  $r_2$  are independent Gaussian random variables with variance  $\sigma^2 = N_0/2$  and means  $\sqrt{\varepsilon}$  and 0, respectively. Hence, we have [80]

$$p(r_1, r_2) = \frac{1}{\pi N_0} \exp \left[ -\frac{(r_1 - \sqrt{\varepsilon})^2 + r_2^2}{N_0} \right] \quad (5.2)$$

In order to simplify the calculation, the decision region  $D_m$  could be described in polar coordinates.

So  $(r_1, r_2)$  is transformed as

$$\begin{aligned} V &= \sqrt{r_1^2 + r_2^2} \\ \Theta &= \arctan \frac{r_2}{r_1} \end{aligned} \quad (5.3)$$

from which the joint PDF of  $V$  and  $\Theta$  can be derived as

$$p_{V,\Theta}(v, \theta) = \frac{v}{\pi N_0} \exp \left[ -\frac{v^2 - 2v\sqrt{\varepsilon} \cos \theta + \varepsilon}{N_0} \right] \quad (5.4)$$

By integrating  $V$ , the marginal PDF of  $\Theta$  is derived as

$$p_{\Theta_{MPSK}}(\theta) = \int_0^\infty p_{V,\Theta}(v, \theta) dv = \frac{1}{2\pi} \exp(-\gamma_s \sin^2 \theta) \int_0^\infty v \exp \left[ -\frac{(v - \sqrt{2\gamma_s} \cos \theta)^2}{2} \right] dv \quad (5.5)$$

where  $\gamma_s$  is defined as *SNR per symbol* in which  $\gamma_s = \varepsilon/N_0$ .

In Fig.5.1 (a), suppose the decision region  $D_m$  of  $s_m$  is divided by lines with central angle of  $\alpha_m$ .  $D_m$  is described as  $D_m = \{\theta : \theta_m^- < \theta < \theta_m^+\}$ . Therefore, the symbol error rate for MPSK system with non-equal probable symbols is given by

$$P_{MPSK}(e) = 1 - \sum_{m=1}^M P_m \int_{\theta_m^-}^{\theta_m^+} p_{\Theta_{MPSK}}(\theta) d\theta \quad (5.6)$$

where

$$\begin{aligned} \theta_m^+ &= \alpha_m - \phi_m \text{ if } m = 1, \dots, M \\ \theta_m^- &= \begin{cases} \alpha_M - 2\pi & \text{if } m = 1 \\ \alpha_{m-1} - \phi_m & \text{if } m = 2, \dots, M \end{cases} \end{aligned} \quad (5.7)$$

## MDPSK

Due to the phase difference [80], when symbol  $\mathbf{s}_1 = (\sqrt{\varepsilon}, 0)$  is transmitted, the received vector in (5.1) could be expressed as

$$\begin{aligned} \mathbf{r} &= (r_1, r_2) \\ &= [\sqrt{\varepsilon} + \mathbf{Re}(n_k + n_{k-1}^*), \mathbf{Im}(n_k + n_{k-1}^*)] \end{aligned} \quad (5.8)$$

where  $r_1$  and  $r_2$  are uncorrelated gaussian random variables with identical variance  $\sigma_n^2 = N_0$ . So similar to the phase coherent MPSK system, the PDF of  $\Theta$  in (5.5) could be expressed as

$$\begin{aligned} p_{\Theta_{DMPSK}}(\theta) &= \frac{1}{2\pi} \exp \left[ -\frac{\gamma_s \sin^2 \theta}{2} \right] \\ &\int_0^\infty v \exp \left[ -\frac{(v - \sqrt{\gamma_s} \cos \theta)^2}{2} \right] dv \end{aligned} \quad (5.9)$$

Therefore, the symbol error rate for MDPSK system is also given by

$$P_{DMPSK}(e) = 1 - \sum_{m=1}^M P_m \int_{\theta_m^-}^{\theta_m^+} p_{\Theta_{DMPSK}}(\theta) d\theta \quad (5.10)$$

where  $\theta_m^-$  and  $\theta_m^+$  have the same range defined in (5.7).

Now this problem is almost identical to the problem of MPSK. The only difference is  $\gamma_s$  in MDPSK is half of that in MPSK since the noise variance of MDPSK is twice as large as that of MPSK.

Hence, the problem of optimizing MPSK and MDPSK signalling is simplified to separately finding an optimal system for MPSK or MDPSK, i.e., seeking signal constellation locations  $\{\phi_m\}$  and decision thresholds  $\{\alpha_m\}$ , to reach the minimal average symbol error probabilities  $P(e)$ . The solution to the problem for MPSK and MDPSK system with equal probabilities was described in

details in [80].

### Optimal Detection and Error Probability for MPSK and MDPSK Signaling

Based on the previous analysis of MPSK and MDPSK system, we can see the problem of these two systems are similar except the noise variance. So we can solve these two signaling using the same methodology. In this section, given the probability set  $\{P_m\}$ , we are looking for optimal solutions of  $\{\phi_m\}$ ,  $\{\alpha_m\}$  in  $\{D_m\}$ , and finally get error probability  $P(e)$ . Two methods have been thoroughly considered. The first one is a simplified method, in which decision threshold is approximately optimized based on  $M$  individual 2ASK system. Then we near optimally compute  $\{\phi_m\}$  and  $\{\alpha_m\}$ . The second method is the brute-force optimizing method given the initial value of  $\{\phi_m\}$  and  $\{\alpha_m\}$  calculated from the first one. It presents more accurate result.

#### *Method 1: Simplified Method*

Our observation for the first method is partially based on 2-ASK system in [97], which is illustrated in Fig.5.1 (b). In [97] the optimal decision rule of 2ASK system is if  $s_m < r \leq s_{m+1}$  and  $P_m p_{r|m}(r|m) \geq P_{m+1} p_{r|m+1}(r|m+1)$ , then  $\hat{m} = m$ ; otherwise,  $\hat{m} = m + 1$ . In this method, our whole MPSK system could be considered as  $M$ -2ASK systems. In Fig.5.1 (b), for each individual 2ASK system  $\{s_m, s_{m+1}\}$ , if  $x, y$  axes are rotated to  $x'$  and  $y'$  axes, the noise in  $y'$  will not affect the decision point along  $x'$  axis. If  $P_m = P_{m+1}$ , the decision region should be divided at the middle point  $A$  along the line linked points  $s_m$  and  $s_{m+1}$ . This is the optimal decision point for symbols with equal probabilities. If  $P_m \neq P_{m+1}$ , the decision region will not be divided at the middle point. Without loss of any generality, let us assume  $P_m \geq P_{m+1}$ . Intuitively, the decision point should move toward  $s_{m+1}$ , say point  $B$ . If this is 2ASK signaling, then we can quickly find point

$B$  and draw line (line  $aa$ ) perpendicular to  $x'$  axis, which divides two decision regions. But now, line  $aa$  does not pass the origin of  $(x', y')$  axes required for MPSK signaling. Therefore, in order to meet decision region requirement for MPSK, we need to divide decision region by drawing a line through points  $B$  and centre  $O$  (say line  $bb$ ). The intersection point (point  $C$ ) is a little biased toward  $s_{m+1}$  point. So its performance should be worse than the ASK signaling (i.e., comparing with the decision region divided by line  $aa$  in 2ASK). This insight forms the basis for the first method.

Now we formulate the first method. According to [97], decision point of  $\{s_m, s_{m+1}\}$  could be specified as

$$\mathbf{d}_{m+1} = \begin{cases} \frac{s_m + s_{m+1}}{2} + \frac{0.5N_0 \ln(P_m/P_{m+1})}{s_{m+1} - s_m} & \text{if } m = 1, \dots, M-1 \\ \frac{s_M + s_1}{2} + \frac{0.5N_0 \ln(P_M/P_1)}{s_1 - s_M} & \text{if } m = M \end{cases} \quad (5.11)$$

And we define  $\phi'_m$  as follows as the central angle between each two constellation points, that is,

$$\phi'_{m+1} = \begin{cases} \phi_{m+1} - \phi_m & \text{if } m = 1, \dots, M-1 \\ 2\pi - \phi_M & \text{if } m = M \end{cases} \quad (5.12)$$

Obviously,

$$\sum_{m=1}^M \phi'_{m+1} = 2\pi \quad (5.13)$$

Correspondingly, by adding this constrain, (5.6) and (5.10) in Section 5 could be developed as

$$\Omega = P(e) + \lambda \left( \sum_{m=1}^M \phi'_{m+1} - 2\pi \right) \quad (5.14)$$

Therefore for each new coordinate  $(x', y')$ , in the pair set  $\{\{s_1, s_2\}, \dots, \{s_{M-1}, s_M\}, \{s_M, s_1\}\}$ , each pair could be written as  $\left\{ \left( -\sqrt{\varepsilon} \sin \frac{\phi'_{m+1}}{2}, -\sqrt{\varepsilon} \cos \frac{\phi'_{m+1}}{2} \right), \left( \sqrt{\varepsilon} \sin \frac{\phi'_{m+1}}{2}, -\sqrt{\varepsilon} \cos \frac{\phi'_{m+1}}{2} \right) \right\}$  ( $m=1, \dots,$

M). Since the noise in  $y'$  axis will not affect the decision point along  $x'$  axis, by substituting the pair set into (5.11), on  $x'$  axis we obtain

$$d_{x_{m+1}} = \begin{cases} \frac{0.5N_0 \ln(p_m/p_{m+1})}{2\sqrt{\varepsilon} \sin(\phi'_{m+1}/2)} & \text{if } m = 1, \dots, M-1 \\ \frac{0.5N_0 \ln(p_M/p_1)}{2\sqrt{\varepsilon} \sin(\phi'_{M+1}/2)} & \text{if } m = M \end{cases} \quad (5.15)$$

In Fig. 5.1 (b), by using our knowledge of geometry,  $\theta_{d_{m+1}}$  ( $m = 1, \dots, M$ ) is defined as the central angle  $\angle BOA$ ,

$$\theta_{d_{m+1}} = \arctan \frac{d_{x_{m+1}}}{\sqrt{\varepsilon} \cos(\phi'_{m+1}/2)} \quad (5.16)$$

Therefore, when transforming to MPSK and MDPSK system, the decision region  $D_m$  for each  $s_m$  in (5.7) could be derived as

$$\theta_m^+ = \frac{1}{2}\phi'_{m+1} + \theta_{d_{m+1}} \quad \text{if } m = 1, \dots, M$$

$$\theta_m^- = \begin{cases} -\frac{1}{2}\phi'_{M+1} + \theta_{d_{M+1}} & \text{if } m = 1 \\ -\frac{1}{2}\phi'_m + \theta_{d_m} & \text{if } m = 2, \dots, M \end{cases} \quad (5.17)$$

Hence, by substituting (5.17) in (5.6), we are now able to minimize (5.14) by using the method of Lagrange multipliers. In other words, we are able to find the optimal  $\{\phi'_{m+1}\}$  by taking  $d\Omega/d\phi'_{m+1}$

for  $m = 1, \dots, M$  and  $d\Omega/d\lambda$ , we then obtain expressions with respect to  $\phi'_{m+1}$  as follows

$$\begin{aligned}
& \lambda - \frac{1}{2} [P_m p(\frac{1}{2}\phi'_{m+1} + \theta_{d_{m+1}}) + P_{m+1} p(-\frac{1}{2}\phi'_{m+1} + \theta_{d_{m+1}})] \\
& - \frac{d\theta_{d_{m+1}}}{d\phi'_{m+1}} [P_m p(\frac{1}{2}\phi'_{m+1} + \theta_{d_{m+1}}) - P_{m+1} p(-\frac{1}{2}\phi'_{m+1} + \theta_{d_{m+1}})] \\
& = 0, \text{ if } m = 1, 2, \dots, M - 1 \\
& \lambda - \frac{1}{2} [P_M p(\frac{1}{2}\phi'_{M+1} + \theta_{d_{M+1}}) + P_1 p(-\frac{1}{2}\phi'_{M+1} + \theta_{d_{M+1}})] \\
& - \frac{d\theta_{d_{M+1}}}{d\phi'_{M+1}} [P_M p(\frac{1}{2}\phi'_{M+1} + \theta_{d_{M+1}}) - P_1 p(-\frac{1}{2}\phi'_{M+1} + \theta_{d_{M+1}})] \\
& = 0, \text{ if } m = M
\end{aligned} \tag{5.18}$$

At last, by substituting (5.15) (5.16) into (5.18),  $\{\phi'_m\}$  could be solved. According to (5.12) and (5.7),  $\{\phi_m\}$  and  $\{\alpha_m\}$  will be also obtained.

### *Method 2: Optimized Method*

In this subsection, we introduce a method which directly finds the decision region of MPSK/MDPSK.

Based on the connection of  $\alpha_m$  and  $\phi_m$  shown in (5.7), the symbol error probability  $P(e)$  in (5.6) could be specified as

$$\begin{aligned}
P(e) &= 1 - P(c) \\
&= 1 - P_1 \int_{\alpha_M - 2\pi}^{\alpha_1} p(\theta) d\theta - \sum_{m=1}^{M-1} \int_{\alpha_m - \phi_{m+1}}^{\alpha_{m+1} - \phi_{m+1}} P_{m+1} p(\theta) d\theta
\end{aligned} \tag{5.19}$$

In order to minimize  $P(e)$  and get optimized  $\{\alpha_m\}$  and  $\{\phi_m\}$ , we take derivation of  $P(e)$  as  $dP(e)/d\alpha_m$  ( $m = 1, \dots, M$ ) and  $dP(e)/d\phi_m$  ( $m = 2, \dots, M$ ) in (5.19). Correspondingly, we will obtain  $M$  expressions as the function of  $\alpha_m$ , as well as  $M - 1$  expressions as the function of  $\phi_m$ ,

which are showed separately in (5.20) and (5.21).

$$\begin{aligned}
& - P_1 p(\alpha_1) + P_2 p(\alpha_1 - \phi_2) = 0, \text{ if } m = 1 \\
& - P_m p(\alpha_m - \phi_m) \\
& + P_{m+1} p(\alpha_m - \phi_{m+1}) = 0, \text{ if } m = 2, \dots, M - 1 \\
& - P_M p(\alpha_M - \phi_M) + P_1 p(\alpha_M - 2\pi) = 0, \text{ if } m = M
\end{aligned} \tag{5.20}$$

$$\begin{aligned}
& P_m p(\alpha_m - \phi_m) \\
& - P_m p(\alpha_{m-1} - \phi_m) = 0, \text{ if } m = 2, \dots, M
\end{aligned} \tag{5.21}$$

For both methods, we use Matlab to find solutions to these nonlinear equations. In Method 1, the number of variables ( $\phi'_m$ ) needs to set up initial points is  $M$ ; whereas in Method 2, this number increases to  $2M$  ( $\phi_m$  and  $\alpha_m$ ) which gains the difficulty in solving non-linear equations when  $M$  is very large. So in Method 2, we employ the results of  $\{\phi_m\}$  and  $\{\alpha_m\}$  we got from Method 1 as initial points. Finally, an optimized solution in Method 2 is able to generate, which presents slightly better results shown in the subsection of Numerical Result.

Comparing between Method 1 and 2, Method 1 is simpler. Method 2 is more computationally complex. Method 1 may also use to find good initial points, then we run Method 2 to get more accurate results.



## Extension to Optimizing Dual-ring QAM Signaling with Non-equal Symbol Probabilities

In this section, a dual-ring QAM signaling system with non-equal symbol probabilities will be discussed. At first, a general dual-ring QAM system will be introduced. Then how to solve decision boundaries of  $M$  constellation points on dual rings will be developed. Moreover, based on the methods we developed in MPSK/MDPSK, an optimizing solution to optimize dual-ring QAM will be presented.

### *Signals and Systems of Dual-ring QAM*

Considering M-QAM system, there are many possibilities for designing M-QAM signal points. In this problem, we shall consider the star QAM signal constellation shown in Fig. 5.2, which consists of two rings with two amplitude levels and  $M/2$  points located on each one. Signal points are non-equiprobable.

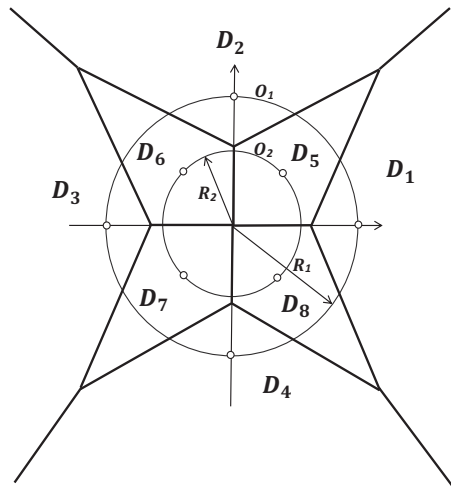


Figure 5.2: Signal Constellation for Dual-ring QAM

Suppose  $M/2$ -point signal set  $\mathbf{S}_{O_1} = \{\mathbf{s}_1, \dots, \mathbf{s}_{M/2}\}$  located on large outer ring  $O_1$  with high amplitude; set  $\mathbf{S}_{O_2} = \{\mathbf{s}_{M/2+1}, \dots, \mathbf{s}_M\}$  located on small inner ring  $O_2$  with low amplitude. Each symbol is with probability  $0 \leq P_m \leq 1 (m = 1, \dots, M)$ . Suppose the coordinates  $(s_{mx}, s_{my})$  for each signal point  $\mathbf{s}_m$  satisfies with

$$s_{mx}^2 + s_{my}^2 = \begin{cases} R_1^2 & \text{if } m = 1, \dots, M/2 \\ R_2^2 & \text{if } m = M/2 + 1, \dots, M \end{cases} \quad (5.22)$$

where  $R_1$  is the radius of outer ring  $O_1$  and  $R_2$  is the radius of inner ring  $O_2$ , and  $R_1 > R_2$ .

Let's make the sum probability of  $\mathbf{S}_{O_1}$  be  $P_{11} = \sum_{m=1}^{M/2} P_m$  and sum probability of  $\mathbf{S}_{O_2}$  be  $P_{22} = \sum_{m=M/2+1}^M P_m$ , so the average transmitted signal energy is

$$\varepsilon = R_1^2 P_{11} + R_2^2 P_{22} \quad (5.23)$$

where  $P_{11} < P_{22}$ . And

$$0 < R_2 < \sqrt{\varepsilon} < R_1 < \sqrt{\frac{\varepsilon}{P_{11}}} \quad (5.24)$$

When symbol  $\mathbf{s}_m$  is transmitted, at the receiver,

$$p_{QAM}(r_1, r_2 | \mathbf{s}_m) = \frac{1}{\pi N_0} \exp \left[ -\frac{(r_1 - s_{mx})^2 + (r_2 - s_{my})^2}{N_0} \right] \quad (5.25)$$

Assume  $D_m$  is the decision region of each symbol  $\mathbf{s}_m$ , so the symbol error rate for QAM system is given by

$$P_{QAM}(e) = 1 - \sum_{m=1}^M P_m \iint_{D_m} p_{QAM}(r_1, r_2 | \mathbf{s}_m) dr_1 dr_2 \quad (5.26)$$

Apparently, the problem has been clarified as follows: for fixed symbol probabilities  $\{P_m\}$ , given signal energy  $\varepsilon$ , finding proper symbol coordinates  $\{(s_{mx}, s_{my})\}$  as well as their decision region

$\{D_m\}$ , to get  $\min P_{QAM}(e)$ . As we see, the points could be moving on any place of the ring and the sizes of both rings are uncertain, so how to decide decision boundaries of each signal point is important.

### *Computing Decision Region for Dual-ring QAM System*

Fig. 5.2 also shows a particular example of decision boundaries for eight-point QAM constellation with equal symbol probabilities [100]. For each point  $\mathbf{s}_m$ , any other point  $\mathbf{s}_n$  will generate a decision region with  $\mathbf{s}_m$  as a 2ASK system, labeled  $D_{mn}$ . So  $D_m$  of each  $\mathbf{s}_m$  are the intersection of sets  $\{D_{mn}\}$ , i.e.,  $D_m = D_{m1} \cap D_{m2} \dots \cap D_{mn}$ , where  $n \neq m$ . If received vector  $\mathbf{r} \in D_m$ , then  $\hat{n} = m$ .

In general case, we obtain  $\{D_{mn}\}$  with any probability set  $\{P_m\}$  as follows: suppose any two constellation points  $\mathbf{s}_m = (s_{mx}, s_{my})$ ,  $\mathbf{s}_n = (s_{nx}, s_{ny})$  and decision point  $\mathbf{d}_{mn} = (d_{mnx}, d_{mny})$ , similar to (5.11),  $\mathbf{d}_{mn}$  can be specified as

$$\mathbf{d}_{mn} = \left( \frac{s_{mx} + s_{nx}}{2} + \frac{0.5 \ln(P_m/P_n)}{s_{nx} - s_{mx}}, \right. \\ \left. \frac{s_{my} + s_{ny}}{2} + \frac{0.5 \ln(P_m/P_n)}{s_{ny} - s_{my}} \right) \quad (5.27)$$

For any  $\mathbf{r} = (r_1, r_2)$ ,  $D_{mn}$  of symbol  $\mathbf{s}_m$  could be described as  $D_{mn} = \{\overrightarrow{\mathbf{s}_m \mathbf{s}_n} \cdot \overrightarrow{\mathbf{d}_{mn} \mathbf{r}} < 0\}$  which is

$$r_1(s_{nx} - s_{mx}) + r_2(s_{ny} - s_{my}) - \ln(P_m/P_n) \\ - \frac{1}{2}(s_{nx}^2 + s_{ny}^2) + \frac{1}{2}(s_{mx}^2 + s_{my}^2) < 0 \quad (5.28)$$

where  $n, m = 1, \dots, M, n \neq m$ . And  $s_{nx}^2 + s_{ny}^2, s_{mx}^2 + s_{my}^2$  follows the rule in (5.22).

Then  $D_m$  will be generated as the intersection of  $D_{mn}$  ( $n = 1, \dots, M$  and  $n \neq m$ ), which is

composed of  $M - 1$  inequalities of (5.28). For instance, (5.29) shows an example of  $D_1$ .

$$\begin{cases} r_1(s_{nx} - s_{1x}) + r_2(s_{ny} - s_{1y}) - \ln\left(\frac{P_1}{P_n}\right) < 0 \\ \text{if } n = 2, \dots, M/2 \\ r_1(s_{nx} - s_{1x}) + r_2(s_{ny} - s_{1y}) - \ln\left(\frac{P_1}{P_n}\right) - \frac{R_2^2}{2} + \frac{R_1^2}{2} < 0 \\ \text{if } n = M/2 + 1, \dots, M \end{cases} \quad (5.29)$$

After substituting (5.28) in (5.26), since the limits of the integral is neither a real number nor a continuous function, without knowing the shape and coordinates of the region  $D_m$ , (5.26) cannot be presented as a closed-form expression. In this case, since  $p_{QAM}(r_1, r_2|s_m)$  is continuous in each  $D_m$ , we could apply the Riemann integral in the two-dimensional space. And also, an identifier symbol  $I_m$  has been brought in for indicating if the sampling points of Riemann integral located in  $D_m$  or not.

For  $(r_1, r_2) \in \mathbf{R}^2$ , suppose either of  $r_1$  and  $r_2$  is on a closed, large bounded interval  $[A_{min}, A_{max}]$ . After dividing  $N$  subintervals on  $[A_{min}, A_{max}]$ , the length of one subinterval is

$$\Delta_{r_1} = \Delta_{r_2} = \frac{A_{max} - A_{min}}{N} \quad (5.30)$$

Also,  $\forall \xi_i \in \Delta_{r_1i}, \forall \eta_j \in \Delta_{r_2j}, i, j \in \{1, 2, \dots, N\}$

$$\begin{aligned} \xi_i &= i\Delta_{r_1} - \Delta_{r_1}/2 + A_{min} \\ \eta_j &= j\Delta_{r_2} - \Delta_{r_2}/2 + A_{min} \end{aligned} \quad (5.31)$$

Therefore, (5.26) could be derived as

$$P_{QAM}(e) \approx 1 - \sum_{i=1}^N \sum_{j=1}^N \sum_{m=1}^M P_m p_{QAM}(\xi_i, \eta_j | s_m) I_m(\xi_i, \eta_j) \Delta_{r1} \Delta_{r2} \quad (5.32)$$

where

$$I_m(\xi_i, \eta_j) = \begin{cases} 1 & \text{if } (\xi_i, \eta_j) \in D_m \\ 0 & \text{else} \end{cases} \quad (5.33)$$

Hence, when we set appropriate  $A_{min}$  and  $A_{max}$ , the larger sampling points  $N$  chosen, the more accurate  $P_{QAM}(e)$  we will achieve.

### *Optimizing Symbol Error Rate for Dual-ring QAM*

Based on the aforementioned illustration, for fixed  $\{P_m\}$  and  $\varepsilon$ , we are able to compute the symbol error rate based on any  $\{D_m\}$  and  $\{(s_{mx}, s_{my})\}$ . In this subsection, we engage in seeking an optimizing solution of  $\{(s_{mx}, s_{my})\}$  and corresponding  $\{D_m\}$  to get the min  $P_{QAM}(e)$  for dual-ring QAM system with non-equal probabilities.

To solve dual-ring QAM signaling optimization problem, we take two steps: **Step (1)** Use MPSK signaling optimization method in Section 5 and 5 to obtain optimal constellation for each ring with  $M/2$  symbols.  $P_m$  of symbol could be redefined as

$$P'_m = \begin{cases} \frac{P_m}{P_{11}} & \text{if } m = 1, \dots, M/2 \\ \frac{P_m}{P_{22}} & \text{if } m = M/2 + 1, \dots, M \end{cases} \quad (5.34)$$

That is, using Method 2 derived in section 5, given certain value of  $R_1$  ( $R_2$ ) and  $\{P'_m\}$ , we calculate  $\{\phi_m\}$  of symbols separately on each ring, then compute  $\{(s_{mx}, s_{my})\}$ .

**Step (2)** Fix the outer ring  $O_1$ , i.e.,  $\phi_1 = 0$  and rotate the inner ring  $O_2$  with angle  $\beta = \phi_{M/2+1} - \phi_1$ , for each given  $\beta$  we can compute  $(s_{mx}, s_{my})$  as :

$$\begin{cases} s_{mx} = R_1 \cos \phi_m \\ s_{my} = R_1 \sin \phi_m \end{cases} \quad \text{if } m = 1, \dots, M/2 \quad (5.35)$$

$$\begin{cases} s_{mx} = R_2 \cos(\phi_m + \beta) \\ s_{my} = R_2 \sin(\phi_m + \beta) \end{cases} \quad \text{if } m = M/2 + 1, \dots, M \quad (5.36)$$

Since (5.23),  $P_{QAM}(e)$  could be optimized given  $R_1$  and  $\beta$ . After searching minimum  $P_{QAM}(e)$  over a range of  $R_1$  and  $\beta$ , we optimize dual-ring QAM signaling.

### Numerical Results

In this section, we will show our numerical results for MPSK systems with  $M = 3, 4, 8$ , MDPSK system with  $M = 8$ , dual-ring QAM system with  $M = 8$ . For MPSK/MDPSK, we evaluate symbol error rate as a function of information bit energy to noise density ratio ( $\varepsilon_{cb}/N_0$ ) in four system cases, 1) a system with non-equal symbol probabilities using source coding, 2) a system with non-equal symbol probabilities in equal space constellation, 3) a system with non-equal symbol probabilities with optimal MPSK/MDPSK signaling using Method 1 and 4) using Method 2. For dual-ring QAM, system 1), 2) and 5) systems with non-equal probabilities with optimizing dual-ring QAM signaling are thoroughly considered.

When comparing in symbol error rate per information bit, the average symbol energy  $\varepsilon$  will increase to  $\varepsilon_c = \varepsilon \log_2(M)/H$ , where  $H = -\sum_{m=1}^M P_m \log_2(P_m)$  is the entropy of the original symbols. And bit energy per information bit will become  $\varepsilon_{cb} = \varepsilon/H$  instead of  $\varepsilon_b = \varepsilon/\log_2(M)$ .

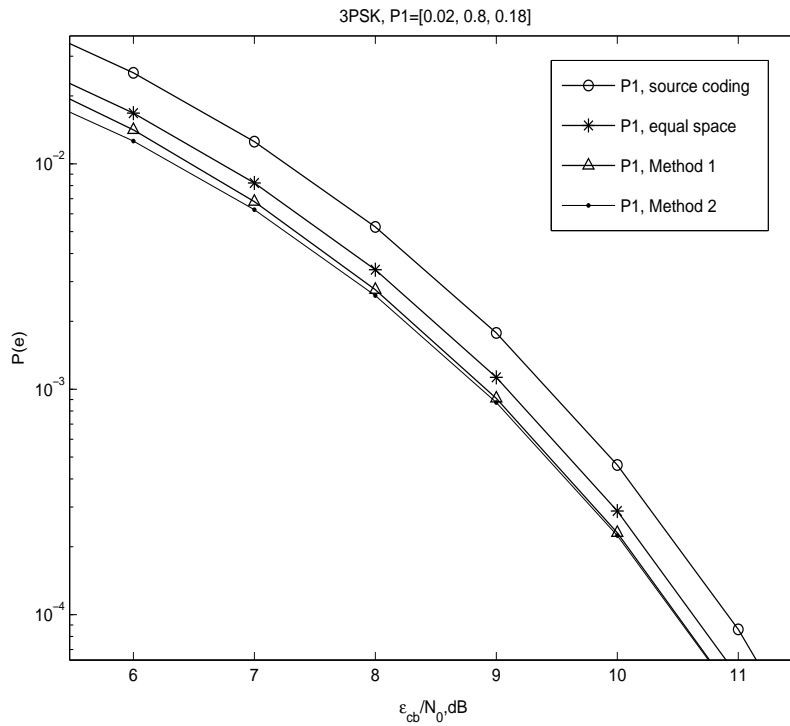


Figure 5.3:  $P(e)$  of Optimized and Non-optimized 3PSK Signaling Per Information Bit Energy

Specifically in case 1), in a conventional system, symbols with non-equal probabilities are first converted using source coding [31] into equiprobable symbols with a reduced symbol rate. These equiprobable symbols are then transmitted as equiprobable signals with an increased average energy. Furthermore, in case 2), symbols are fixed in equal-spaced constellation, but the decision regions are decided by symbol probabilities which are non-equiprobable.

### Numerical Result of MPSK

#### a. 3PSK

We consider  $M=3$  with  $\mathbf{P}=[0.02, 0.8, 0.18]$ . The entropy is  $H = -0.02 \log_2(0.02) - 0.8 \log_2(0.8) -$

$0.18 \log_2(0.18) = 0.8157$ . Fig. 5.3 shows  $P(e)$  for our optimal systems is smaller than  $P(e)$  for the equal space constellation (case 1), and conventional source coding system (case 2). And when comparing Method 1 (case 3) and Method 2 (case 4)), Method 2 shows slightly better than Method 1. Specifically, when  $\varepsilon_b/N_0 = 7$  dB, four cases are shown in Table 5.1. The final result for optimized 3PSK signaling using Method 1 is given in the third row, where we have  $P(e) = 0.00679$ ,  $\phi_m = [0, 2.061, 4.513]$ ,  $\alpha_m = [0.558, 3.566, 5.666]$ . And also, we are able to obtain the result for Method 2 as  $P(e) = 0.00625$ ,  $\phi_m = [0, 2.057, 4.547]$ ,  $\alpha_m = [0.651, 3.462, 5.632]$ , which is recorded in the fourth row. If we use the equal space constellation, then we have  $P(e) = 0.00822$  and other parameters are listed in the second row in Table 5.1. For 3PSK using source coding, we have  $P(e) = 0.0125$  and other parameters are listed in the first row.

#### b. 4PSK

Fig. 5.4 shows symbol error rate as a function of  $\varepsilon_{cb}/N_0$  per information bit for 4PSK system when  $\mathbf{P} = [0.08, 0.02, 0.75, 0.15]$ . It is clear to see the performance of two optimal methods also bid other two cases. When comparing Method 1 and 2, at high SNR, their curves are overlapped, so their performance are almost the same.

Table 5.2 presents four cases for 4PSK at  $\varepsilon_b/N_0 = 7$  dB. The corresponding entropy is  $H = -0.08 \log_2(0.08) - 0.02 \log_2(0.02) - 0.75 \log_2(0.75) - 0.15 \log_2(0.15) = 1.126$ . The optimized 4PSK signaling in Method 1 is presented in the third row, where  $P(e) = 0.00834$ ,  $\phi_m = [0, 1.240, 2.815, 4.718]$ ,  $\alpha_m = [0.749, 1.717, 3.916, 5.556]$ . The fourth row is 4PSK signaling in Method 2, where  $P(e) = 0.00826$ ,  $\phi_m = [0, 1.240, 2.816, 4.718]$ ,  $\alpha_m = [0.735, 1.744, 3.888, 5.548]$ . And also, when we use the equal space constellation,  $P(e) = 0.0108$  and other parameters are listed in the second row. Row 1 displays  $P(e) = 0.0174$  and other parameters for 4PSK using source coding.



c. 8PSK

Fig. 5.5 displays  $P(e)$  of optimized and non-optimal 8PSK signaling. Method 1 and 2 starts overlap after  $\varepsilon_b/N_0=7$  dB. It is clear to see the performance gain of  $P(e)$  in our optimal methods is dramatic. At  $P(e) = 10^{-2}$ , our proposed methods show approximately 0.7 dB gain comparing with the system using equal space constellation, and 1 dB gain than the system using source coding.

Table 5.1:  $P(e)$ ,  $\phi_m$  and  $\alpha_m$  for 3PSK at  $\varepsilon_b/N_0 = 7$  dB

| Case | $\mathbf{P}$      | $P(e)$  | $\phi_m$                  | $\alpha_m$                     |
|------|-------------------|---------|---------------------------|--------------------------------|
| 1)   | [0.02, 0.8, 0.18] | 0.0125  | [0, $2\pi/3$ , $4\pi/3$ ] | [ $\pi/3$ , $\pi$ , $5\pi/3$ ] |
| 2)   | [0.02, 0.8, 0.18] | 0.00822 | [0, $2\pi/3$ , $4\pi/3$ ] | [0.567, 3.349, 5.537]          |
| 3)   | [0.02, 0.8, 0.18] | 0.00679 | [0, 2.061, 4.513]         | [0.558, 3.566, 5.666]          |
| 4)   | [0.02, 0.8, 0.18] | 0.00625 | [0, 2.057, 4.547]         | [0.651, 3.462, 5.632]          |

Table 5.2:  $P(e)$ ,  $\phi_m$  and  $\alpha_m$  for 4PSK at  $\varepsilon_b/N_0 = 7$  dB

| Case | $\mathbf{P}$             | $P(e)$  | $\phi_m$                         | $\alpha_m$                                   |
|------|--------------------------|---------|----------------------------------|----------------------------------------------|
| 1)   | [0.08, 0.02, 0.75, 0.15] | 0.0174  | [0, $\pi/2$ , $\pi$ , $3\pi/2$ ] | [ $\pi/4$ , $3\pi/4$ , $5\pi/4$ , $7\pi/4$ ] |
| 2)   | [0.08, 0.02, 0.75, 0.15] | 0.0108  | [0, $\pi/2$ , $\pi$ , $3\pi/2$ ] | [0.908, 2.046, 4.069, 5.553]                 |
| 3)   | [0.08, 0.02, 0.75, 0.15] | 0.00834 | [0, 1.240, 2.815, 4.718]         | [0.749, 1.717, 3.916, 5.556]                 |
| 4)   | [0.08, 0.02, 0.75, 0.15] | 0.00826 | [0, 1.240, 2.816, 4.718]         | [0.735, 1.744, 3.888, 5.548]                 |

*Numerical Result of MDPSK*

Fig. 5.6 displays  $P(e)$  of optimized and non-optimal 8DPSK signaling in four cases. This figure shows our two methods have a significant improvement in  $P(e)$  when comparing with 8DPSK constellation using source coding as well as 8DPSK in equal space constellation. The result shows at symbol error rate of  $10^{-2}$ , our Method 2 is about 0.7 dB better than the curve of "p1, equal space" and at  $P(e) = 10^{-3}$ , it is 0.4 dB better.

### Numerical Result of Dual-ring QAM

Fig. 5.7 shows  $P(e)$  of optimizing and non-optimizing dual-ring 8QAM with equal and non-equal probabilities. Our optimizing method displays a dramatic gain in  $P(e)$  when comparing with other cases. In Fig. 5.7, at  $P(e) = 10^{-3}$ , the optimizing method we developed shows approximately 2.6 dB gain comparing the curve "P1, source coding", and approximately 2.8 dB gain comparing the curve "P1, equal space". Fig. 5.8 illustrates optimal constellation points  $\{s_m\}$  and decision regions  $\{D_m\}$  as an example when  $\mathbf{P}=[0.03, 0.01, 0.04, 0.02, 0.25, 0.15, 0.20, 0.30]$ ,  $\varepsilon_{cb}/N_0=10.17$  dB, which demonstrate why we can achieve this significant gain. In Fig. 5.8,  $R_1 = 1.85\sqrt{\varepsilon}$ ,  $R_2 = 0.855\sqrt{\varepsilon}$ ,  $\beta = 8\pi/25$ ,  $\{\phi_m\} = \{0, 1.491, 3.028, 4.678, 1.010, 2.537, 4.025, 5.636\}$ .

From all the results above, we can summarize for MPSK/ MDPSK/ dual-ring QAM signaling system as 1) when symbol probabilities are non-equiprobable, the optimal decision regions are not equal spaced. 2) The signal constellations and decision regions for non-equal symbol probabilities developed by our methods achieve a significant improvement in symbol error rate. 3) In dual-ring QAM,  $\{D_m\}$  is determined by  $\{s_m\}$ , and there is no closed-form expression of  $\{D_m\}$ , where kinds of optimizing methods cannot be applied there. So the solution we provide is an approximate optimizing method where we employed the optimizing location of  $s_m$  from optimal MPSK system. 4) If  $\mathbf{P}$  changes all the time, the complexity of our scheme will be high due to continuous updating constellations and decision threshold. However, if  $P_m$  is fixed, the only increase in complexity is the initial value for computations of constellation and decision thresholds. Once the computation has been done, the receiver complexity is identical to conventional signaling systems.

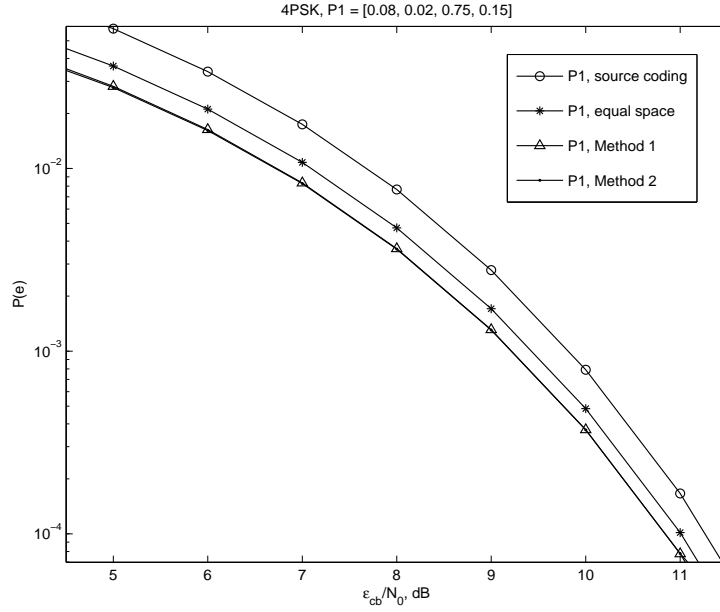


Figure 5.4:  $P(e)$  of Optimized and Non-optimized 4PSK Signaling Per Information Bit Energy

### Summary

In this chapter, we derive formulas to compute the optimizing signal constellation, decision regions, and symbol error rate for MPSK, MDPSK, dual-ring QAM with non-equal symbol probabilities. Several optimizing methods have been well developed and compared. The results show when comparing at the same information bit energy, all of our optimal systems have lower error probabilities than the conventional systems. Meanwhile, a novel methodology of pursuing approximately optimal dual-ring QAM system with non-equal symbol probabilities are also led in this chapter. The improvement reaches 2.8 dB, which shows that our approximately optimizing approach can deliver dramatic performance improvement. These improvement may be very attractive in wireless sensor systems where optimal source coding may not be able to implement. On the other side, considering the non-ideal construction and formats in bio-system, in the future, we could extend our current findings to accommodate or better understand bio-neural information processing system.

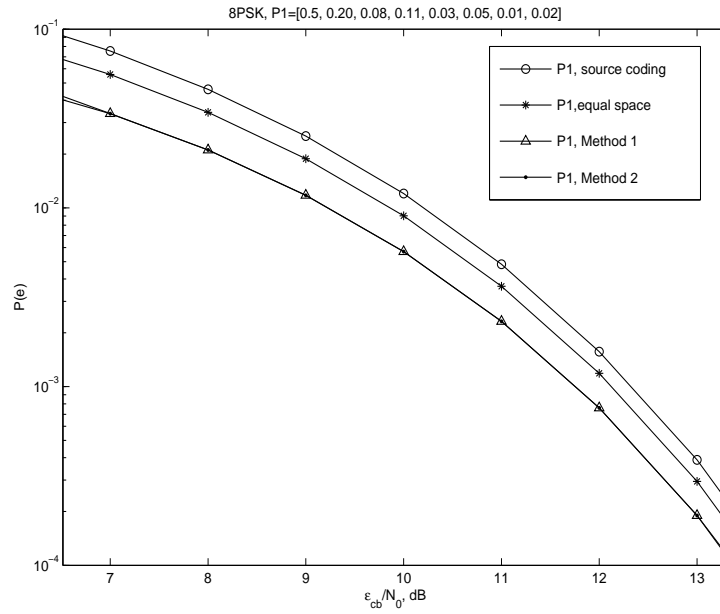


Figure 5.5:  $P(e)$  of Optimized and Non-optimized 8PSK Signaling Per Information Bit Energy

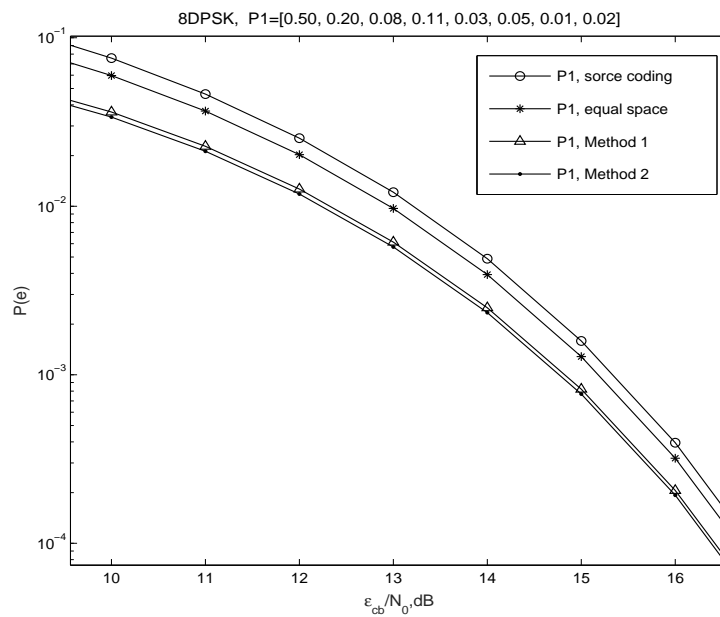


Figure 5.6:  $P(e)$  of Optimized and Non-optimized 8DPSK Signaling Per Information Bit Energy

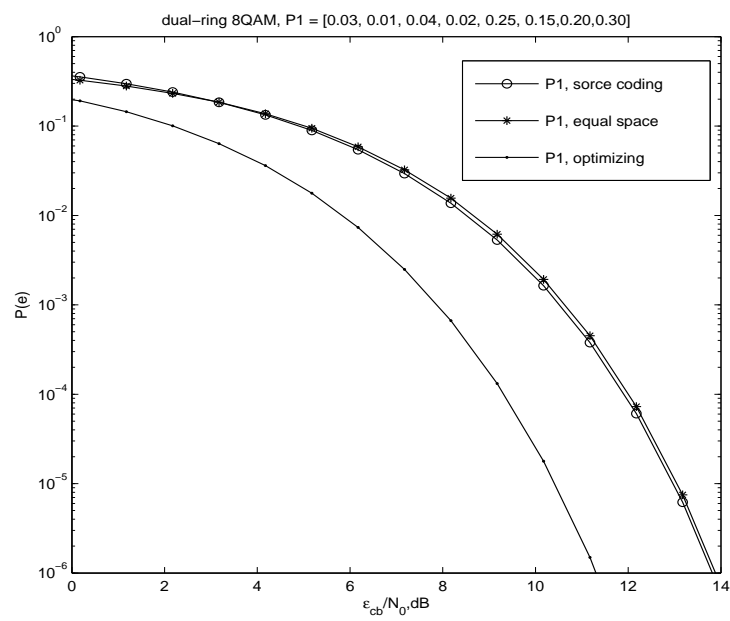


Figure 5.7:  $P(e)$  of Optimized and Non-optimized Dual-ring 8QAM Signaling Per Information Bit Energy

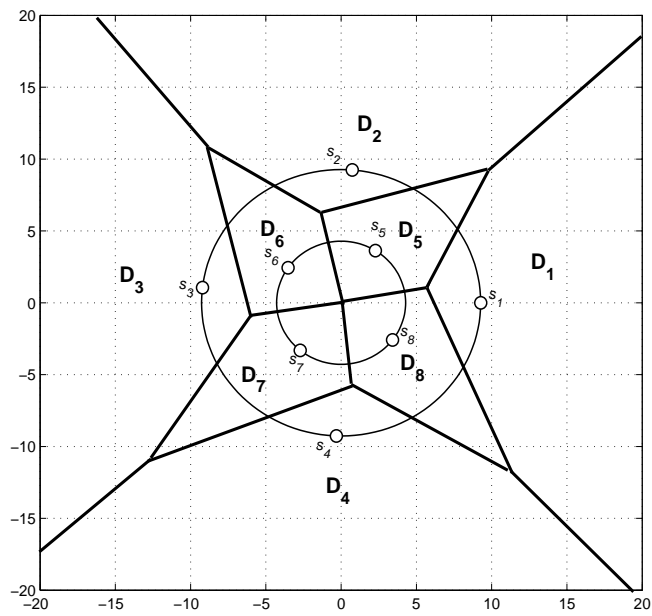


Figure 5.8: Signal Constellations and Decision Regions for Optimal 8QAM When  $P_m = \{0.03, 0.01, 0.04, 0.02, 0.25, 0.15, 0.20, 0.30\}$ ,  $\varepsilon_b/N_0 = 10.17\text{dB}$

## CHAPTER 6: CONCLUSION AND FUTURE WORK

### Conclusion

General Purpose Representation and Association Machine (GRPAM), as a novel information transmission/ reception mechanism, is proposed to employ versatile approach with hierarchical representation and association structures to do a quick and rough assessment on multitasks. It significantly improves our understanding in intelligent information representation and processing in biological brains.

In this dissertation, we have overviewed the concept of GPRAM and constructed the prototype of GPRAM system using (7, 4) Hamming and simple LDPC codes. Furthermore a switch function has been introduced, which is functioning similar to the sleep mode in bio-systems. For the longer codeword, several improvements on the learning progress have been discussed, i.e., how to improve learning for three-state systems; how to learn multitasks using perfect codewords, and finally how to perform progressive learning methods in learning multitasks.

Next, we have constructed Image Processing Unit (IPU) to recognize simple pattern and face images for GRPAM system. Inspired by bio-visual system, IPU is designed as a gateway for information processing in recognition. In this part, we put efforts in designing a more integrated system solving the problem of recognizing simple patterns and face images. Our study on recognizing low-resolution images with serious image quality degradation is closely experienced in the human visual system, similar from animals as well. We have also mimicked the learning and recognition processes inspired by bio-visual systems. Our results show that our GPRAM Image Processing Unit performs the similar behavior to human visual systems which could recognize heavily distorted low-resolution human faces and its performance in terms of recognition successful rate and false

alarm rate is almost as good as those achieved from state-of-the-art machine learning algorithms on images without severe distortion.

Finally, formulas have been derived to compute the optimizing signal constellation, decision regions, and symbol error rates for MPSK, MDPSK, dual-ring QAM with non-equal symbol probabilities. Several optimizing methods had been well developed and compared. The results show when comparing at the same information bit energy, all of our optimal systems have lower error probabilities than the conventional systems. Meanwhile, a novel methodology of pursuing approximately optimal dual-ring QAM system with non-equal symbol probabilities are also led in this dissertation. Since in bio-systems it is highly-unlikely to follow the ideal setting and uniform construction of single type of system, our methods in optimizing systems with non-equal symbol probabilities could be further employed in the bio-neural information processing systems.

### Future Work

In this dissertation, we have shown the success in establishing the prototype of GPRAM using communication theory. Our study on recognizing low-resolution images with serious image quality degradation is closely experienced in the human visual system, similar from animals as well. Nevertheless, this is not the end of the research. There are many interesting works need to be investigated in the future. We summarize the future work as follows.

In IPU, we have presented the result focusing on the random regular LDPC codes. Although the current results show the system is well-performed, there is another way that could improve the recognition performance, which is improving the current structure of LDPC codes. For instance, we could alter the regular LDPC codes to irregular, or modify the XOR gates to AND or OR gates with non-equiplorable symbols in LDPC codes.



Although we have achieved impressive recognition accuracy compared to state-of-the-art approaches, many improvements inspired by bio-systems could be implemented. For example, it has been well-known that stage by stage progressive learning can dramatically improve learning efficiency in bio-systems [106] [107]. In the future study, we could also consider implementing similar features, that is we improve performance stage by stage, closely tied to bio-evolution.

On the other side, we could build hierarchical layers of IPU to make more adaptable and reliable decisions though upon various noise. By adding up the multiple layers, we could also increase the capability of IPU, LDPC codes and iterative decoders. Again, the ultimate goal is to build a robot-like system which can display behaviors similar to biological brain. Once we reach this stage, then the system can be used as a test dummy to develop, evaluate, and improve our research tools for study.

## LIST OF REFERENCES

- [1] A.M. Turing, "On Computable Numbers, with an Application to the Entscheidungsproblem," *Proceedings of the London Mathematical Society*, ser. 2, vol. 42, pp. 230-265. 1937.
- [2] A.M. Turing, "On Computable Numbers, with an Application to the Entscheidungsproblem: A correction," *Proceedings of the London Mathematical Society*, ser. 2, vol. 43, pp. 544-546, 1937.
- [3] J. Von Neumann, *The computer and the brain*, Yale University Press, USA, 2nd edition, 2000
- [4] C. E. Shannon, "A mathematical theory of communication," *Bell System Technical Journal*, vol. 27, pp. 379-423, 1948
- [5] L. Wei, "General Purpose Representation and Association Machine, Part 1: Introduction and Illustrations," *IEEE Southeastcon*, pp. 1-5, Mar. 2012.
- [6] L. Wei, "General Purpose Representation and Association Machine, Part 2: Biological Implications," *IEEE Southeastcon*, pp. 1-5, Mar. 2012.
- [7] B. A. Olshausen, "20 years of learning about vision: Questions answered, questions unanswered, and questions not yet asked," *Twenty Years of Computational Neuroscience*, pp. 243-270, New York: Springer.
- [8] P. Sinha, B. Balas, Y. Ostrovsky, and R. Russell, "Face recognition by humans: Nineteen results all computer vision researchers should know about," *Proceedings of the IEEE*, vol. 94, no. 11, pp.1948 - 1962, 2006.
- [9] L. D. Harmon and B. Julesz, "Masking in visual recognition: effects of two-dimensional filtered noise," *Science*, vol. 180, no. 4091, pp. 1194-1197, 1973.

- [10] L. D. Harmon, "The recognition of faces," *Scientific American*, vol. 229, no. 5, pp. 70-83, 1973.
- [11] A. Yip and P. Sinha, "Role of color in face recognition," *Perception*, vol. 31, pp. 995-1003, 2002.
- [12] J. Sadr, I. Jarudi, and P. Sinha, "The role of eyebrows in face recognition," *Perception*, vol. 32, pp. 285-293, 2003.
- [13] G. Davies, H. Ellis, and J. Shepherd, "Cue saliency in faces as assessed by the 'Photofit' technique," *Perception*, vol. 6, pp. 263-269, 1977.
- [14] I. H. Fraser, G. L. Craig, and D. M. Parker, "Reaction time measures of feature saliency in schematic faces," *Perception*, vol. 19, no. 5, pp. 661-673, 1990.
- [15] A. W. Young, D. Hellawell, and D. C. Hay, "Configurational information in face perception," *Perception*, vol. 16, pp. 747-759, 1987.
- [16] Y. Jia, R. Argueta-Morales, M. Liu, Y. Bai, E. Divo, A. J. Kassab and W. M. Decampoli, "Experimental study of anisotropic stress/strain relationships of the piglet great vessels and relevance to pediatric congenital heart disease," *The Annals of Thoracic Surgery*, vol. 99, no. 4, pp. 1399-1407, 2015.
- [17] B. D. Smedt, M. P. Noel, C. Gilmore and D. Ansari. *Trends in Neuroscience and Education*, vol. 2, pp. 48-55, 2013.
- [18] S. Nishimoto, A. Vu, T. Naselaris, Y. Benjamini, B. Yu, and J. Gallant, "Reconstructing visual experiences from brain activity evoked by natural movies," *Current Biology*, vol. 21, no. 19, pp. 1641-1646, 2011.
- [19] G. Felsen, and Y. Dan. "A natural approach to studying vision," *Nature Neuroscience*, vol.8, pp. 1643-1646, 2015.

- [20] L. Wei, "Robustness of LDPC Codes and Internal Noisy Systems," in *41th Annu. Allerton Conf. Commun, Control Comput.*, Illinois, USA, Oct. 2003, pp. 1665-1674.
- [21] L. Wei, "Biologically Inspired Statistical Matched Filter," in *IEEE ICC*, vol. 2, pp. 810-814, May 2005.
- [22] L. Wei, "Biologically Inspired Amorphous Communications," in *IEEE ISIT*, pp. 1005, Sept 2005.
- [23] N. Wei and Y. Wan, "Optimal constellation for general rectangular PAM/QAM with arbitrary code mapping," in *Proc. ICC 2007*, pp. 2749-2754, Jun. 2007.
- [24] B. Dai, H. Li and L. Wei, "Image Processing Unit for General-Purpose Representation and Association System for Recognizing Low-Resolution Digits With Visual Information Variability," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 2016.
- [25] J. Hawkins and S. Blakeslee, *On Intelligence*. Times Books, 2004.
- [26] H. B. Barlow, *Intelligence*, guesswork, language. *Nature*, vol. 304, pp. 207-209, 1983.
- [27] M. D. Fox and M. E. Raichle, "Spontaneous fluctuations in brain activity observed with functional magnetic resonance imaging," *Nature Reviews Neuroscience*, vol. 9, pp. 700-711, 2007.
- [28] G. Roth and U. Dicke, "Evolution of the brain and intelligence. Trends in Cognitive Sciences," vol. 9, no. 5, pp. 250-257, 2005.
- [29] Y. Li, Y. Liu, J. L. W. Qin, K. Li, C. Yu and T. Jiang, "Brain anatomical network and intelligence," *PLoS Computational Biology*, vol. 5, no. 5, pp. 1000395, 2009.
- [30] W. K. Estes, "Is human memory obsolete? Comparisons of computer memory with the picture of human memory emerging from psychological research suggest basic differences in

modes of operation, with little likelihood that one can replace the other,” *American Scientist*, vol. 68, no. 1, pp. 62-69, 1980

- [31] R. G. Gallager, *Information Theory and Reliable Communication*, New York: Wiley, 1968.
- [32] R. M. Tanner, ”A recursive approach to low complexity codes,” *IEEE Trans. Inform. Theory*, vol. 27, pp.533-547, Sept. 1981.
- [33] C. Berrou, A. Glavieux, and P. Thitimajshima, ”Near Shannon limit error-correcting coding and decoding: Turbo codes,” *IEEE Int. Conf. Commun. (ICC)*, pp. 1064-1070, 1993.
- [34] R. J. McEliece, D. J. C. MacKay, Jung-Fu Cheng, ”Turbo decoding as an instance of Pearl’s belief propagation algorithm,” *IEEE JASC* vol. 16, no. 2, pp. 140 -152, 1998.
- [35] Pearl, J., *Probabilistic Reasoning in Intelligent Systems, 2nd Ed*, San Francisco, CA, USA, Kaufmann, 1988.
- [36] G. D. Forney, Jr., ”Codes on graphs: normal realization,” *IEEE Trans. Inform. Theory*, vol. 47, pp. 520-548, Feb. 2001.
- [37] F. R. Kschischang, B. J. Frey and H. A. Loeliger, ”Factor graphs and the sum-product algorithm,” *IEEE Trans. Inform. Theory*, vol. 47, pp. 498-519, Feb. 2001.
- [38] L. Wei and H. Qi, ”Near Optimal Limited Search Decoding on ISI/CDMA channels and decoding of long convolutional codes,” *IEEE Trans. on Inform. Theory*, vol. 46, No. 4, pp.1459 -1482, July 2000.
- [39] L. Wei, ”High Performance Iterative Viterbi Algorithm for Conventional Serial Concatenated Systems,” *IEEE Trans. on Information Theory*, Vol. 48, pp. 1759-1771, July 2002.

- [40] Q. Wang, L. Wei and R. A. Kennedy "Iterative Viterbi Decoding, Trellis Shaping and Multi-level Structure for High-Rate Concatenated TCM," *IEEE Trans on Comm.*, vol. 50, pp.48-55, Jan. 2002.
- [41] L. Wei, "Iterative Viterbi Algorithm: Implementation Issues," *IEEE Trans on Wireless Comm.*, vol. 3, pp. 382 - 386, Mar. 2004,
- [42] L. Wei, "Several properties of short LDPC codes," in *IEEE Trans. Commun.*, vol.52, pp. 721-727, May 2004.
- [43] Q. Wang and L. Wei, "Graph-Based Iterative Decoding Algorithms for Parity-Concatenated Trellis Codes," *IEEE Trans. on Information Theory*, Vol. 47, pp.1062-1074, Mar. 2001.
- [44] L. Wei, "Connectivity Reliability of Large Scale Random Ad Hoc Networks," *Procs of MIL-COM 2003*, Boston, USA, October 13-16.
- [45] I. Chaaban and M. R. Scheessele, "Human performance on the USPS database," *Technical Report*, Indiana University, 2007.
- [46] L. Sirovich and M. Kirby, "Low-Dimensional procedure for the characterisation of human faces," *J. Optical Soc. Of Am.*, vol. 4, pp. 519-524, 1987.
- [47] M. Kirby and L. Sirovich, "Application of the Karhunen- Love procedure for the characterisation of human faces," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 12, pp. 831-835, Dec. 1990.
- [48] M. Turk and A. Pentland, "Eigenfaces for recognition," *J. Cognitive Neuroscience*, vol. 3, pp. 71-86, 1991.
- [49] A. Pentland, B. Moghaddam, and T. Starner, "View-Based and modular eigenspaces for face recognition," *Proc. IEEE CS Conf. Computer Vision and Pattern Recognition*, pp. 84-91, 1994.

- [50] L. Zhao and Y.H. Yang, "Theoretical analysis of illumination in pcbased vision systems," *Pattern Recognition*, vol. 32, pp. 547-564, 1999.
- [51] L. Hong and A. Jain, "Integrating faces and fingerprints for personal identification," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 20, no. 12, pp. 1295-1307, Dec. 1998.
- [52] P. Verlinde, G. Matre, and E. Mayoraz, "Decision fusion using a multilinear classifier," *Proc. Intl Conf. Multisource-Multisensor Information Fusion*, vol. 1, pp. 47-53, Jul. 1998.
- [53] K.K. Sung and T. Poggio, "Learning human face detection in cluttered scenes," *Computer Analysis of Image and patterns*, pp. 432-439, 1995.
- [54] S. Lawrence, C.L. Giles, A.C. Tsoi, and A.D. Back, "Face recognition: A convolutional neural-network approach," *IEEE Trans. Neural Networks*, vol. 8, pp. 98-113, 1997.
- [55] J. Weng, J.S. Huang, and N. Ahuja, "Learning recognition and segmentation of 3D objects from 2D images," *Proc. IEEE Int'l Conf. Computer Vision*, pp. 121-128, 1993.
- [56] F. Samaria and F. Fallside, "Face identification and feature extraction using hidden markov models," *Image Processing: Theory and Application*, G. Vernazza, ed., Elsevier, 1993.
- [57] F. Samaria and A.C. Harter, "Parameterisation of a stochastic model for human face identification," *Proc. Second IEEE Workshop Applications of Computer Vision*, 1994.
- [58] S. Tamura, H. Kawa, and H. Mitsumoto, "Male Female identification from 8\_6 very low resolution face images by neural network," *Pattern Recognition*, vol. 29, pp. 331-335, 1996.
- [59] T. Kanade, "Picture processing by computer complex and recognition of human faces," *technical report*, Dept. Information Science, Kyoto Univ., 1973.
- [60] R. Bruneli and T. Poggio, "Face recognition: features versus templates," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 15, pp. 1042-1052, 1993.

- [61] R.J. Baron, "Mechanism of human facial recognition," *Int'l J. Man Machine Studies*, vol. 15, pp. 137-178, 1981.
- [62] Z. Wang, Z. Miao, Q. M. Wu, Y. Wan, Z. Tang, "Low-resolution Face Recognition: a review," *The Visual Computer -Springer Journals*, Vol 30, no. 4, pp. 359-386, 2014.
- [63] A. S. Tolba, A. H. El-baz and A. A. El-Harby, "Face Recognition: A Literature Review," *International Journal of Signal Processing*, Vol.2, no. 2, pp. 88-103, 2006
- [64] W. Zhao, R. Chellappa, and P. Phillips, "Subspace linear discriminant analysis for face recognition," Center for Automation Research, University of Maryland, College Park, Tech. Rep. CAR-TR-914, 1999.
- [65] A. Lemieux, M. Parizeau, "Experiments on eigenfaces robustness," *Proc. IAPR/IEEE 16th Int. Conf. on Pattern Recognition (ICPR)*, Quebec, Canada, Aug. 2002, vol. 1, pp. 421-424, 2002
- [66] B.J. Boom, G.M. Beumer, L.J. Spreeuwers, R.N.J. Veldhuis, "The effect of image resolution on the performance of a face recognition system," *Proc. IEEE 9th Int. Conf. on Control, Automation, Robotics and Vision (ICARCV)*, pp. 1-6, 2006
- [67] P. Hennings-Yeomans, B.V.K. Vijaya Kumar, S. Baker, "Robust Low-resolution Face Identification and Verification using High-resolution Features," *Image Processing (ICIP), 2009 16th IEEE International Conference on*, pp 33-36, 2009
- [68] O. Arandjelovic and R. Cipolla, "A manifold approach to face recognition from low quality video across illumination and pose using implicit super-resolution," *ICCV*, October 2007.
- [69] Kui Jia and Shaogang Gong, "Multi-modal tensor face for simultaneous super-resolution and recognition," *ICCV*, pp. 1683-1690, 2005.



- [70] B.K. Gunturk, A.U. Batur, Y. Altunbasak, M.H. Hayes, and R.M. Mersereau, "Eigenface-domain superresolution for face recognition," *IEEE Trans. on Im. Proc.*, vol. 12, no. 5, pp. 597-606, May 2003.
- [71] Xiaogang Wang and Xiaoou Tang, "Hallucinating face by eigentransformation," *IEEE Trans. on Syst., Man and Cyb., Part C*, vol. 35, no. 3, pp. 425-434, 2005.
- [72] R. G. Gallager, "Low-density parity-check codes," *IRE Trans. Inform. Theory*, IT-8:21-28, January 1962.
- [73] R. G. Gallager, *Low-Density Parity-Check Codes*, MIT Press, Cambridge, MA, 1963.
- [74] D. MacKay and R. Neal, "Good codes based on very sparse matrices," *Cryphotography and Coding, 5th IMA Conf.*, C. Boyd, Ed., *Lecture Notes in Computer Science*, pp. 100-111, Berlin, Germany: Springer, 1995.
- [75] D. MacKay, "Good error correcting codes based on very sparse matrices," *IEEE Trans. Information Theory*, pp.399-431, Mar. 1999.
- [76] L. Shu and D. Costello, *Error Control Coding*, Prentice Hall, 2nd edition, 2004.
- [77] S.-Y. Chung, "On the construction of some capacity-approaching coding schemes," Ph.D. dissertation, Massachusetts Institute of Technology, Cambridge, 2000.
- [78] H. Dehghani, M. Ahmadi, S. Alikhani and R. Hasni, "Calculation of Girth of Tanner Graph in LDPC Codes," *Trends in Applied Sciences Research*, No. 7, pp 929-934, 2012
- [79] W. E. Ryan, "An introduction to LDPC codes," Dept. of electrical and communication Engg, The university of Arizona, [online] Available: [www.telecom.tuc.gr/~alex/papers/ryan.pdf](http://www.telecom.tuc.gr/~alex/papers/ryan.pdf)
- [80] J. B. Proakis and M. Salehi, *Digital Communications*, 5th edition. McGraw-Hill, 2008

- [81] B.Sklar, P.K.Ray, *Digital Communications*, Fundamentals and Applications, second edition, Pearson Education, Inc., 2001.
- [82] T.Schilling, *Principles of Communications Systems*, second edition, Tata McGraw-Hill publishing Company Limited, New Delhi
- [83] M.Tao, "Principles of Communications, Chapter-8: Digital Modulation Techniques," *Shanghai Jiao Tong University*.
- [84] J. M. Wozencraft and I. M. Jacobs, *Principles of Communication Engineering*, John Wiley & Sons, Inc., 1965.
- [85] A. de Lucena, J. Mota, and C. Cavalcante<sup>2</sup>, "Optimum detection of non-orthogonal QAM signals with spectral overlapping," *IET Communications*, vol 3, no. 2, pp. 249 - 256, 2009
- [86] N. Ermolova, "Average symbol error probability of arbitrary rectangular quadrature amplitude modulation in generalised shadowed fading channels," *IET Communications*, vol. 2, no. 7, pp. 903 - 908, 2008
- [87] Y. Zhong, F. Alajaji and L. L. Campbell, "On the joint source-channel coding error exponent for discrete memoryless Systems," *Information Theory, IEEE transactions on*, vol. 52, no. 4, pp. 1450-1468, Apr. 2006.
- [88] F. Alajaji, N. Phamdo, and T. Fuja, "Channel codes that exploit the residual redundancy in CELP-encoded speech," *IEEE Trans. Speech Audio Processing*, vol. 4, no. 5, pp. 325-336, Sept. 1996.
- [89] W. Xu, J. Hagenauer, and J. Hollmann, "Joint source-channel decoding using the residual redundancy in compressed images," in *Proc. Int. Conf. Communications*, Dallas, TX, Jun. 1996.

- [90] H. Behroozi, F. Alajaji and T. Linder, "On the Optimal Performance in Asymmetric Gaussian Wireless Sensor Networks with Fading," *IEEE Transactions on Signal Processing*, Vol. 58, No. 4, pp. 2436-2441, Apr. 2010.
- [91] H. Kuai, F. Alajaji, and G. Takahara, "Tight error bounds for nonuniform signaling over AWGN channels," *Information Theory, IEEE transaction on*, vol. 46, no. 7, pp. 2712-2718, Nov. 2000.
- [92] Victor V. Zhirnov, Ralph K. Cavin III, *Microsystems for Bioelectronics: Scaling and Performance Limits*, 2nd edition, William Andrew, 2015.
- [93] I. Korn and J. Fonseka, "Optimized receiver for binary signals and nonequal probabilities," *International Conference on Communications, ICT-2000*, Acapulco, Mexico, pp.597-601, May 2000.
- [94] I. Korn, J. P. Fonseka, and S. Xing, "Optimal binary communications with nonequal probabilities," *IEEE Trans. Commun.*, vol. 51, no. 9, pp. 1435-1438, Sep. 2003.
- [95] H. Nguyen and T. Nechiporenko, "Quarternary signal sets for digital communications with nonuniform sources," in *Proc. IEEE Can. Conf. Elect. Comput. Eng.*, pp. 2085-2088, May 2005.
- [96] B. Moore, G. Takahara and F. Alajaji, "Pairwise optimization of modulation constellations for non-uniform sources," *IEEE Can. J. Elect. Comput. Eng.*, vol. 34, no. 4, pp. 167-177, 2009.
- [97] L. Wei and I. Korn, "Optimal M-amplitude shift keying/quadrature amplitude shift keying with non-equal symbol probabilities," *IET Communications*, vol. 5, no. 6, pp. 745-752, Apr. 2011.
- [98] L. Wei, "Optimal ASK and QAM with non-equal symbol probabilities on rayleigh fading channels," *Electronic Letters*, vol. 47, no. 1, pp. 63-65, Jan. 2011.

- [99] L. Wei, "Optimized M-ary orthogonal and bi-orthogonal signaling using coherent receiver with non-equal symbol probabilities," *Communication Letters, IEEE*, vol. 16, no. 6, pp. 793-796, Jun. 2012.
- [100] L. Szczeciński, S. Aissa, C. Gonzalez and M. Bacic, "Exact evaluation of bit- and symbol-error rates for arbitrary 2-D modulation and nonuniform signaling in AWGN channel," *communications, IEEE Transaction on*, vol. 54, no. 6, 2006.
- [101] H. Saeedi and A. H. Banihashemi, "Performance of belief propagation for decoding LDPC codes in the presence of channel estimation error", *IEEE Trans. Commun.*, vol. 55, no. 1, pp. 83-89, Jan. 2007.
- [102] L. Wei, D. M. Levi, R. Li, S. Klein "Feasibility Study on a Hyper-acuity Device with Motion Uncertainty: Two-point Stimuli," *IEEE Trans. Syst. Man, Cybern. B.*, vol. 37, no. 2, pp. 385-397, Apr. 2007.
- [103] ORL face database, AT&T Laboratories, Cambridge, U. K. [Online] Available: <http://www.uk.research.att.com/facedatabase.html>.
- [104] A. Ramamoorthy and R. Wesel, "Construction of short block length irregular low-density parity-check codes," in *IEEE Int. Conf. Communications*, vol. 1, pp. 410-414, Jun. 2004.
- [105] D. Cai , X. He , Y. Hu , J. Han and T. S. Huang, "Learning a spatially smooth subspace for face recognition", *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, pp. 138-142, 2007.
- [106] H. Li, B. Dai, S. Schultz, and L. Wei, "General Purpose Representation and Association Machine, Part 3: Prototype Study using LDPC codes," in *Proc. IEEE Southeastcon*, pp. 1-5, Mar. 2013.

- [107] H. Li and L. Wei, "General Purpose Representation and Association Machine, Part 4: Improve learning for three-states and multi-tasks," in *Proc. IEEE Southeastcon*, pp.1-5, Mar. 2013.