Robust Image Hash Functions using Higher Order Spectra

Brenden Chong Chen
BEng (Hons, 1st Class)

A Thesis Submitted in Fulfilment of the Requirements for the Degree of

Doctor of Philosophy

at the

Queensland University of Technology
Speech, Audio, Image and Video Technology Research Laboratory
Science and Engineering Faculty

2012
Keywords

robust hashing, image hashing, higher order spectra, bispectrum, adaptive deterministic quantization, random projection, biometric template security, non-negative matrix factorization, Fourier-Mellin, Gray code, distance distortion.
Abstract

Robust hashing is an emerging field that can be used to hash certain data types in applications unsuitable for traditional cryptographic hashing methods. Traditional hashing functions have been used extensively for data/message integrity, data/message authentication, efficient file identification and password verification. These applications are possible because the hashing process is compressive, allowing for efficient comparisons in the hash domain but non-invertible meaning hashes can be used without revealing the original data. These techniques were developed with deterministic (non-changing) inputs such as files and passwords. For such data types a 1-bit or one character change can be significant, as a result the hashing process is sensitive to any change in the input. Unfortunately, there are certain applications where input data are not perfectly deterministic and minor changes cannot be avoided. Digital images and biometric features are two types of data where such changes exist but do not alter the meaning or appearance of the input. For such data types cryptographic hash functions cannot be usefully applied.

In light of this, robust hashing has been developed as an alternative to cryptographic hashing and is designed to be robust to minor changes in the input. Although similar in name, robust hashing is fundamentally different from cryptographic hashing. Current robust hashing techniques are not based on cryptographic methods, but instead on pattern recognition techniques. Modern robust hashing algorithms consist of feature extraction followed by a randomization stage that introduces non-invertibility and compression, followed by quantization and binary encoding to produce a binary hash output. In order to preserve robustness of the extracted features, most randomization methods are linear and this is detrimental to the security aspects required of hash functions. Furthermore, the quantization and encoding stages used to binarize real-valued features requires the learning of appropriate quantization thresholds. How these thresholds are
learnt has an important effect on hashing accuracy and the mere presence of such
thresholds are a source of information leakage that can reduce hashing security.

This dissertation outlines a systematic investigation of the quantization and
encoding stages of robust hash functions. While existing literature has focused
on the importance of quantization scheme, this research is the first to emphasise
the importance of the quantizer training on both hashing accuracy and hashing
security. The quantizer training process is presented in a statistical framework
which allows a theoretical analysis of the effects of quantizer training on hashing
performance. This is experimentally verified using a number of baseline robust
image hashing algorithms over a large database of real world images.

This dissertation also proposes a new randomization method for robust image
hashing based on Higher Order Spectra (HOS) and Radon projections. The
method is non-linear and this is an essential requirement for non-invertibility.
The method is also designed to produce features more suited for quantization
and encoding. The system can operate without the need for quantizer training, is
more easily encoded and displays improved hashing performance when compared
to existing robust image hashing algorithms. The dissertation also shows how
the HOS method can be adapted to work with biometric features obtained from
2D and 3D face images.
# Table of Contents

Abstract iii  
Table of Contents v  
List of Figures ix  
List of Tables xv  
List of Abbreviations xvii  
List of Publications xix  
Statement of Original Authorship xxi  
Acknowledgements xxiii  

## Chapter 1 Introduction 1  
1.1 Motivation and Overview 1  
1.2 Research Scope 4  
1.3 Aims and Objectives 4  
1.4 Outline of Thesis 6  
1.5 Original Contributions of the Research 7  

## Chapter 2 Review of Robust Hashing 9  
2.1 Introduction 9  
2.2 Cryptographic Hash Functions 11  
2.3 Overview of Robust Hashing 16  
2.3.1 Feature Extraction 20  
2.3.2 Randomization 24  
2.3.3 Quantization 27
## TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>Chapter</th>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.3.4</td>
<td>Encoding</td>
<td>30</td>
</tr>
<tr>
<td>2.4</td>
<td>Evaluating Robust Hash Functions</td>
<td>31</td>
</tr>
<tr>
<td>2.4.1</td>
<td>Accuracy Properties</td>
<td>31</td>
</tr>
<tr>
<td>2.4.2</td>
<td>Distance Distortion</td>
<td>33</td>
</tr>
<tr>
<td>2.4.3</td>
<td>Security Properties</td>
<td>35</td>
</tr>
<tr>
<td>2.5</td>
<td>Image Databases</td>
<td>37</td>
</tr>
<tr>
<td>2.5.1</td>
<td>The UCID database</td>
<td>37</td>
</tr>
<tr>
<td>2.5.2</td>
<td>The MIRFLICKR database</td>
<td>37</td>
</tr>
<tr>
<td>2.5.3</td>
<td>The XM2VTS database</td>
<td>38</td>
</tr>
<tr>
<td>2.5.4</td>
<td>Image Normalisation</td>
<td>39</td>
</tr>
<tr>
<td>2.6</td>
<td>Baseline Systems</td>
<td>40</td>
</tr>
<tr>
<td>2.6.1</td>
<td>DWT Based Features</td>
<td>41</td>
</tr>
<tr>
<td>2.6.2</td>
<td>Fourier Mellin Based Features</td>
<td>41</td>
</tr>
<tr>
<td>2.6.3</td>
<td>NMF Based Features</td>
<td>42</td>
</tr>
<tr>
<td>2.7</td>
<td>Shortcomings of Existing Robust Hash Functions</td>
<td>43</td>
</tr>
</tbody>
</table>

### Chapter 3  Effects of Quantizer Training on Hashing Performance 45

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.1 Introduction</td>
<td>45</td>
</tr>
<tr>
<td>3.2 Theoretical analysis of the effects of the secret-key (K) on Randomized feature Distributions</td>
<td>47</td>
</tr>
<tr>
<td>3.3 Effects of Secret-keys on Feature Component pdfs</td>
<td>52</td>
</tr>
<tr>
<td>3.4 Effects of Image Data on Feature Component pdfs</td>
<td>53</td>
</tr>
<tr>
<td>3.5 Performance Evaluation under Different Quantizer Training Scenarios</td>
<td>61</td>
</tr>
<tr>
<td>3.5.1 Fragility of Hashes</td>
<td>62</td>
</tr>
<tr>
<td>3.5.2 Hash Entropy</td>
<td>67</td>
</tr>
<tr>
<td>3.5.3 Hashing Accuracy</td>
<td>68</td>
</tr>
<tr>
<td>3.6 Discussion on Security</td>
<td>76</td>
</tr>
<tr>
<td>3.7 Chapter Summary</td>
<td>77</td>
</tr>
</tbody>
</table>

### Chapter 4  Robust Hashing using Angle Features 79

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.1 Introduction</td>
<td>79</td>
</tr>
<tr>
<td>4.2 Proposed Method</td>
<td>80</td>
</tr>
<tr>
<td>4.2.1 Comparison to Biophasors</td>
<td>81</td>
</tr>
<tr>
<td>4.3 Encoding Benefits</td>
<td>83</td>
</tr>
<tr>
<td>4.3.1 Experimental Results</td>
<td>92</td>
</tr>
</tbody>
</table>
## Table of Contents

4.4  Quantization Benefits .................................................. 98  
4.5  Chapter Summary .......................................................... 100  

**Chapter 5  Robust Hashing Using Bispectra**  
5.1  Introduction ............................................................... 101  
5.2  Higher Order Spectra ..................................................... 102  
5.3  The Radon Transform .................................................... 106  
5.4  Normalisation of Radon Projections .................................. 107  
5.5  Properties of Radon Projections and their Bispectral Invariants .................................................. 109  
  5.5.1  Robustness Analysis .................................................. 110  
  5.5.2  Sensitivity Analysis .................................................. 122  
5.6  Proposed Scheme .......................................................... 130  
  5.6.1  Security Analysis ..................................................... 133  
5.7  Experimental Results .................................................... 135  
5.8  Computational Complexity ............................................. 145  
5.9  Chapter Summary .......................................................... 147  

**Chapter 6  Conclusions and Further Work**  
6.1  Conclusions ............................................................... 149  
6.2  Summary of Contributions ............................................. 150  
6.3  Future Work .............................................................. 151
List of Figures

2.1 An original uncompressed image (left) and its JPEG compressed counterpart using 50% quality (right). The images are visually identical but have different pixel values. ................................ 10

2.2 Results returned by Google Images using the search query 'House' which can refer to a structural dwelling or to the popular US TV show .......................................................... 10

2.3 The Merkle-Damgard Construction, where the Final hash value is created by chaining together the outputs of a compression function applied to each block of the input. ......................... 14

2.4 The four steps of the robust hashing process, using a 2-bit standard binary quantizer where $N = 7$ and $M = 4$. The feature vector $F$ and quantization thresholds used are arbitrarily selected for illustrative purposes. ........................................... 21

2.5 Swaminathan’s technique obtains features by summing along circles of different radii, $\rho$ (rho) in the frequency domain. Three such circles are shown above with $\rho$ of 0.02, 0.2 and 0.4. Note that in the above image the frequency domain is log-transformed then normalized for display. .................................................. 22

2.6 The Lena image and its reconstructions using NMF decomposition. ................. 23

2.7 Uniform quantization applied to a Gaussian distributed feature. The uniformly spaced thresholds (in red) divides the data into uneven levels. ................................................. 28

2.8 Example of a ROC curve, the black dashed line marks in the line of Equal Error Rate. .......................................................... 33
# LIST OF FIGURES

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.9</td>
<td>Example of 2-bit adaptive deterministic quantization on a Gaussian distributed real-valued feature (left). The quantization training determines threshold locations (in red) that divide the features into discrete levels of equal probability. The distribution of quantized features is always uniform (right).</td>
<td>36</td>
</tr>
<tr>
<td>2.10</td>
<td>Example images from the UCID database.</td>
<td>38</td>
</tr>
<tr>
<td>2.11</td>
<td>Example images from the MIRFLICKR database.</td>
<td>39</td>
</tr>
<tr>
<td>2.12</td>
<td>Example images from the XM2VTS database.</td>
<td>39</td>
</tr>
<tr>
<td>2.13</td>
<td>Histogram equalisation can have undesirable results on the input image.</td>
<td>40</td>
</tr>
<tr>
<td>2.14</td>
<td>Example of the image normalisation procedure used in this dissertation.</td>
<td>40</td>
</tr>
<tr>
<td>2.15</td>
<td>Block diagram of Venkatesan’s image hashing algorithm.</td>
<td>41</td>
</tr>
<tr>
<td>2.16</td>
<td>Block diagram of Swaminathan’s image hashing algorithm.</td>
<td>42</td>
</tr>
<tr>
<td>2.17</td>
<td>Block diagram of Monga’s image hashing algorithm.</td>
<td>43</td>
</tr>
<tr>
<td>3.1</td>
<td>Probability density functions of the 1st, 24th, 48th and 64th randomized features generated using DWT method on the MIRFLICKR image set for different secret-keys.</td>
<td>54</td>
</tr>
<tr>
<td>3.2</td>
<td>Probability density functions of the 1st, 24th, 48th and 64th randomized features generated using Fourier Mellin method on the MIRFLICKR image set for different secret-keys.</td>
<td>55</td>
</tr>
<tr>
<td>3.3</td>
<td>Probability density functions of the 1st, 24th, 48th and 64th randomized features generated using NMF method on the MIRFLICKR image set for different secret-keys.</td>
<td>56</td>
</tr>
<tr>
<td>3.4</td>
<td>Probability density functions of the 1st, 24th, 48th and 64th randomized features generated using DWT method and three different image sets (MIRFLICKR, UCID and XM2VTS) using same secret-key.</td>
<td>57</td>
</tr>
<tr>
<td>3.5</td>
<td>Probability density functions of the 1st, 24th, 48th and 64th randomized features generated using Fourier Mellin method and three different image sets (MIRFLICKR, UCID and XM2VTS) using same secret-key.</td>
<td>58</td>
</tr>
</tbody>
</table>
3.6 Probability density functions of the 1st, 24th, 48th and 64th randomized features generated using the NMF method and three different image sets (MIRFLICKR, UCID and XM2VTS) using same secret-key. ......................................................... 59

3.7 Fragility curves and expected discriminability (ED) of DWT scheme for different quantizer training scenarios using 1, 2, 3 and 4 bit quantization. Scenario 3 uses the XM2VTS data set of only facial images and presents a significantly different fragility curve compared to the others. The DWT method is sensitive to input data but not to the key. ................................................................. 64

3.8 Fragility curves and expected discriminability (ED) of NMF hash for different quantizer training scenarios using 1, 2, 3 and 4 bit quantization. The NMF method is sensitive to the key. It is not sensitive to differences between UCID and MIRFLICKR images but when all images are faces as in XM2VTS there is a difference as in scenario 3. ................................................................. 65

3.9 Fragility curves and expected discriminability (ED) of Fourier Mellin scheme for different quantizer training scenarios using 1, 2, 3 and 4 bit quantization. The sensitivity here is similar to that of the NMF method ................................................................. 66

3.10 ROC curves for DWT scheme for different quantizer training scenarios using 1, 2, 3 and 4 bit quantization. ................................................................. 69

3.11 ROC curves for Fourier Mellin scheme for different quantizer training scenarios using 1, 2, 3 and 4 bit quantization. ................................................................. 70

3.12 ROC curves for NMF scheme for different quantizer training scenarios using 1, 2, 3 and 4 bit quantization. ................................................................. 71

3.13 A 2-bit adaptive deterministic quantizer is trained with a low variance pdf (red) and the thresholds (blue) are applied to quantize a high variance pdf (green). The learned quantization thresholds will produce a poor partitioning of the high variance pdf. Resulting in an under-quantized area with no thresholds ($x > -3$) and an over-quantized area containing all of the thresholds ($-5 \leq x \leq -3$). ................................................................. 73
3.14 A 2-bit adaptive deterministic quantizer is trained with a low variance pdf (red) and the thresholds (blue) are applied to quantize another low variance pdf (green). When the two pdfs do not overlap, the resulting quantization and hence EER performance will be poor regardless of $b$. However, if the two pdfs significantly overlap then good quantization is still achievable.

3.15 A 1-bit (a) and 4-bit (b) adaptive deterministic quantizer is trained with a high variance pdf (red) and the thresholds (blue) are applied to quantize a low variance pdf (green). A 1-bit quantizer may not partition the low variance pdf at all and EER performance will be poor. When $b$ is increased to 4 a reasonable quantization of the low variance pdf can be achieved.

4.1 The function $y = \arctan(x)$ asymptotes for large values of $x$.

4.2 Performance comparisons for no encoding, standard binary and GrayCode for Fourier-Mellin features and random projection.

4.3 Performance comparisons for no encoding, standard binary and GrayCode for NMF features and random projection.

4.4 Performance comparisons for no encoding, standard binary and GrayCode for Fourier-Mellin features and Complex random projection.

4.5 Performance comparisons for no encoding, standard binary and GrayCode for NMF features and Complex random projection.

5.1 Region of computation of the bispectrum. Feature are obtained by integrating (summing) the bispectrum along radial lines of varying gradients $a$.

5.2 Direct and indirect methods of computing invariant features from the bispectrum of a 1D input signal.

5.3 Example of the Radon projection obtained from an image $I(n, m)$ at angle $\theta$.

5.4 Radon projections ($\theta = 0$) of star shaped object for different scaling and translation.

5.5 Lena image and its rotated counterpart. The Radon projection of the rotated Lena image displays a triangular shape.
<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>5.6</td>
<td>The Radon projection of the Lena image with $\theta = 45$ is normalized using the Radon projection of the ones image with $\theta = 45$.</td>
</tr>
<tr>
<td>5.7</td>
<td>Behaviour of Radon projections and the integrated bispectrum $(V)$ to brightness adjustment</td>
</tr>
<tr>
<td>5.8</td>
<td>Behaviour of Radon projections and the integrated bispectrum $(V)$ to Contrast Stretching</td>
</tr>
<tr>
<td>5.9</td>
<td>Behaviour of Radon projections and the integrated bispectrum $(V)$ to histogram equalisation</td>
</tr>
<tr>
<td>5.10</td>
<td>Behaviour of Radon projections and the integrated bispectrum $(V)$ to image scaling where image down sampled by factor of 2 then scaled up to original size (using Bilinear interpolation)</td>
</tr>
<tr>
<td>5.11</td>
<td>Behaviour of Radon projections and the integrated bispectrum $(V)$ to mean filtering with 9x9 kernel</td>
</tr>
<tr>
<td>5.12</td>
<td>Behaviour of Radon projections and the integrated bispectrum $(V)$ to image rotation of 2 degrees</td>
</tr>
<tr>
<td>5.13</td>
<td>Behaviour of Radon projections and the integrated bispectrum $(V)$ to Gaussian noise</td>
</tr>
<tr>
<td>5.14</td>
<td>Behaviour of Radon projections and the integrated bispectrum $(V)$ to JPEG compression at 10% Quality</td>
</tr>
<tr>
<td>5.15</td>
<td>The $\theta = 0$ Radon Projection of the Lena (left) and Mandrill (right) images for three different permutations.</td>
</tr>
<tr>
<td>5.16</td>
<td>The positive half of the Fourier domain obtained from the DTFT of the $\theta = 0$ Radon Projection of the Lena (left) and Mandrill (right) images for three different permutations.</td>
</tr>
<tr>
<td>5.17</td>
<td>The real component of the Bispectral invariants derived from the $\theta = 0$ Radon Projection of the Lena (left) and Mandrill (right) images for three different permutations.</td>
</tr>
<tr>
<td>5.18</td>
<td>The imaginary component of the Bispectral invariants derived from the $\theta = 0$ Radon Projection of the Lena (left) and Mandrill (right) images for three different permutations.</td>
</tr>
<tr>
<td>5.19</td>
<td>Distribution of intra-class Euclidean distances for Radon projections $(d_E(\tilde{x}_k, \tilde{x}_l))$ given different image transformations. The fragility and unpredictability is also shown beside for reference.</td>
</tr>
<tr>
<td>Figure</td>
<td>Description</td>
</tr>
<tr>
<td>--------</td>
<td>-------------</td>
</tr>
<tr>
<td>5.20</td>
<td>Distribution of intra-class Euclidean distances for the real component of the integrated bispectrum ( d_E(\mathbf{V}^R_k, \mathbf{V}^R_l) ) given different image transformations. The fragility and unpredictability is also shown beside for reference.</td>
</tr>
<tr>
<td>5.21</td>
<td>Distribution of intra-class Euclidean distances for the imaginary component of the integrated bispectrum ( d_E(\mathbf{V}^I_k, \mathbf{V}^I_l) ) given different image transformations. The fragility and unpredictability is also shown beside for reference.</td>
</tr>
<tr>
<td>5.22</td>
<td>Reconstructions of the original Lena image using different types of image features.</td>
</tr>
<tr>
<td>5.23</td>
<td>Lena image after being transformed by JPEG compression, Gaussian noise, mean filtering, cropping, rotation and histogram equalisation.</td>
</tr>
<tr>
<td>5.24</td>
<td>ROC curves for different hashing methods using a 2-bit Quantizer and Quantizer Training Scenario 1.</td>
</tr>
<tr>
<td>5.25</td>
<td>ROC curves for different hashing methods using a 4-bit Quantizer and Quantizer Training Scenario 1.</td>
</tr>
<tr>
<td>5.26</td>
<td>ROC curves for different hashing methods using a 2-bit Quantizer and Quantizer Training Scenario 6.</td>
</tr>
<tr>
<td>5.27</td>
<td>ROC curves for different hashing methods using a 4-bit Quantizer and Quantizer Training Scenario 6.</td>
</tr>
<tr>
<td>5.28</td>
<td>Unpredictability of hashes for fixed image but varying secret-key measured using binary entropy.</td>
</tr>
<tr>
<td>5.29</td>
<td>Percentage reduction in processing time due to each additional processing core.</td>
</tr>
</tbody>
</table>
List of Tables

<table>
<thead>
<tr>
<th>Table</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1</td>
<td>Metrics used in this dissertation for measuring similarity between two vectors a and b of dimension N. For Hamming distance, the vectors are assumed to be binary valued in each dimension.</td>
</tr>
<tr>
<td>3.1</td>
<td>Entropy of binary hashes for different quantizer training scenarios measured in bits. Note that the maximum entropy for the 1, 2, 3 and 4-bit quantizers are 64, 128, 192 and 256 bits respectively.</td>
</tr>
<tr>
<td>4.1</td>
<td>Distance distortions for standard and cyclic integers encoded using standard binary and GrayCode for b = 1, \ldots, 8.</td>
</tr>
<tr>
<td>4.2</td>
<td>EER of hashing schemes for different quantizer training scenarios using Uniform Quantization.</td>
</tr>
<tr>
<td>4.3</td>
<td>Fragility of hashing schemes for different quantizer training scenarios using Uniform Quantization.</td>
</tr>
<tr>
<td>5.1</td>
<td>Robustness of the Radon projections x and their derived HOS invariants V for common image transformations. Both the mean and standard deviation is shown.</td>
</tr>
<tr>
<td>5.2</td>
<td>Fragility of the Radon projections x and their derived HOS invariants V. Both mean and standard deviation is shown.</td>
</tr>
<tr>
<td>5.3</td>
<td>Sensitivity of the ( \theta = 0 ) Radon projection x and their derived HOS invariants V of image 1 in the MIRFLICKR database to a changing secret-key. Both mean and standard deviation is shown.</td>
</tr>
<tr>
<td>5.4</td>
<td>Comparison of hashing Fragility for the different hashing techniques under different Quantizer Training and Quantization Schemes.</td>
</tr>
<tr>
<td>5.5</td>
<td>Comparison of equal error rates for Quantizer Training Scenario 1 using different hashing techniques and different image transformations.</td>
</tr>
</tbody>
</table>
LIST OF TABLES

5.6 Comparison of equal error rates for Quantizer Training Scenario 6
using different hashing techniques and different image transforma-
tions. ................................................................. 143

5.7 Average time (in seconds) required to hash a single image using a
2.0 GHz CPU with 1, 2, 3 and 4 CPU cores. ....................... 145
List of Abbreviations

AES Advanced Encryption Standard

CBIR Content Based Image Retrieval

DTFT Discrete Time Fourier Transform

DCT Discrete Cosine Transform

DET Detection Error Tradeoff

DWT Discrete Wavelet Transform

ED Expected Discriminability

EER Equal Error Rate

FM Fourier Mellin

HMAC Hash-based Message Authentication Code

HMM Hidden Markov Model

HOS Higher Order Spectra

LDA Linear Discriminant Analysis

LSH Locality-Sensitive Hashing
LIST OF ABBREVIATIONS

MAC Message Authentication Code
MD Message Digest
MIRFLICKR Multimedia Information Retrieval Flickr Database
NMF Non-negative Matrix Factorization
PCA Principal Component Analysis
PIN Personal Identification Number
PRNG Pseudo-Random Number Generator
ROC Receiver Operating Characteristic
RP Random Projection
SHA Secure Hash Algorithm
SVD Singular Value Decomposition
SVM Support Vector Machine
TIFF Tagged Image File Format
UCID Uncompressed Colour Image Database
XM2VTS Extended MultiModal Verification for Teleservices Security applications Database
List of Publications

Conference Papers


5. V. Chandran and B. Chen, ”Simultaneous biometric verification and random number generation”, in Proceedings, The 5th Workshop on Internet, Telecommunications and Signal Processing, Hobart, Australia, 2006

Journal Papers

1. B. Chen and V. Chandran, ”A Non-linear and Image Dependent Projection for Image Hashing based on Higher Order Spectra”, to be submitted to the IEEE Transactions on Image Processing

2. B. Chen and V. Chandran, ”The Effects of Quantizer Training on Adapative Deterministic Quantization-Based Image Hashing”, to be submitted to the IEEE Transactions on Image Processing
Statement of Original Authorship

The work contained in this thesis has not been previously submitted to meet requirements for an award at this or any other higher education institution. To the best of my knowledge and belief, the thesis contains no material previously published or written by another person except where due reference is made.

Signature: ______________________

Date: ______________________
Acknowledgements

The pursuit of my PhD has been the most challenging task of my life. Its completion would not have been possible without the significant help and contribution of many others. My principal supervisor, Professor Vinod Chandran has provided invaluable assistance throughout the course of my PhD. His knowledge, guidance, support and patience have contributed greatly to my work. I would also like to thank Professor Sridha Sridharan for all he has done in overseeing the SAIVT laboratory. The SAIVT lab is a great environment for researches and many of its members, namely Jamie Cook, Frank Lin, David Wang and Daniel Chen have helped me during my research.

Last, but definitely not least, I am grateful for the support of Jie Han, my family and friends during my postgraduate years.
Chapter 1

Introduction

1.1 Motivation and Overview

The emergence of the digital age has necessitated the need to store and transmit large volumes of digital data. Some of this data is confidential and there is a need to both protect and verify data integrity. Traditional cryptographic techniques such as encryption and hashing have been used to achieve this. Encryption algorithms such as the advanced encryption standard (AES) and hashing algorithms such as the message-digest (MD) and secure hash algorithm (SHA) families are pillars of internet and computer security.

Cryptographic hash algorithms produce a short discrete binary output known as a checksum or hash. The hashing process is both compressive and non-invertible. Compression results in small checksums that can be easily stored and compared, while non-invertibility means the original input cannot be recovered from its checksum. Hashing is a crucial step in generating digital signatures and hash-based message authentication codes (HMAC) used to verify data integrity and data ownership. Checksums are also used for efficient indexing/retrieval of files and to preserve the privacy of password and PIN’s stored in traditional identity authentication systems.

Traditional cryptographic hashing methods are designed for use with non-changing digital data such as files, executables and passwords. As such they are sensitive to even 1-bit changes in the input. This is known as the ‘avalanche effect’ and means even the slightest change in the input will produce large changes in the resulting checksum. However, there are certain types of digital data where small variations cannot be avoided. In digital imaging, many common trans-
formations such as JPEG compression, scaling, cropping and enhancement do not alter the perceptual content of the image but will change the pixel values. In biometric systems, biometric feature data will vary between different acquisitions, for example face biometric features will vary due to pose, illumination and expression.

The sensitivity of traditional hashing methods prevents their use on image and biometric data. Large image and video databases such as Google Images, FLICKR, Youtube and Facebook cannot take advantage of hash based indexing/retrieval, authentication and digital rights management. Similarly, the large biometric databases being compiled by government organisations for border control, voter registration and citizen registration cannot use traditional hashing methods to ensure privacy of templates.

Recent research has attempted to develop robust hash functions that are able to produce identical (or a near identical) checksums for two similar inputs whilst preserving the compressive and non-invertible properties. Robust image hashing was first proposed in 1996 [1] and a primitive method of robust hashing for biometric template security was first introduced by Ratha et al. in 2001 under the term cancellable biometrics [2]. Since then, robust hash functions have been widely applied for number of different applications

- Image indexing and retrieval - images are hashed to produce content dependent binary strings. These binary sequences are relatively short and can be much more efficiently indexed and searched than non-binary features.

- Image watermarking - is used for copyright protection, ownership assertion and integrity verification. A hashed version of the image can be embedded as (or part of) the watermark. The use of a robust hash is advantageous because the watermark is insensitive to common image transformations but is a function of the image itself.

- Biometric template security - biometrics have become an increasingly popular alternative to traditional password authentication systems. However, the storage of large biometric databases poses both privacy and security issues. Biometric data is fuzzy in nature and robust hash functions are one method of protecting biometric data whilst allowing matchability in the hash domain.
1.1 Motivation and Overview

The robust hashing methods used in different applications have developed in isolation but they all consist of a common four-stage process:

1. **Feature Extraction** - Extracts robust features from the input. This step is omitted for biometric template security, since a biometric template already consists of features.

2. **Randomization** - A one-way dimensionality reduction is applied to the extracted feature to introduce compression and non-invertibility. A seed, usually a secret-key, is used to control the randomization. The use of a secret-key serves two purposes. Firstly, hashes can be revoked (cancelled) by altering the secret-key. Secondly, the secret-key can be source/user specific and is required to authenticate data integrity.

3. **Quantization** - The real-valued randomized features are quantized into discrete levels. Quantization requires the estimation of randomized feature pdfs in order to determine threshold locations.

4. **Encoding** - Each quantized level is assigned a unique binary sequence so the output checksum is binary.

Much of the early literature on robust hashing focused on the feature extraction and randomization stages. The randomization methods employed in early algorithms were often simple and easily invertible. The random projection (RP) was one of the first randomization methods to gain considerable popularity. It has been used extensively in many robust hashing algorithms for image hashing [3, 4], biometric template security [5, 6], privacy preserving data mining [7] and locality-sensitive hashing [8]. However, RP is a linear process and can be inverted if enough input/hash pairs are known, this fundamental weakness has been described and practically implemented in [9, 10, 11, 12, 13]. Non-linear randomization techniques would not be as easily invertible as linear techniques but limited research has been aimed at non-linear transformations suitable for use in robust hashing.

The quantization and encoding stages have also been largely ignored until recent work in [14, 15] and [16]. The importance of the Quantization stage on robust hashing performance is identified in [15] but the work is limited to consider only quantization scheme and overlooks the impacts of quantizer training. This is a substantial oversight since effective quantization relies heavily on the accurate
estimation of randomized feature distributions (pdfs) determined in quantizer training.

The work in [14] introduced the distance distortion metric and used it to quantitatively compare the distortion of distance metrics introduced by different encoding schemes. These distortions arise because the distance between two real-valued numbers are not necessarily preserved by the Hamming distances of their respective binary representations. The authors identify Gray-code as the most optimal encoding scheme, but distortions still remain and no work has been directed on improving encoding to reduce or eliminate such distortions. Because the quantization and encoding stages are applied to features obtained from the randomization stage it is possible to address these issues at the randomization stage by producing better features suited for quantization and encoding.

1.2 Research Scope

Robust hash functions are used in numerous applications and many approaches exist. A rigorous analysis of all existing hashing methods across different applications is beyond the scope of a single dissertation. This is because different applications have different performance criteria and certain robust hash functions are simply not applicable for certain applications. The scope of this research dissertation is limited to robust image hashing applications with particular focus on methods using random projection. Random projection is a popular method in all applications of robust hashing because it generalizes to all input data types. Therefore, much of the analysis on random-projection based image hashing quantization and randomization presented in this thesis is directly applicable to other applications such as biometric template security and locality-sensitive hashing.

1.3 Aims and Objectives

This thesis aims to improve robust hashing by examining a number of issues at the quantization, randomization and encoding stages. The first is to examine the importance of quantization training on robust hashing performance, an area that has so far being overlooked in current literature. The second is to develop non-linear randomization techniques that can improve non-invertibility of hashes and allow for improved quantization and encoding.
1.3 Aims and Objectives

Quantization

The quantization stage of robust hashing determines thresholds used to quantize the real-valued randomized features into discrete levels allowing features to be binarised by the encoding stage. The process of quantization requires the learning of feature distributions in order to determine threshold locations. A number of quantization schemes can be used to determine threshold locations and are discussed in section 2.3.3. This thesis examines issues with quantization that prevent accurate feature distributions from being estimated. These issues are:

- Sensitivity of randomized feature distributions to the secret-key or token.
- Sensitivity of the randomized feature distributions to content of training examples.

The presence of these sensitivities can cause quantization to over-train to sample data. This would prevent quantization thresholds from generalising to unseen data and hashing accuracy will decrease. Surprisingly, the methodology applied to evaluate hashing performance in the existing literature uses the same dataset for both quantizer training and performance evaluations. Chapter 3 of the thesis examines these sensitivities in detail and evaluates the hashing performance of three robust image hashing algorithms under scenarios where quantizer training data is different to the evaluation data.

Randomization

Random projection based randomization techniques combine features extracted from the input with the secret-key/token in a linear manner. As stated, this linearity can be exploited to recover the original features or secret-key/token. It has been suggested that randomization should be non-linear or use a secret-key/token that is dependent on the input [9].

Furthermore, the sensitivities at the quantization stage could be alleviated if the randomization stage produced bounded features with fixed (or known) distributions regardless of input or secret-key/token. Distance distortions introduced at the encoding stage could be reduced if randomized features displayed cyclic properties similar to Gray-code.

Chapter 4 explores the use of a simple non-linear transformation used in conjunction with random projection to produce angle features. These features are
CHAPTER 1. INTRODUCTION

theoretically shown to suffer less distance distortion in the encoding stage. The use of a non-linear input dependent random projection is described in Chapter 5, the method is developed for image hashing purposes and relies on higher order spectra (HOS) to combine the input image with the secret-key/token in a non-linear manner.

1.4 Outline of Thesis

The remainder of the thesis is outlined as follow.

Chapter 2 provides the necessary background material related to the fields of traditional cryptographic hashing, robust image hashing and biometric template security. A review of all four stages of the robust hashing process (feature extraction, randomization, quantization and encoding) is provided. This chapter also describes the evaluation metrics commonly used to measure hashing performance and hashing security. A description of baseline systems and datasets used for benchmarking is also contained in this chapter.

Chapter 3 presents an investigation into the sensitivities of hashing performance to quantizer training. This includes a theoretic analysis of the Random projection method showing that distributions of randomized features are a function of the choice of secret-key/token. Experimental evaluations are carried out to verify the theory and show both hashing performance and security are significantly impacted by quantizer training. The implications of these results and the potential security threats are also discussed.

Chapter 4 examines a modification to the random projection method that produces complex-number outputs. Complex numbers can be represented in polar form containing a magnitude and angle. The angle component is of particular interest because it displays numerous properties that are advantageous for robust hashing. Both analytic proofs and experimental evaluations are employed to compare the performance of angle feature versus standard real-valued features for robust hashing.

Chapter 5 describes the advantages of HOS and explores its suitability for extracting robust hash features from images. A HOS based image hashing
1.5 Original Contributions of the Research

This thesis makes a number of contributions to the field of robust hashing. Briefly, these are:

- The analysis of the effects of quantizer training on random projection based robust hashing algorithms. Although the importance of quantization scheme has been previously documented, no study has explored the importance of quantizer training. This research reveals that quantizer training is as important as choice of quantization scheme. Furthermore, this work reveals a number of fundamental weaknesses introduced by quantizer training.

- Improved hashing performance by using angle features. The random projection process is modified to produce a complex output of which only the angle component is retained as the randomized feature. It is shown that the distance distortion suffered by angle features encoded using Gray-code is less than that suffered by real-valued features encoded with Gray-code. Angles are also naturally bounded between $0$ and $2\pi$, this simplifies the quantization process since the limits of the feature distribution is known for all inputs.

- A new method of randomization based on HOS is introduced for robust image hashing. HOS invariants are extracted from the 1D Radon projections of the input image. The extraction of HOS invariants is a non-linear process and is made a function of the secret-key/token by applying a secret-key/token dependent permutation to the 1D Radon projections. The resulting invariants are shown to be insensitive to many common image transformations but highly sensitive to the secret-key/token. The output of HOS is

Chapter 6 concludes with a summary of the thesis contents and highlights the achievements and contributions made. Directions for areas of future research are also discussed.
a set of complex valued vectors that can be used as a set of basis to project a feature vector. We apply the HOS based randomization to a number of different image feature types and obtain superior performance compared to the standard random projection.

- Furthermore, the HOS based randomization produces angle features that display a bounded and near uniform distribution when applied on certain types of image features. This is highly desirable because quantizer training is not required given a bounded uniform distribution. It is shown that hashing performance for the HOS based randomization remains superior to random projection based methods even without quantizer training.
Chapter 2

Review of Robust Hashing

2.1 Introduction

The aim of a robust hash function is to map a real-valued input signal into a short binary string based on its content. Robust hash functions are therefore closely related to cryptographic hash functions which map a large binary input into a small fixed-length binary string. The key difference of robust hash functions is their tolerance to minor incidental changes in the input sequence whilst remaining sensitive to large content changes. Traditional cryptographic hash functions, such as SHA-1 [17] and MD-5 [18], do not have this property because they are designed to be sensitive to even one-bit changes in the input. Robust hash functions have found application in many fields of computer science such as image hashing [4], biometric template security [5, 19], audio hashing [20] and privacy preserving data mining [7]. The primary focus of the remainder of this thesis is on robust image hashing.

Robust Image Hashing

Transformations such as JPEG compression, image scaling and enhancement techniques such as contrast stretching and histogram equalisation are commonly applied to images. These transformations do not significantly alter the perceptual content of the image but may significantly change its pixel values (see Figure 2.1). The presence of these minor changes make traditional cryptographic hash functions unsuitable and the use of robust hash functions in image hashing was first proposed in 1996 by Schneider et al. [1]. Reliably converting an image into a shortened binary sequence has applications in image retrieval, image watermark-
CHAPTER 2. REVIEW OF ROBUST HASHING

ing, broadcast monitoring and content authentication [21].

![Figure 2.1: An original uncompressed image (left) and its JPEG compressed counterpart using 50% quality (right). The images are visually identical but have different pixel values.](image)

Current image retrieval is still mainly reliant on user-provided meta-data such as filenames or keywords and not on the actual appearance of the image. File-names and keywords are unreliable because they can be arbitrarily changed and may not correspond to the content of the image. There also exists ambiguity in certain keywords, an example is shown in Figure 2.2.

![Figure 2.2: Results returned by Google Images using the search query 'House' which can refer to a structural dwelling or to the popular US TV show](image)

The use of robust hashing for image retrieval is different from two closely related fields, content based image retrieval (CBIR) and locality-sensitive hashing (LSH). Content based image retrieval relies on identifying higher level features such as the presence of objects, shapes, colour or textures [22]. It is therefore
possible for CBIR methods to return images that are perceptually very different but contain a common object. Image retrieval using robust hashing relies only on perceptual features and attempts to identify visually identical (or near-identical) images.

LSH is very closely related to robust hashing and can be considered a sub-field. LSH uses approximate search schemes in order to speed up searching large databases. Large image databases such as Google images or Flickr can contain millions of images, nearest-neighbour matching for large databases is computationally expensive. The main objective of LSH is to produce short hashes that can be searched accurately and efficiently \[23\]. LSH differs from robust hashing because security properties such as non-invertibility, secret-key dependence and collision resistance are not important in the field of LSH.

Another application is for digital watermarking, where the image hash is embedded into the image and used for copyright protection and broadcast monitoring. Copyright protection is well known and deals with the detection and prevention of unauthorized use of media content. The presence of a watermark in the image can identify the original author. Broadcast monitoring, tracks the number of times an image or video sequence is used and can be achieved by comparing hashes. Existing techniques can be circumvented by applying content preserving transformations to the image in order to change its digital form so it no longer matches the embedded data without altering perceptual quality. The insensitivity of robust hash functions to such changes is a major advantage and is a key motivation for using robust image hashing in watermarking.

### 2.2 Cryptographic Hash Functions

A cryptographic hash function is a one-way transformation that accepts an input of any size and produces a fixed size output. The cryptographic hash functions used in modern computing require a binary input and produce a fixed length binary output usually around 512 or 1024 bits in length. The process is deterministic, meaning the same input always produces the same output, but is also highly sensitive so even a one-bit change in the input will produce a very different output. Cryptographic hash functions are an essential part of secure digital communications and has many applications in internet communications.

There are two primary benefits of cryptographic hash functions. Firstly, any
arbitrary sized input can be converted to a much smaller sized output. This means
two inputs can be compared much more quickly by using their cryptographic
hashes. Secondly, since the hashing process is one-way it is not possible to obtain
the original input from its hash. By using hashes, untrusting parties can compare
a secret without needing to divulge the secret itself. The first advantage is used
in message authentication where the sender transmits a message along with the
hashed version. Upon receiving the message the recipient can hash the received
message and compare it to the received hash, a mismatch will mean the message
has been modified due to malicious tampering or transmission error. The second
advantage is exploited in password systems for secure password storage. Password
and PIN based verification systems require the storage of the secret password/PIN
to allow verification of users. But passwords are private and should not be stored
in plain text and the use of encryption alone is not ideal because the system
administrator can decrypt and obtain the passwords. Using hashes overcomes
these problems, hashing is deterministic so two passwords can be compared using
only the hashes and the one-way property guarantees that passwords cannot be
accessed by anyone.

A cryptographic hash function \( H \{ \} \) applied to a binary vector or element of
set \( \{0, 1\}^N \) (denoted as \( I \)) is described by Stamp and Low \[24\] as having a number
of necessary properties:

1. Speed: It must be easy to compute the hash function, \( H \{ I \} \) for any input \( I \).

2. Compression: The output of the hash function \( Y = H \{ I \} \) should be small
   (and ideally a fixed length) regardless of the length of the input \( I \).

3. Sensitivity: Every bit of the output should be dependent on every bit of
   the input.

4. Non-invertibility: It should be computationally infeasible to reverse the
   hash function. That is given a hash value \( Y \), we should not be able to find
   an \( I \) such that \( Y = H \{ I \} \).

5. Weak collision resistance: Given an input and its corresponding hash value
   it should be computationally infeasible to find another input that will pro-
   duce the same hash value. That is given \( I \) and \( H \{ I \} \), it should not be
   possible to find \( J \), where \( J \neq I \) but \( H \{ J \} = H \{ I \} \).
6. Strong collision resistance: It should be computationally infeasible to find any arbitrary pair \( I \) and \( J \), where \( J \neq I \) but \( H \{ J \} = H \{ I \} \).

It is important to remember with properties 4 and 5 that collisions are impossible to eliminate since the compression property requires hash functions transform arbitrarily long inputs to small fixed-length outputs. However, it is not the presence of collisions but rather the ability to easily find collisions that is important to collision resistance. In order to satisfy this, cryptographic hash functions display the avalanche effect, where a small change in input causes very large and unpredictable changes in the output.

Many modern cryptographic hash functions such as those in the MD and SHA family satisfy the above properties by using a structure known as the Merkle-Damgard construction. Merkle and Damgard proposed that a non-invertible, collision resistant hash function can be created by iterating a smaller collision resistant compression function. The compression function can be viewed as a hash function but with a small fixed-length input \( L \). The function is a non-linear process and accepts an initial state or value and uses the input block to modify this initial value which becomes the output. Since the overall hash function must work with inputs of any length the input is first split into a set of \( L \)-length blocks, padding is applied if needed to make the input length divisible by \( L \). Each block is processed by compression function in turn and the output is fed back as the initial value for the compression function of the next block. The initial value used for the first block of the first round is usually a fixed constant defined by the hash function. The compression function output of the last block is the resulting hash value.

The MD and SHA families of cryptographic hash functions all follow this general process. They differ in choice of initial value, compression function and creation of the chaining value that is fed back into the next block. A brief overview of the compression function, the chaining value process and how they satisfy cryptographic hash function properties is provided next.

Compression Function: The compression function applied is usually made up of a combination of bitwise operations i.e. AND, OR, NOT and XOR. Hash functions normally possess multiple compression functions (although each is just a different combination of the common bitwise operations) that are used at different stages of the process. These compression functions are designed with the purpose of being highly non-linear, quick to produce the avalanche effect, easy
CHAPTER 2. REVIEW OF ROBUST HASHING

Figure 2.3: The Merkle-Damgard Construction, where the final hash value is created by chaining together the outputs of a compression function applied to each block of the input.

to implement, computationally efficient but difficult to analyse [24]. However, bitwise operations are not non-invertible; given the output and one of the inputs it is easy to find the other input.

Chaining Value: Creation of the chaining value takes the output of the compression function and applies two new kinds of operators, modular addition and bit shift/rotations. It is this use of modular addition that introduces non-invertibility to compression function. For example the modular addition:

\[ A = B + C \mod D \] (2.1)

Given knowledge of \( A, D \) and \( B \) it is not possible to find \( C \) precisely since the wraparound nature of modular arithmetic means an infinite number of values of \( C \) would satisfy the addition. Although it is impossible to identify the exact \( C \) that resulted in \( A \) (thus non-invertible) it is trivial to find a \( C \) that results in \( A \) (thus easy to find weak-collisions). The use of the chaining value as the input for the next block introduces a feedback mechanism. That helps prevent collisions and inversion by preventing the meet-in-the-middle attack.

Each element of the Merkle-Damgard model possesses one of the desired properties of hash functions. Compression function introduces the avalanche effect and collision resistance, modular arithmetic adds non-invertibility and the feedback mechanism also enhances collision resistance. It is the combination of all three
2.2 Cryptographic Hash Functions

elements that produce an overall strong cryptographic hash function. Crypto-
graphic hashing is a crucial component in message authentication and password
protection but it is still vulnerable to attack when used on its own.

Hashing for Password Protection

Password protection schemes rely on the deterministic nature of the hashing pro-
cess to allow verifications of the password to be carried out in the hashed domain.
However, the deterministic nature of hash functions means three vulnerabilities
exist:

1. People often use common dictionary words as passwords. An attacker can
identify these by pre-computing the hashes of dictionary words (called a
hash table) and comparing each stored password with hashes in the hash
table.

2. It is possible to find users with passwords that have the same hash. For
example, if Bob’s password is known and it is found that Alice and Bob have
the same password hash, then Bob’s password can be used to successfully
impersonate Alice.

3. It becomes possible to check if a particular user is using the same password
across different systems (known as cross matching).

To overcome these vulnerabilities a technique called password salting is em-
ployed. In its simplest form, password salting concatenates the associated user-
name with the password prior to hashing. The username is referred to as the
’salt’. The addition of the salt prevents hash-table based attacks because each
user in the system has a unique username, so even if the password is the same
the inclusion of the username will produce unique hashes for every individual in
the system. This protects against both vulnerabilities 2 and 3.

The use of a salt introduces cancelability; an identical password can produce
different hash values by altering the salt. Because the hash function is non-
invertible the salt can be made public without compromising the hashed password.
The first vulnerability is not completely eliminated but is made much harder since
the attacker must now concatenate every username to every dictionary word.
CHAPTER 2. REVIEW OF ROBUST HASHING

Hashing for Data Verification

The task of data verification encompasses both data integrity and data authentication. Data integrity deals with the faithfulness of the transmitted information and incorporates completeness, accuracy and timeliness of the message. Data integrity can be validated by comparing hashes, this involves transmitting the original message along with its hashed version called the checksum. The recipient can hash the received message and compare it to the received checksum. If the two are equal then the received message is the same as the transmitted message.

The use of checksums can ensure data integrity but it cannot verify the identity of the sender. For example, the man-in-the-middle attack involves a malicious third-party hijacking communications between two trusting parties. Suppose Alice and Bob are trusting parties that wish to communicate and Mallory is the malicious third-party. Alice sends a message and checksum to Bob but this is intercepted by Mallory. Mallory can replace the original message and checksum with another valid, albeit fake message and checksum before sending to Bob. Bob will successfully verify the integrity of the received message because Mallory replaced both the message and checksum. As a result Bob has no way of authenticating the message is indeed from Alice.

The man-in-the-middle attack can be overcome by a number of cryptographic techniques. The simplest is the Hash-based Message Authentication Code (HMAC) which generates a hash-like binary sequence known as the message authentication code (MAC). Unlike a standard hash, a MAC is a function of both the message and a secret-key. In the above example, the secret-key must be known by both Alice and Bob but kept secret from Mallory. If Mallory intercepts a message, she cannot replace it without detection since she does not know the secret-key required to generate the MAC. So when Bob receives a message along with its MAC, he can verify using the MAC to ensure data integrity and authenticity.

2.3 Overview of Robust Hashing

The applications of cryptographic hash functions namely, message authentication and secure matching provide advantages that are highly desirable in both image processing and biometrics. Cryptographic hash functions are unsuitable for this because unlike passwords and PINs, biometrics from the same user and perceptu-
2.3 Overview of Robust Hashing

ally identical images can contain variations. Acquisitions of the same biometric will vary due to natural changes in the biometric and different environmental conditions at time and place of acquisition. Similarly, an image can be altered so that it remains perceptually identical but have very different pixel values. Cryptographic hash functions by design are highly sensitive to these variations and cannot produce useful hashes.

Robust hash functions have been proposed as an alternative to cryptographic hash functions in image hashing, biometric template security as well as other fields such as privacy preserving data mining and audio hashing. A robust hash function has many of the same properties as a cryptographic hash function except for the sensitivity to small changes in the input. In other words, it is a hash function that is robust to minor changes.

Robust hash functions need to tolerate minor changes in the input and this complicates the matching process. Firstly, one must quantifiably define how much change is minor and tolerable. This is difficult because the concept of similarity is subjective and dependent on the application at hand. In an image hashing context, perceptually identical versions of the same image should produce the same hash. While in biometrics, features obtained from the same individual should all hash to the same value. Two inputs that produce the same hash can be defined as belonging to the same class. Although the concept of class is logically simple it is difficult to quantifiably define. Machine learning and pattern classification algorithms attempt to separate inputs into distinct ‘classes’ using a two-stage process of feature extraction followed by feature matching. Robust hashing follows this methodology, and similarity is measured by comparing some underlying feature. The similarity between two hash vectors of equal length can be quantifiably measured using a meaningful distance metric, such as Euclidean Distance or Hamming Distance, and a threshold (τ) can be set to determine the amount of tolerated change.

Feature extraction, is a method of dimensionality reduction that attempts to extract useful information about the input while discarding the redundant data. A good feature extractor produces features (information) that enables the accurate and reliable separation of inputs into meaningful classes. Once features are extracted, feature matching can be performed. Feature matching is a quantifiable way of measuring the similarity between two (or more) sets of features. It is used to identify which class the input most likely belongs to.
### Table 2.1: Metrics used in this dissertation for measuring similarity between two vectors $a$ and $b$ of dimension $N$. For Hamming distance, the vectors are assumed to be binary valued in each dimension.

<table>
<thead>
<tr>
<th>Name</th>
<th>Notation</th>
<th>Formula</th>
</tr>
</thead>
<tbody>
<tr>
<td>Manhattan (or Cityblock) Distance</td>
<td>$d_1(a, b)$</td>
<td>$\sum_{i=0}^{N}</td>
</tr>
<tr>
<td>Euclidean Distance</td>
<td>$d_E(a, b)$</td>
<td>$\sqrt{\sum_{i=0}^{N} [a(i) - b(i)]^2}$</td>
</tr>
<tr>
<td>Normalized Hamming (or Edit) Distance</td>
<td>$d_H(a, b)$</td>
<td>$\frac{\sum_{i=0}^{N} \text{XOR}(a(i), b(i))}{N}$</td>
</tr>
</tbody>
</table>

A multitude of techniques such as distance measures, statistical models, neural networks, support vector machines and hidden Markov models have been used. The method that is ultimately chosen often comes down to the application and the type, length and format of features extracted. Throughout this dissertation the distance measures presented in table 2.1 are used.

Euclidean distance is best suited for comparing real-valued features produced by the feature extraction and randomization stages. The Manhattan distance is used to compare the discrete integer features after the quantization stage. The final hash outputs are fixed length binary vectors and Hamming distance is a simple and intuitive way of comparing two hashes. In this dissertation, the Hamming distance is normalized by the length of the binary feature vector. This produces a value between 0 and 1 that represents the proportion of binary features that are the same, where 0 represents no bits are the same and 1 meaning all bits are the same.

In pattern recognition, two feature sets are deemed to match if their distance is less than a predetermined threshold ($\tau$). A cryptographic hash function is designed to have no robustness and performs matching using the most strict threshold of $\tau = 0$ (no error tolerance). In a cryptographic hash function two hashes are a match only if every bit is identical. But a robust hash function must tolerate variation in the input, this is achieved using a pattern recognition framework of feature extraction in conjunction with feature matching using a threshold $\tau > 0$.

A robust hash function does not require a binary input, this is because many image and biometric features are real-valued in nature. Features are extracted from $I$ and combined with the secret-key $K$. Since both the features and the
secret-key are real-valued a quantization and encoding stage is employed to produce the final binary hash output \( Y \). The secret-key \((K)\) is derived a password or PIN that serves as the seed of a pseudorandom number generator (PRNG) used in the randomization stage hashing. Thus \( Y \) is a set of binary vectors of lower dimension and the normalized Hamming distance is used to compare hashes quickly.

We define a robust hash function as \( H \{ \} \) which takes an input \( I \) and secret-key \( K \) to produce an output hash \( Y \).

\[
Y = H \{ I, K \}
\]  

(2.2)

It is widely accepted that a robust function should display the following desirable properties.

1. Robustness - Image hashes \( Y_1 \) and \( Y_2 \) obtained from two inputs belonging to the same class, \( I_1 \) and \( I_2 \) should have a hamming distance less than a selected acceptance threshold \( \tau \) with high probability.

2. Fragility - The hashes \( Y_1 \) and \( Y_2 \) obtained from inputs of different classes, \( I_1 \) and \( I_2 \). Should, with high probability, have a distance greater than \( \tau \). Fragility is the sensitivity of the hash to inputs that display major differences.

3. Unpredictability - Two hashes \( Y_1 \) and \( Y_2 \) obtained from an identical \( I \), but with two different secret-keys \( K_1 \) and \( K_2 \) should with high probability, have a distance greater than \( \tau \). Unpredictability can also be viewed as the sensitivity of the hash to different secret-keys.

4. Security - The hashing process should be key-dependent so that \( Y \) cannot be obtained without knowledge of \( K \) even if \( I \) is known.

5. Non-invertability - The input \( I \) should not be recoverable from \( Y \) even with full knowledge of \( K \) and \( H \{ \} \). Likewise the secret-key \( K \) should not be recoverable from \( G \) and \( I \).

Current robust hashing techniques contain four distinct processes: feature extraction, randomization, quantization and encoding. A flowchart of this process is shown in Figure 2.4.
Feature extraction is needed to satisfy the robustness property or robust hashing. Hashes are generated from the input using extracted features that display invariance to common intra-class variations. For example, the features for image hashing should be invariant to image transformations such as compression and noise. This step is omitted from certain applications such as biometric template security, since a biometric template is itself a set of invariant features no further features are extracted. The output of the feature extraction stage is a length \( N \) real-valued feature vector \( \mathbf{F} = [F_1, F_2, \ldots, F_N] \).

The process of feature extraction is often invertible and not secret-key dependent. A randomization step is therefore required to introduce these properties. Randomization is achieved using a one-way secret-key dependent function. It also serves as dimensionality reduction where the real-valued \( N \)-dimensional input feature vector \( \mathbf{F} \) is compressed into the real-valued randomized feature vector \( \mathbf{R} = [R_1, R_2, \ldots, R_M] \) of length \( M \) where \( M < N \).

Quantization is used to discretize the randomized features \( \mathbf{R} \) into a vector of discrete quantized features \( \mathbf{Q} = [Q_1, Q_2, \ldots, Q_M] \). The quantization phase does not binarize the data, it only quantizes data into discrete levels based on a set of thresholds. The actual number of levels and the decision criteria used to determine threshold locations are important in the quantization stage.

It is the final encoding stage that converts the quantized features \( \mathbf{Q} \) into a binary sequence \( \mathbf{Y} = [Y_1, Y_2, \ldots, Y_{M \times b}] \). This is done by assigning each quantization level with a unique bit-string of length \( b \). Each real-valued feature is assigned the bit-string corresponding to the quantization level it belongs to. Encoding schemes assign bit-strings using either standard binary or GrayCode.

Research has shown that all four steps can have profound impact on hashing performance. The first two steps are not always sequential and the randomization can occur as part of the feature extraction process. Despite this, the four processes are usually independent of each other, as such the following section is split into four parts with each part being a review of each of the four steps of robust hashing.

### 2.3.1 Feature Extraction

Numerous feature extraction techniques have been used for image hashing. This is due to the wealth of image processing research conducted in recent decades and the development of many feature extraction methods. Early image hashing methods used simple statistical features such as the mean, variance or moments of the
2.3 Overview of Robust Hashing

Figure 2.4: The four steps of the robust hashing process, using a 2-bit standard binary quantizer where $N = 7$ and $M = 4$. The feature vector $F$ and quantization thresholds used are arbitrarily selected for illustrative purposes.

$F = [F_1, F_2, \ldots, F_N] = [2.1, 3.4, -5.3, 0.2, 9.6, -2.4, 1.4]$

$R = [R_1, R_2, \ldots, R_M] = [1.3, -2.9, -0.3]$

$Q = [Q_1, Q_2, \ldots, Q_M] = [3, 4, 1, 2]$

$Y = [Y_1, Y_2, \ldots, Y_{M\times b}] = [1, 0, 1, 1, 0, 0, 0, 1]$
CHAPTER 2. REVIEW OF ROBUST HASHING

greyscale intensity values of image subsections \[1\]. The robustness properties and discriminant ability of these basic features are limited because two perceptually different images can have similar histograms. Other techniques rely on extracting features from coarse representations of the image. Venkatesan \[27\] used averages of low-frequency sub-bands of the discrete wavelet transform (DWT) and the variances of high-frequency sub-bands. Mihcak \[28\] proposed a technique using only the DC (lowest frequency) sub-band of the discrete wavelet transform (DWT). The discrete cosine transform (DCT) has also been used to extract coarse image information, and methods such as those by Lin \[29\] and Fridrich \[30\] use the low-frequency coefficients of the DCT. Swaminathan \[4\] proposed a technique based on low-frequency Fourier-Mellin coefficients. Features are extracted by summing along a set of radii (\(\rho\)) in the frequency domain representation of the image (see Figure 2.5). The features are restricted to the lower frequency components by limiting the radii to be between 0 and 0.4 of the normalized radius of the image.

![Figure 2.5: Swaminathan’s technique obtains features by summing along circles of different radii, \(\rho\) (rho) in the frequency domain. Three such circles are shown above with \(\rho\) of 0.02, 0.2 and 0.4. Note that in the above image the frequency domain is log-transformed then normalized for display.](image)

Techniques based on low rank matrix approximations are also motivated by retaining coarse image content, since the low rank components preserve structural information but discards the fine detail that are sensitive to image transformations. An example of such a low-rank matrix decomposition is shown for the Lena image in Figure 2.6. Their advantage over other low frequency methods

22
is superior performance under lossy geometric transformations such as cropping. Singular value decomposition (SVD) was the first to be proposed \cite{31} and more recently Monga’s non-negative matrix factorization (NMF) has gained attention \cite{3}. Unlike SVD which produces bases that are orthogonal, NMF produces bases that are positive and therefore better represents the additive parts-based nature of images.

![Figure 2.6: The Lena image and its reconstructions using NMF decomposition.](image)

Alternatively, some methods have used high frequency image features such as edges in \cite{32}, where a 3x3 Prewitt’s operator is applied to an image to obtain the edge map. The image is then subdivided into 250 overlapping blocks of size 40x40 pixels, a single feature component is extracted from each block by taking the mean gradient coefficients. Although shown to be superior to other hashing methods the experimental analysis was carried out using only a very small data set of 87
CHAPTER 2. REVIEW OF ROBUST HASHING

images and the discriminant ability of these edge coefficient features is yet to be validated. Other high-frequency feature methods use landmark feature points such as corners and line-ends, one such method [33] uses end-stopped wavelets to extract image features. End-stopped wavelets behave similarly to cells in the human visual cortex that respond strongly to corners and other points of high curvature [33]. The technique searches for robust feature points iteratively. At each iteration, the image is smoothed and feature points are extracted using the end-stopped wavelet and compared with the feature points from the previous iteration. The process repeats until feature points from subsequent iterations are near identical. This results in a set of prominent feature points in the image that are consistently detected after repeated smoothing. However these techniques display either poor robustness or are computationally expensive [3].

Parallel beam projections obtained from the Radon transform have also been used as image hashing features, first by Lefebvre [34] and more recently by Ou [35] who extracted DCT coefficients from the projection vectors. However, Radon transform based methods have not gained significant traction in the field.

### 2.3.2 Randomization

Feature randomization ensures that the final hash vector ($G$) is dependent on the secret-key ($K$). So that without knowledge of the correct key the correct hash vector cannot be produced. The secret-key is used as the seed for the PRNG that produces random numbers needed in the randomization step. Simple forms of randomization involve permuting the feature vector or selecting a random subset of values from the feature vector but these have long been considered insecure. More recent techniques using random projections have gained popularity. Random projections have been used extensively in image hashing [3, 4], biometric template security [5, 6], privacy preserving data mining [7] and audio hashing [36].

### Random Projections

Random projection belongs to a class of dimensionality reduction techniques that use a sub-space projection. This class of projections include the well known principal component analysis (PCA) and linear discriminant analysis (LDA) techniques that are fundamental to the field of pattern recognition. These methods
2.3 Overview of Robust Hashing

involve projecting an \(N\)-dimensional vector into a \(M\)-dimensional subspace where \(M < N\). The subspace is usually chosen based on some statistical criteria. In PCA, the subspace is selected to minimize the mean-square error of the resulting projection. Whilst LDA selects a subspace in which inputs from different classes can be most easily separated. In both methods, identifying the appropriate subspace requires training examples in order to learn the statistical properties of the input vectors. However, sufficient training examples are required in order to identify a meaningful and accurate subspace. This is a major limitation of techniques such as PCA and LDA and is known as the small sample size problem.

The random projection is much simpler than its PCA and LDA counterparts. It’s aim is to preserve Euclidean distance relationship in the projected subspace. It is shown by Johnson-Lindenstrauss Lemma [37] that Euclidean distance can be adequately preserved by projecting into any subspace where the projection basis are randomly drawn from a Gaussian distribution of zero-mean and unit variance. This is a major advantage of random projection, since no training examples are needed to identify an appropriate subspace. Furthermore, the distance preservation property allows matching of inputs to be carried out using the projected versions of the input, thus keeping the original inputs private.

Like all subspace projections the random projection is performed using matrix multiplication. If \(F = \begin{bmatrix} F_{i,j} \end{bmatrix}_{1 \times N}\) is the input feature vector and \(V = \begin{bmatrix} V_{i,j} \end{bmatrix}_{N \times M}\) is the randomly chosen basis then the result of the random projection is:

\[
R = F \times V
\]  

(2.3)

If we let \(v_k\) denote the \(k\)-th column vector of \(V\), that is \(v_k = V_{1...N,k}\) then equation (2.3) can be rewritten as

\[
R = (\langle F, v_1 \rangle, \langle F, v_2 \rangle, \ldots, \langle F, v_M \rangle)
\]  

(2.4)

Where \(\langle \rangle\) denotes the inner product. It can be seen from (2.4) that the \(k\)-th value in \(G\) is the inner product of the original feature vector \(F\) with the \(k\)-th column of \(V\). Compression and non-invertibility of the original feature vector will be achieved if \(M < N\).

In image hashing and biometric template security where a secret-key is required the projection basis \(V\) is secret-key dependent. In applications that do not require a secret-key, such as locality-sensitive hashing for image retrieval the
projection basis \( \mathbf{V} \) is constant. But regardless of whether a secret-key is required or not the randomization process is the same and involves the matrix multiplication of \( \mathbf{F} \) and \( \mathbf{V} \).

However, matrix multiplication is a linear transformation, if an attacker has knowledge of multiple input/hash pairs that are projected using the same basis \( \mathbf{V} \), they can create a set of linear equations from which \( \mathbf{V} \) or \( \mathbf{F} \) can be obtained. For example, given features derived from two different inputs where \( \mathbf{F} = (F_1, F_2) \) and \( \mathbf{F}' = (F'_1, F'_2) \) if both are projected with \( \mathbf{V} = \begin{pmatrix} V_{1,1} & V_{1,2} \\ V_{2,1} & V_{2,2} \end{pmatrix} \) (where \( M = 2 \)) then from (2.3):

\[
\begin{align*}
\mathbf{R} &= (F_1.V_{1,1} + F_2.V_{2,1}, \quad F_1.V_{1,2} + F_2.V_{2,2}) \\
\mathbf{R}' &= (F'_1.V_{1,1} + F'_2.V_{2,1}, \quad F'_1.V_{1,2} + F'_2.V_{2,2})
\end{align*}
\] (2.5)

Using (2.5) we can create the following system of equations

\[
\begin{align*}
\mathbf{R}(1) &= F_1.V_{1,1} + F_2.V_{2,1} \\
\mathbf{R}(2) &= F_1.V_{1,2} + F_2.V_{2,2} \\
\mathbf{R}'(1) &= F'_1.V_{1,1} + F'_2.V_{2,1} \\
\mathbf{R}'(2) &= F'_1.V_{1,2} + F'_2.V_{2,2}
\end{align*}
\]

Since \( \mathbf{R} \), \( \mathbf{R}' \), \( \mathbf{F} \) and \( \mathbf{F}' \) are known, \( \mathbf{V} \) can be determined provided the above set of fully determined equations is not ill-posed. Even with less than \( M \)-sets an estimate of \( \mathbf{V} \) is still obtainable using techniques for solving under determined systems such as the least-squares method. The estimate of \( \mathbf{V} \) becomes more accurate as the number of input/hash pairs increase and Mao et al. [10] shows the task of estimating \( \mathbf{V} \) is similar to Shannon’s unicity distance attack on ciphers.

This weakness has been practically demonstrated for Swaminathan’s image hashing scheme [10], Monga’s image hashing scheme [9] and random projection based privacy preserving data mining [11]. The weakness can be eliminated by removing the linear dependence between every input feature vector and the corresponding projection but the challenge is in doing so while preserving distances such that matching can still be performed in the projected space. The author’s of [9] suggest the use of input-dependent secret-keys but no discussion on how such keys can be generated were provided.
2.3 Overview of Robust Hashing

Importance of the Secret-key

The presence of a secret-key is a standard feature of robust hashing algorithms. It is however, not essential in every application of robust hash functions. For example in content based retrieval the secret-key is unnecessary and can be removed or kept constant. However, in many other applications the secret-key it is a critical component. In authentication and integrity verification applications (such as image watermarking) both content and source must be verified. The output hash needs to be a function of both the input content and the user. The secret-key serves as a unique identifier belonging to the content author. Therefore when the hash is checked, it needs to be generated using the secret-key associated with the supposed author and this allows integrity to be verified [38]. In biometric template security the secret-key behaves like the salt used in password protection schemes described in Section 2.2. Because the hash is a function of both the biometric template and the secret-key, it is necessary in order to prevent cross matching across multiple databases that contain the same biometric. Further, cancelability requirement of biometric template security schemes can be simultaneously satisfied by altering the secret-key [39].

2.3.3 Quantization

The output of the feature extraction and randomization stages are real-valued. However, searching through large databases using real-valued distance measures such as Euclidean distance is computationally expensive. Instead, randomized features are often binarized by first quantizing the real-valued vector into one of discrete integers. The encoding stage (discussed in the next section) then assigns a unique binary sequence to each discrete quantization level.

Quantization in image hashing converts the the real-valued randomized feature vector, \( \mathbf{R} = (R_1, R_2, \ldots, R_M) \), into a discrete vector \( \mathbf{Q} = (Q_1, Q_2, \ldots, Q_M) \) with \( L = 2^b \) discrete levels where \( Q = \{ q \in \mathbb{Z} \mid 1 \leq q \leq L \} \). The number of discrete levels \( L \) is restricted to a power of 2 so that the subsequent encoding (or binarization) stage can assign each of the \( L \) discrete levels with a unique \( b \)-bit binary sequence.

The process of quantization involves thresholding real-valued data. The simplest quantization scheme is the uniform quantizer, which produces equal-width intervals using thresholds uniformly spaced between the minimum and maximum
values of the randomized feature component. These minimum and maximum values are learnt from randomized features produced using a training set. According to [15] the minimum and maximum value for each of the $M$ components are the minimum and maximum values of all randomized feature components pooled together. If $L$ denotes the number of quantization levels then the distance between neighbouring quantization thresholds $X_a$ and $X_{a+1}$ for $a = 1, 2, \ldots, L - 1$ is:

$$\frac{\max(R) - \min(R)}{L}$$  \hspace{1cm} (2.6)

The uniform quantizer produces a single set of thresholds that divides the feature space into $L$ equal-distance levels. However, for non-uniformly distributed features a uniform quantization will result in some quantization levels having more samples than others (see Figure 2.7). This reduces the entropy and fragility of the resulting binary hashes making them more easily guessed. Furthermore, it has been shown that the pdf of different feature components can be significantly different from each other and this also means the pdf of a single feature component can be different from the pdf of all feature components combined [15]. In other words, the pdf of the first randomized feature component $p(R(1))$ is not necessarily equal to the pdf of the second randomized feature component $p(R(2))$ nor is it equal to the pdf of all feature components pooled together. Under such scenarios the simple uniform quantizer would perform poorly since $\max(R)$ and $\min(R)$ are the maximum and minimum feature values for all $R$ and may not
produce optimal thresholds when applied to each individual feature component.

Adaptive deterministic quantization, first proposed by Lloyd [40] and Max [41] in the 1960’s for digital communications was first used in robust image hashing by Mihcak [20]. Unlike uniform quantization, which has a single set of equal-distance thresholds used for all feature components. Adaptive deterministic quantization uses $M$ sets of equal-probability thresholds, where each of the $M$ sets is used for the corresponding feature component. The thresholds in adaptive deterministic quantization are therefore obtained separately for each of the $M$ randomized feature components, using only the pdf of the relevant component ($p(R(i))$) where $i = 1, 2, \ldots, M$. Thresholds are placed such that each quantization level occurs with equal probability. Thus any two neighbouring quantization thresholds for the $i$-th feature component $X^i_a$ and $X^i_{a+1}$ for $a = 1, 2, \ldots, L - 1$ must satisfy:

$$\int_{X^i_a}^{X^i_{a+1}} p(R(i)) \, df = \frac{1}{L} \quad (2.7)$$

This produces quantized feature vectors where each discrete level for every feature component occurs with equal probability. This has the desirable effect of maximising the entropy and fragility of the resulting hash. For example, consider a 1-bit deterministic adaptive quantizer, where 1’s and 0’s both occur with probability $p = 0.5$ and each feature bit has the maximum entropy of $-p \log_2(p) - (1 - p) \log_2(1 - p) = 1$. Zhu et al. [15] showed that the maximum fragility of $\frac{1}{2}$ can be achieved when deterministic adaptive quantization is used and higher fragility usually corresponds to a better equal error rate (EER).

Since adaptive deterministic quantization decides bit assignments using the pdf of feature components, accurately determining feature component pdfs is crucial. However, image hash features are a function of both the input image and the secret-key, changing either can alter the distribution of feature components. This has significant consequences for image hashing applications because the quantizer thresholds learnt from training data may not divide unseen data into equal probability levels. This would result in a reduction in fragility, entropy and EER performance.

The effects of quantization scheme (uniform of adaptive) on hashing performance has been analyzed by [15], but the importance of quantizer training, particularly the accuracy of the feature pdf, has so far been ignored. In current literature, hashing performance is evaluated with the same data used to train the
quantizer. This guarantees that the feature pdf used in quantizer training is a perfect match to the test data and the resulting performance evaluation shows only the best-case scenario. This reliance on accurate quantization training has a number of performance and security implications which are further explored in Chapter 3.

2.3.4 Encoding

The encoding step converts the quantized features into a binary bit-string by assigning each discrete level \( l \) with a unique bit-string of \( b \)-bits where \( l = 1, \ldots, L \) and \( L = 2^b \). The two most common forms of encoding use standard Binary and Gray code. The binary-codeword matrix \( B^{(b)} \) is first defined as

\[
B^{(1)} = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \quad \text{and} \quad B^{(k+1)} = \begin{bmatrix} z^{(k)} & B^{(k)} \\ o^{(k)} & B^{(k)} \end{bmatrix}
\]  

(2.8)

The values \( z^{(k)} \) and \( o^{(k)} \) are column vectors of length \( 2^k \) containing zeros and ones respectively. Similarly, the Gray-codeword matrix \( G^{(b)} \) is defined using

\[
G^{(1)} = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \quad \text{and} \quad G^{(k+1)} = \begin{bmatrix} z^{(k)} & G^{(k)} \\ o^{(k)} & \tilde{I}^{(k)}G^{(k)} \end{bmatrix}
\]  

(2.9)

where \( \tilde{I}^{(k)} \) is the \( 2^k \times 2^k \) exchange matrix containing 1’s along the anti-diagonal and zero’s elsewhere. Let \( b_i^{(b)} \) for \( i = 1, \ldots, 2^b \) denote the \( i \)-th row of \( B^{(b)} \) and \( g_i^{(b)} \) for \( i = 1, \ldots, 2^b \) denote the \( i \)-th row of \( G^{(b)} \), then encoding is performed by assigning each of the \( L \) discrete quantization levels with the binary string represented by the corresponding row in either \( B^{(b)} \) or \( G^{(b)} \). That is the \( l \)-th quantization level is encoded as \( b_l^{(b)} \) or \( g_l^{(b)} \) where \( l = 1, \ldots, L \) and \( L = 2^b \). Gray code has an advantage over standard binary because neighbouring codewords in Gray code always differ by only a single bit.

In 2006 Swaminathan et al. [4] were the first to introduce randomness at the encoding stage. The approach applied a secret-key dependent permutation to the final binary hash. Recently in 2011, Zhu et al. [14] showed that permutation of the final binary hash does not change the number of '0' and '1' in the binary hash. Zhu et al. proposed an alternative method of randomizing the encoding stage using an encoding technique called random Gray code. This technique involves randomizing the Gray codeword matrix \( G^{(b)} \) by circularly shifting the rows by an
amount determined using the secret-key. A circular shift preserves the Hamming distance relationship between neighbouring codewords, \( d_H(g_l^{(b)}, g_{l+1}^{(b)}) = 1 \), which is not the case with permutation. Randomizing the Gray codeword matrix in this manner has the added advantage of changing the number of ‘0’ and ‘1’ in the final hash, which is not possible with a permutation of the final hash.

\[ \text{mean} [d_H (Y_c, Y_{cm}^c)] = \frac{1}{N \times M} \sum_{c=1}^{N} \sum_{m=1}^{M} d_H (Y_c, Y_{cm}^c) \]  

(2.10)

Robustness is the average intra-class distance of hashes, intra-class examples are slightly modified versions of an original image. A lower value indicates higher robustness.

Fragility is determined by the sensitivity of the hash function to inter-class changes in the input. This can be measured using the expected discriminantability metric proposed by Zhu [15] which is the average Hamming distance between...
hashes produced from inputs belonging to different classes. If the test set contains \( N \) classes then the expected discriminantability is:

\[
\text{mean } [d_H (Y_c, Y_d)] = \frac{2}{N^2 - N} \sum_{c=1}^{N-1} \sum_{d=c+1}^{N} d_H (Y_c, Y_d) \quad (2.11)
\]

This is equivalent to the average inter-class distance between hashes and a higher value indicates better fragility. In order to measure robustness and fragility accurately the secret-key \((K)\) must be kept constant. This ensures that any variation in the final hash is due only to changes in the input and not to changes in the secret-key \((K)\). The effects on the output hash due to varying \(K\) is considered by the unpredictability property and is discussed in the following section.

Fragility and robustness cannot be used in isolation when evaluating robust hash functions. Consider a robust hash function \(H\) that behaves like a uniform random number generator. Such a system would display excellent fragility but no robustness. On the other hand, a robust hash function that produces a fixed output regardless of the input would have perfect robustness but zero fragility. The relationship between fragility and robustness requires the two properties to be considered jointly. This is not unlike pattern recognition evaluation, where average intra-class distance and average inter-class distance can be jointly considered by evaluating verification performance.

The task of verification can be simplified down to a decision,

\[
\text{if } d_H (Y_{\text{claimed}}, Y_{\text{input}}) < \tau \text{ then } Y_{\text{input}} = Y_{\text{claimed}} \text{ else } Y_{\text{input}} \neq Y_{\text{claimed}}.
\]

The value \(\tau\) is the decision threshold. If two hashes have a hamming distance less than \(\tau\) they are accepted as being from the same class. If the hamming distance is larger than \(\tau\) then the two hashes are judged as being from different classes. In some applications a second threshold \(\tau_2\) is used where \(\tau > \tau_2\). This creates an intermediate region designated as do not know. In this thesis only a single threshold is used producing only definite accept or reject decisions.

It is clear that the choice of \(\tau\) has a significant impact on verification performance. A small value of \(\tau\) reduces false matches (also known as false acceptances) but increases false non-matches (also known as false rejections or misses). Similarly, a large value of \(\tau\) has the opposite of effect of increasing false matches (also known as false alarms) but decreasing false non-matches.
2.4 Evaluating Robust Hash Functions

Since verification performance is a function of $\tau$, performance evaluation is carried out by varying the value of $\tau$ from one extreme to the other. At each point, the false non-match rate and false match rates are recorded. This data can be visualised by plotting the curve of the false non-match rate versus the false match rate known as the receiver operating characteristic (ROC) curve or the detection error tradeoff (DET) curve. This curve gives an indication of verification accuracy regardless of $\tau$, an example of which is shown in Figure 2.8. ROC and DET curves allow verification performance to be compared visually.

Figure 2.8: Example of a ROC curve, the black dashed line marks in the line of Equal Error Rate.

However, if a single value is required to quantify accuracy then the false non-match/false match rate at a single point along the ROC or DET curve must be quoted. The most commonly used point is where false non-match = false match. This value is known as the equal error rate (EER) and along with ROC curves is the principal value used to evaluate hashing accuracy in this dissertation.

2.4.2 Distance Distortion

The accuracy properties of robust hashes are effect by all stages of the robust hashing process. While robustness, fragility and EER are useful tools in deter-
mining accuracy of the overall system they give little indication on the individual phases of the hashing process. It is only recently that researches have attempted to quantify accuracy of the individual steps of the hashing process.

In 2011, Zhu [14] analyzed the encoding phase of image hashing and revealed that encoding introduces distance distortion that can effect the EER of the hashing system. The concept of distance distortion is best illustrated with an example. Consider a real-valued randomized feature \( R \) quantized using a 3-bit quantizer. Suppose that \( R_1 \) is quantized into the 4th quantization level and the same feature from another image \( R_2 \) is quantized into the 5th level. \( R_1 \) and \( R_2 \) are therefore in neighbouring quantization levels and have a quantized magnitude distance in \( L_1 \) of 1. Now assume that \( R_1 \) and \( R_2 \) are encoded using standard 3-bit binary, \( R_1 \) becomes \( b^{(3)}(4) = 011 \) (binary representation of 4) and \( R_2 \) becomes \( b^{(3)}(5) = 100 \) (binary 5) and the Hamming distance between \( b^{(3)}(4) \) and \( b^{(3)}(5) \) is 3. It can be seen that \( d_1(R_1, R_2) \neq d_H(b^{(3)}(4), b^{(3)}(5)) \) where \( d_1 \) denotes the \( L_1 \)-norm distance and \( d_H \) is the Hamming distance and the encoding process has introduced a distance distortion.

This distortion can be theoretically calculated for any encoding scheme [14]. A number of necessary definitions are first given.

**Definition 1.** (From Zhu et al. [14]). The matrix \( D_i^{(b)} = [\alpha_{i,j}^{(b)}]_{L \times L} \) for \( b \in \mathbb{Z}^+ \) is called the standard-distance matrix of bit-order \( b \), where \( \alpha_{i,j}^{(b)} = |i - j| \) for \( i, j = 1, \ldots, L \) and \( L = 2^b \). In other words, \( \alpha_{i,j} \) is the \( L_1 \)-norm distance between the \( i \)-th and \( j \)-th quantization levels when no encoding is applied.

**Definition 2.** (From [14]). The matrix \( D_B^{(b)} = [\beta_{i,j}^{(b)}]_{L \times L} \) for \( b \in \mathbb{Z}^+ \) is called the binary-distance matrix of bit-order \( b \) where \( \beta_{i,j}^{(b)} = d_H(b_i^{(b)}, b_j^{(b)}) \) for \( i, j = 1, \ldots, L \) where \( b_i^{(b)} \) and \( b_j^{(b)} \) are the \( i \)-th and \( j \)-th rows of the standard binary matrix \( B^{(b)} \) and \( L = 2^b \). In other words, \( \beta_{i,j} \) is the Hamming distance between the standard binary representation of the \( i \)-th and \( j \)-th quantization levels.

**Definition 3.** (From [14]). The matrix \( D_G^{(b)} = [\gamma_{i,j}^{(b)}]_{L \times L} \) for \( b \in \mathbb{Z}^+ \) is called the Gray-distance matrix of bit-order \( b \) where \( \gamma_{i,j}^{(b)} = d_H(g_i^{(b)}, g_j^{(b)}) \) for \( i, j = 1, \ldots, L \) where \( g_i^{(b)} \) and \( g_j^{(b)} \) are the \( i \)-th and \( j \)-th rows of the Gray code matrix \( G^{(b)} \) and \( L = 2^b \). In other words, \( \gamma_{i,j} \) is the Hamming distance between the Gray code representation of the \( i \)-th and \( j \)-th quantization levels.

From the above definitions the distance distortion introduced when a standard
real-valued feature is encoded using standard binary can be expressed by:

\[
M(D_{I,B}^{(b)}) = \frac{1}{L^2} \sum_{i=1}^{L} \sum_{j=1}^{L} |\alpha_{i,j}^{(b)} - \beta_{i,j}^{(b)}| 
\]  

(2.12)

and similarly when a real-valued feature is encoded using Gray code:

\[
M(D_{I,G}^{(b)}) = \frac{1}{L^2} \sum_{i=1}^{L} \sum_{j=1}^{L} |\alpha_{i,j}^{(b)} - \gamma_{i,j}^{(b)}| 
\]  

(2.13)

According to equations 2.12 and 2.13 the distance distortion of encoding is the average difference between the ideal distance \(\alpha_{i,j}\) and the appropriate Hamming distance either \(\gamma_{i,j}\) for GrayCode or \(\beta_{i,j}\) for standard binary. Zhu et al. [14] showed that distance distortion can be used to compare performance of different binary encodings, those with less distance distortion have better hashing performance.

### 2.4.3 Security Properties

The unpredictability of robust hash functions was first quantifiably measured by Swaminathan [4]. His approach used differential entropy to measure the randomness of the real-valued randomized features extracted from a fixed input using different secret-keys \(K\). It is well known in information theory that the differential entropy of a random variable is a function of its distribution and the variance of the distribution. The Gaussian distribution has the largest differential entropy compared to all other distributions for a given variance. This has motivated many methods to use feature extraction and randomization processes that produce Gaussian distributed features. However, there are two problems with measuring unpredictability in this way:

1. Differential entropy is a function of the variance of the feature distribution, a larger variance equals higher differential entropy. Therefore, an arbitrary scaling of randomized features will change differential entropy even though security remains unchanged [16].

2. The process of adaptive deterministic quantization converts real-valued features into a discrete feature of uniform distribution (see Figure 2.9). Therefore the differential entropy of the real-valued features is not necessarily
equal to that of the quantized features. Measuring differential entropy prior to quantization reveals little about the unpredictability of the final hash.

Because of this, unpredictability is measured in this dissertation at the final hash level using binary entropy. This can be calculated using a single input $I$ hashed with $S$ different secret-keys $K_k$ where $k = 1, 2, \ldots, S$. If the number of real-valued features is denoted by $N$ and the number of bits assigned to each real-valued feature by the quantization stage is $b$ then the total length of the binary hash ($G_k$) is $N \times b$. Therefore the binary entropy of hashes produced from a fixed input using $S$ different secret-keys is:

$$\text{Entropy} = \sum_{i=1}^{N \times b} -p_i \log_2(p_i) - (1 - p_i) \log_2(1 - p_i)$$ (2.14)

Where $p_i$ is the probability that the $i$-th binary feature bit ($G_k(i)$) is a binary 1. If $G_k(i)$ is the $i$-th bit of the hash produced using the $s$-th secret-key ($K_s$) then these probabilities are obtained using:

$$p_i = \frac{\sum_{k=1}^{S} G_k(i)}{S}$$ (2.15)
2.5 Image Databases

A number of different image databases have been used to evaluate the techniques and theories described in this thesis. The databases used are:

- The uncompressed colour image database (UCID) [42]
- Multimedia information retrieval Flickr (MIRFLICKR) database [43]
- The extended multi modal verification for teleservices and security applications (XM2VTS) face database [44]

The UCID and MIRFLICKR databases contain a large number of images taken of different scenes and objects. While the XM2VTS database contains 2D face images of more than 300 individuals. All three databases are publicly available (links are provided in the bibliography) and have been used extensively by the research community for different image processing and biometric applications.

2.5.1 The UCID database

The UCID database was produced in 2004 as a response to the lack of publicly available large size standardised image databases for the purposes of content based image retrieval. The database consists of 1338 uncompressed colour images in tagged image file format (TIFF). The images contain both indoor and outdoor scenes of different objects (both natural and man-made) at varying distances. Images were captured using a Minolta Dimage 5 digital camera using automatic camera settings. All images have the same dimensions, 512x384 pixels for landscape orientation and 384x512 pixels for portrait. The type of images contained in the UCID database are shown in figure 2.10. The database also contains metadata for each image in the form of keywords relating to image content. This metadata is aimed at facilitating benchmarking of content based image retrieval algorithms. It is unnecessary for robust hashing purposes and is unused in our tests.

2.5.2 The MIRFLICKR database

The MIRFLICKR database was created in 2008 and is another database developed for image retrieval research. The original database contained 25,000 colour
images obtained from the social photography website Flickr. The images are selected for their high 'interestingness' rating, which is a metric produced by Flickr based on the number of user comments, views and times favourited. The MIRFLICKR database is diverse containing a wide range of objects and scenes. In fact, some of the images in the database are not photographs but picture based artwork (see figure 2.11). The images are submitted by different individuals, captured using different hardware and are of different sizes but no larger than 500x500 pixels. The database was recently expanded in 2010 to include 1 million images, but the experiments described in this thesis use only the original 25,000 images.

2.5.3 The XM2VTS database

The XM2VTS database is a 2D face database containing 295 subjects. The database was collected at the University of Surrey in 2003 and was meant as an extension to the original M2VTS database. Each subject was recorded on video uttering a sequence of words and numbers. This was repeated 3 times, for a total of 4 sessions, where each session was separated by a number of months. Unlike the previous two databases which contain a diverse range of images, the XM2VTS contains images that all follow a similar structure.
2.5 Image Databases

Figure 2.11: Example images from the MIRFLICKR database.

Figure 2.12: Example images from the XM2VTS database.

2.5.4 Image Normalisation

All image hashing algorithms apply some form of normalisation prior to robust hashing. Unfortunately there is no standard for image normalisation and different hashing methods have applied different normalisation as part of their algorithm.

In order to allow a fair comparison of hashing methods, the image normalisation process in this thesis is the same for all hashing algorithms. The use of image enhancement techniques such as histogram equalisation and image smoothing are avoided due to their ability to significantly alter image appearance (see Figure 2.13). The essential steps of the normalisation procedure are:

1. Convert image to greyscale.
2. Scale image to 501x501 pixels using bilinear interpolation. Odd dimensions
are used so the image has a definite center point.

Figure 2.13: Histogram equalisation can have undesirable results on the input image.

The result of this normalisation process is shown in Figure 2.14.

Figure 2.14: Example of the image normalisation procedure used in this dissertation.

2.6 Baseline Systems

In this section we describe three baseline robust image hashing algorithms based on popular image hashing techniques. These techniques will be used to illustrate the theories presented in this thesis as well as serve as benchmarks for comparison of new algorithms proposed. The three algorithms chosen are:

- Venkatesan et al.’s DWT features [27].
- Swaminathan et al.’s Fourier Mellin features [4].
- Monga and Mihcak’s NMF based features [3].
2.6 Baseline Systems

The three algorithms are chosen for a number of reasons. Firstly, they are three of the most well known robust image hashing algorithms. Their respective publications are frequently cited in the field of robust image hashing. Secondly, each method uses a different form of feature extraction and the three methods use two different randomization techniques.

2.6.1 DWT Based Features

In 2000 Venkatesan et al. \cite{27} proposed one of the earliest robust image hashing algorithms. The method relied on extracting features from the DWT representation of the input image. Randomization involves dividing the image into random subsections based on the secret-key \( K \). This type of randomization has been shown to be insecure \cite{10} but the method is frequently used as a baseline system in robust image hashing literature \cite{3,4}. The steps in our implementation are:

1. Divide the image in \( N \) sub-images. The location of each sub-image is produced from a uniform PRNG using the secret-key (\( K \)) as the seed.
2. A 4th order Daubechies wavelet is then used to perform a level-3 wavelet decomposition on each of the \( N \) sub-images.
3. The average of each sub-image in the DWT domain is retained to form a length \( N \) real valued randomized feature vector.
4. Quantization and encoding is applied to produce a binary hash vector \( Y \).

![Block diagram of Venkatesan’s image hashing algorithm.](image)

2.6.2 Fourier Mellin Based Features

Swaminathan et al. \cite{4} proposed the well known Fourier-Mellin based image hashing algorithm in 2006. This technique applies a 2D Fourier transform to the
CHAPTER 2. REVIEW OF ROBUST HASHING

input image and obtains features by summing the magnitudes of Fourier coefficients that lie on circles at selected radii away from the zero-frequency component in the 2D frequency space. This technique uses two steps of randomization, the first selects radii based on the secret-key ($K$). This is then followed by a random projection using a set of basis generated from $K$. The steps in our implementation are:

1. Apply the 2D discrete Fourier transform to the image.
2. Sum the magnitudes of Fourier coefficients that lie on circles of radii ($\rho$). The $N$ values of $\rho$ are randomly selected using $K$ and restricted between 0 and 0.4 of the normalized size of the image.
3. A random projection is applied using set of basis generated from $K$.
4. Quantization and encoding is applied to produce a binary hash vector $Y$.

Figure 2.16: Block diagram of Swaminathan’s image hashing algorithm.

2.6.3 NMF Based Features

The most well known matrix factorization based image hashing algorithm is Monga and Mihcak’s non-negative matrix factorization (NMF) method [3]. This approach treats the input image as a 2D matrix, to which a low rank decomposition is applied to extract robust features. Randomization is achieved using a combination of image subdivision and random projections. The steps of the method are:

1. Divide the image into 10 sub-images, each of size 175x175 pixels. The location of each sub-image is randomly chosen using a uniform PRNG with the secret-key ($K$) as the seed.
2. A rank 2 NMF decomposition is applied to each sub-image using the multiplicated update method of NMF by Lee and Seung [45]. This decomposition
2.7 Shortcomings of Existing Robust Hash Functions

produces a set of $175 \times 2$ bases and a set of $2 \times 175$ coefficients for each sub-image.

3. This set of 10 bases and coefficients are appropriately transposed and concatenated to form a new matrix of size $175 \times 40$.

4. A rank 1 NMF decomposition is applied to this new matrix to create an $175 \times 1$ bases vector and a $40 \times 1$ vector of coefficients. These are concatenated to form a single 1D vector of length $175 + 40 = 215$.

5. A random projection is then performed using a matrix of size $215 \times N$ derived from the secret-key $K$ to create a final randomized feature vector of length $N$.

6. Quantization and encoding is applied to produce a binary hash vector $Y$.

Figure 2.17: Block diagram of Monga’s image hashing algorithm.

2.7 Shortcomings of Existing Robust Hash Functions

This thesis attempts to address the shortcomings of existing robust hashing methods. The most common randomization method, the random projection, has been shown to be insecure due to it’s inherent linearity. The use of non-linear randomization techniques that prevent the application of unicity distance based attacks is explored in this work.

Secondly, random projection can be viewed as a fusion of image feature data with image independent secret-key data. The secret-key data therefore does not contribute to hashing accuracy. A method where projection information is both image AND secret-key dependent is developed in this work. This has potential for improving the classification performance of the randomized features.
CHAPTER 2. REVIEW OF ROBUST HASHING

The quantization stage is heavily dependent on feature distribution and most methods model feature distribution using training data. This opens the possibility that feature distributions are overtrained, however most image hashing benchmarks train and test using the same data. This is a best case scenario that would hide the impacts of overtraining. The effects of quantizer training data on hashing performance are explored. Adaptive deterministic quantization also adds training overheads and complications to the process. The development of a method of hashing that does not require quantizer training is attempted in this work. This can be achieved if randomized features are naturally limited and follow a uniform distribution.

The encoding process introduces distance distortion and Zhu et al. showed both Binary code and GrayCode has distance distortion that negatively impact on hashing accuracy [14]. A method that uses features that can be quantized and encoded with less distance distortion is sought to be developed. This would be possible using randomized features that are cyclic in nature, this would mirror the behaviour of GrayCode and reduce distance distortion.
Chapter 3

Effects of Quantizer Training on Hashing Performance

3.1 Introduction

The quantization stage of robust hashing can have significant impact on the performance, but existing research focuses only on the choice of quantization scheme. Work by Zhu et al. [15] showed uniform quantization produces hashes of low fragility. This is because uniform quantization determines thresholds based on the minimum and maximum values obtained from all randomized features $R(i)$ for $i = 1, 2, \ldots, M$. But the pdf of a single real-valued feature component $F(j)$ may not be the same as the pdf of another feature component $F(l)$ or the pdf of all feature components combined. This is especially evident in Fourier-Mellin features which are extracted from the magnitudes of different frequency bands in the Fourier spectrum. It is a well known that the Fourier spectra of typical images have larger magnitudes for lower frequency components compared to those from higher frequencies. Since randomized features ($R(i)$) are produced from $F(i)$ this bias in the feature distributions is also true for randomized features.

To overcome this, uniform quantization has been replaced with adaptive deterministic quantization (described in section 2.3.3), which is trained to produce a set of thresholds for each individual feature component. In order to do this the quantizer requires separate pdfs of each feature component $R(i)$ for $i = 1, 2, \ldots, M$ and these pdfs are estimated from a small sample of training data. Therefore the quantization thresholds are highly dependent on the training data used and the possibility of overtraining or poor generalization becomes a problem. The
CHAPTER 3. EFFECTS OF QUANTIZER TRAINING ON HASHING PERFORMANCE

quantizer training stage is often overlooked in robust hashing literature.

The addition of the quantizer training stage makes the robust hashing evaluation process identical to face recognition evaluation. In face recognition literature, evaluation protocols such as the Lausanne protocol [44] specifically defines the training, evaluation and test phases. The purpose of the training phase is to initialise a non-client specific background model of features and this is the equivalent of quantizer training in robust hashing. The evaluation phase enrolls clients into the system and uses an evaluation data set to determine operating thresholds. A final test phase is then carried out to benchmark system performance. Separate data is used in all three phases and this is important to ensure the system generalizes well and does not overtrain.

Surprisingly this methodology is not followed in the robust hashing field. Many methods evaluate and test hashing performance on the same data used to train the quantizer. This is a naive approach and presents only the best case scenario. In reality, quantizer training would be performed when the system is first deployed using a ‘representative’ dataset of inputs that are expected to be hashed. But what constitutes ‘representative’ may not always be easily definable, particularly for image hashing applications where databases can be large and diverse. Furthermore, randomized features are a function of both the secret-key $K$ and the input data $I$, as such the pdf of randomized features is also sensitive to the choice of $K$.

In this chapter, the effects of quantizer training on hashing performance is explored in greater detail than what exists in current published literature. In the next section it is analytically shown for random projection methods that the choice of secret-key $K$ can alter the pdf of resulting randomized features $R(i)$. These results are experimentally confirmed for secret-keys ($K$) and input types $I$ in sections 3.3 and 3.4 respectively. The investigations are restricted to image hashing algorithms using the baseline methods described in section 2.6. Section 3.5 presents a performance evaluation of the baseline systems under number of different quantizer training scenarios which use different sets of $K$ and $I$ for quantizer training.
3.2 Theoretical analysis of the effects of the secret-key (K) on Randomized feature Distributions

In random projection based methods the randomized feature vector $R$ is obtained from the real-valued features ($F$) and the secret-key dependent projection basis $V$. Recall from section 2.3.2 that $F$ is a vector of length $N$ and $V$ is a $N \times M$ matrix. Each randomized feature component $R(i)$ is the inner product of $F$ with the $i$-th column ($v_i$) in $V$. From (from equation 2.4):

$$R = (\langle F, v_1 \rangle, \langle F, v_2 \rangle, \ldots, \langle F, v_M \rangle)$$  (3.1)

In other words, each component $R(i)$ is given by

$$R(i) = \sum_{j=1}^{N} F(j)v_i(j)$$  (3.2)

Where $F(j)$ is the $j$-th real-valued feature component, $j = 1, 2, \ldots, N$ and $v_i(j)$ is the $j$-th element in the $i$-th column of $V$ and $i = 1, \ldots, M$. By definition of the random projection, the values of $V$ are normally distributed with mean of 0 and variance of 1, therefore the elements of $v_i(j)$ are i.i.d.

It is shown by Monga et al. [3] and Swaminathan et al. [4] that the distribution of randomized features $R(i)$ is approximately Gaussian for their respective methods. Feature distributions can therefore be described using their mean and variance. We begin first with a number of standard definitions.

**Definition 4.** Let $E[X]$ denote the expectation operation applied to the random variable $X$ with a pdf described by $f(x)$. Then $E[X]$ is population (or ensemble) mean defined as

$$E[X] = \int_{-\infty}^{\infty} x.f(x)dx$$  (3.3)

**Definition 5.** Let $\tilde{E}[x]$ denote the expectation operation applied on the vector $x$ of finite length $N$. Then $\tilde{E}[x]$ is the sample (or empirical) mean defined as

$$\tilde{E}[x] = \frac{1}{N} \sum_{j=1}^{N} x_j$$  (3.4)
CHAPTER 3. EFFECTS OF QUANTIZER TRAINING ON HASHING PERFORMANCE

However, if we assume $N = \infty$ then the sample mean ($\bar{E}[x]$) equals the population (or ensemble) mean $E[X]$.

**Definition 6.** Let $\text{Var}[X]$ denote the variance of the random variable $X$. Then $\text{Var}[X]$ is the population variance defined as

$$\text{Var} [X] = E [X^2] - (E [X])^2$$  \hspace{1cm} (3.5)

**Definition 7.** Let $\tilde{\text{Var}}[x]$ denote an estimate of the variance of the random variable $X$ obtained from a finite length realization $x$.

$$\text{Var} [X] = \bar{E} [x^2] - (\bar{E} [x])^2$$  \hspace{1cm} (3.6)

Again, if we assume $N = \infty$ then the variance estimate ($\tilde{\text{Var}}[x]$) equals the population (or ensemble) variance $\text{Var}[X]$.

**Lemma 8.** The mean of randomized feature components $E[R(i)] = 0$ where $i = 1, \ldots, M$ and $j = 1, \ldots, N$ assuming $N = \infty$.

**Proof.** Using equation 3.2

$$E [R(i)] = E \left[ \sum_{j=1}^{N} F(j)v_i(j) \right]$$  \hspace{1cm} (3.7)

The associativity property of the summation and expectation operations means they can be swapped

$$E [R(i)] = \sum_{j=1}^{N} E [F(j)v_i(j)]$$  \hspace{1cm} (3.8)

Because $F(j)$ and $v_i(j)$ are independent the expectation operator can be separated

$$E [R(i)] = \sum_{j=1}^{N} E [F(j)] E [v_i(j)]$$  \hspace{1cm} (3.9)

But $E [v_i(j)] = 0$ so

$$E [R(i)] = \sum_{j=1}^{N} E [F(j)] .0$$

$$= 0$$  \hspace{1cm} (3.10)
Lemma 9. The variance of the randomized feature components \( \text{Var}[R(i)] \) is dependent only on \( F \) and not on the projection basis \( V \). Where \( i = 1, \ldots, M \) and \( j = 1, \ldots, N \) assuming \( N = \infty \).

Proof. From definition 6,

\[
\text{Var}[R(i)] = E[R(i)^2] - (E[R(i)])^2 \\
= E[R(i)^2] \\
= E \left[ \left( \sum_{j=1}^{N} F(j)v_i(j) \right)^2 \right]
\]

(3.11)

But \( v_i(j) \) is uncorrelated with \( F(j) \) by design so

\[
\text{Var}[R(i)] = E \left[ \sum_{j=1}^{N} F(j)^2v_i(j)^2 \right] \\
= \sum_{j=1}^{N} E[F(j)^2] E[v_i(j)^2]
\]

(3.12)

Rearranging the definition of variance for \( E[X^2] \) yields \( E[X^2] = \text{Var}[X] + E[X]^2 \) and equation 3.12 can be rewritten as

\[
\text{Var}[R(i)] = \sum_{j=1}^{N} \left( \text{Var}[F(j)] + E[F(j)]^2 \right) \left( \text{Var}[v_i(j)] + E[v_i(j)]^2 \right)
\]

(3.13)

But since \( \text{Var}[v_i(j)] = 1 \) and \( E[v_i(j)] = 0 \) then

\[
\text{Var}[R(i)] = \sum_{j=1}^{N} \left( \text{Var}[F(j)] + E[F(j)]^2 \right)
\]

(3.14)

Lemma 8 and Lemma 9 show randomize hash features \( R(i) \) are zero mean with variance dependent only on the underlying real-valued feature \( F \). This analysis
is similar to that presented in [33]. From equations 3.10 and 3.14 both $E[R(i)]$ and $Var[R(i)]$ are independent of $i$. This implies that all $M$ randomized features have the same mean and variance and this is a highly desirable trait for robust hashing since it allows a single set of quantization thresholds to be applied to all $M$ randomized features.

However, Lemma 8 and Lemma 9 assume $N = \infty$ and the expectation operator ($E[.]$) refers to the ensemble mean. This is only true theoretically of the population or asymptotically for estimates derived from a large population of samples as the sample size of $v_i(j)$ becomes infinite ($N = \infty$). In other words, the above analysis is valid only in describing the randomized feature distributions produced using all possible secret-keys ($K$) and is only useful as a measurement of sensitivity of randomized hash features to a changing secret-key.

But when the system is considered using the random basis, $\hat{V}$ produced from a single secret-key ($K_1$) then $N$ is not infinite in size (The Fourier Mellin technique has $N = 64$ and NMF based hashing has $N = 215$). The random basis $\hat{V}$ is as a matrix of $M$ columns, where each column is length $N$ and independent of each other. Each column is therefore a small and independent sample of random numbers drawn from Gaussian distribution with zero mean and unit variance. It is a fundamental rule of statistics that the sample mean and sample variance (also known as the empirical mean and variance) of a small sample group is not equal to the corresponding true values for the underlying population (the ensemble mean and variance). We denote the sample mean using $\hat{E}[.]$ and the estimate of the variance as $\hat{Var}[.]$ then $\hat{E}[\hat{v}_i(j)] \neq 0$ and $\hat{Var}[\hat{v}_i(j)] \neq 1$. We now consider the mean and variance of randomized features generated from a single secret-key $\bar{R}(i)$.

**Lemma 10.** Let $\mathbf{\bar{R}}(i)$ be the randomized feature vector produced from a single secret-key. Then the mean of randomized feature components $E[\bar{R}(i)] \neq 0$ where $i = 1, \ldots, M$ and $j = 1, \ldots, N$. Assuming $N << \infty$ then $E[\bar{v}_i(j)] \neq 0$ and is denoted as a constant $c_i$. 

50
3.2 Theoretical analysis of the effects of the secret-key (K) on Randomized feature Distributions

Proof. The expected value of $\bar{R}(i)$ can be obtained by modifying 3.10 to

$$
\bar{E} [\bar{R}(i)] = \sum_{j=1}^{N} \bar{E} [F(j)] \bar{E} [\bar{v}_i(j)] \\
= \sum_{j=1}^{N} c_i \bar{E} [F(j)] \neq 0
$$

(3.15)

Although $c_i$ is small $\bar{E} [\bar{R}(i)]$ can still be large if $\bar{E} [F(j)]$ is large.

\[\square\]

Lemma 11. Let $\bar{R}(i)$ be the randomized feature vector produced from a single secret-key where $N << \infty$ and $\bar{R}(i)$ be the randomized feature vector produced when $N = \infty$. Then $\tilde{\text{Var}}[\bar{R}(i)] < \text{Var}[R(i)]$ where $i = 1, \ldots, M$.

Proof. $\text{Var} [\bar{R}(i)]$ is obtained by modifying equation 3.11

$$
\text{Var} [\bar{R}(i)] = \tilde{E} \left[ (\sum_{j=1}^{N} F(j)\bar{v}_i(j))^2 \right] - \left( \tilde{E} [\bar{R}(i)] \right)^2 \\
= \sum_{j=1}^{N} \left( \text{Var} [F(j)] + \tilde{E} [F(j)]^2 \times \text{Var} [\bar{v}_i(j)] + \tilde{E} [\bar{v}_i(j)]^2 \right) - \left( \tilde{E} [\bar{R}(i)] \right)^2 \\
= \sum_{j=1}^{N} d_i (\text{Var} [F(j)] + \tilde{E} [F(j)]^2) - \left( \tilde{E} [\bar{R}(i)] \right)^2
$$

(3.16)

Where $\text{Var} [\bar{v}_i(j)] \neq 1$ and is denoted with the constant $b_i$ and $(b_i + c_i^2) = d_i$. Equation 3.16 differs from 3.14 due to presence of the constant $d_i$ and the term $- \left( \tilde{E} [\bar{R}(i)] \right)^2$. Since $b_i \approx 0$ and $c_i \approx 1$ then $d_i \approx 1$ and is of no major
consequence. But according to Lemma 10, \( \tilde{E}[\tilde{R}(i)] \neq 0 \) therefore \( \left( \tilde{E}[\tilde{R}(i)] \right)^2 > 0 \). Consequently \( \text{Var}[\tilde{R}(i)] < \text{Var}[R(i)] \).

Lemma 10 and Lemma 11 show the mean and variance of randomized hash features \( \tilde{R}(i) \) is a function of \( c_i \) and \( b_i \) which in turn are functions of the secret-key \( K \). Equations 3.15 and 3.16 show both the mean and variance to be dependent on \( E[F(j)] \) and the sensitivity to the choice of \( K \) is further compounded if the hashing algorithm produces a different \( F \) given different \( K \). This is indeed the case for both the Fourier-Mellin technique and the NMF technique. In the Fourier Mellin method the secret-key affects \( \rho \) which is the radii along which features are summed in the frequency spectrum. The NMF method displays similar behaviour, where the secret-key determines the locations of each image sub-block.

The sensitivity of randomized feature distributions to both the secret-key \( K \) and to \( F \) creates a potential need to train, re-train and store quantizer training data for all secret-keys, input types or a combination of the two! The use of adaptive deterministic quantization therefore introduces the need for a key-management framework that can add considerable overheads to the hashing process. Even more alarming is the potential information leakage and security loop-holes introduced by storing quantizer threshold data for each \( K \) used in the system.

### 3.3 Effects of Secret-keys on Feature Component pdfs

In the previous section it was theoretically shown that the pdf of Randomized features are a function of \( K \). This section experimentally confirms these results using the three baseline implementations described in Chapter 2. These three methods are representative of different image feature extractors and different randomization methods. The NMF and Fourier Mellin methods rely on random projection whilst the DWT method does not. The length of the randomized feature vector is set to sixty-four \( (M = 64) \) for all three methods.

In order to explore the effects of \( K \) on the distribution of \( R(i) \) the set of input images is kept constant but three different secret-key scenarios are used. In all scenarios the first 1000 images of the MIRFLICKR database are used. The first scenario uses 1000 different secret-keys \( (K_{all}) \) so that each image is hashed with
3.4 Effects of Image Data on Feature Component pdfs

a unique key. The two remaining scenarios use a single but different secret-key ($K_1$ and $K_2$) to hash all images.

The results of these experiments are presented in Figures 3.1-3.3. Both the histogram and cumulative histograms are provided for the 1st, 24th, 48th and 64th feature components ($R(1), R(24), R(48)$ and $R(64)$). The cumulative histogram is included because adaptive deterministic quantization places thresholds at the appropriate percentile locations. For example a 2-bit quantizer splits the feature into four equal probability levels by placing thresholds at the lower quartile, median and upper quartile which are the 25th, 50th and 75th percentile, respectively. The cumulative histogram clearly shows the difference in threshold locations that would result from the use of different secret-keys if an Adaptive Deterministic Quantizer were to be applied.

The results shown in figures 3.2 and 3.3 indicate that both Fourier Mellin and NMF features are highly sensitive to secret-key changes and confirms the theoretic analysis of the previous section. For both methods the PDF for 'All Keys' is zero mean and the variance remains consistent for different $i$ and this experimentally confirms Lemma 8 and Lemma 9. The PDF for 'Key 1' and 'Key 2' have non-zero means with variances less than those obtained from 'All Keys' which is consistent with Lemma 10 and Lemma 11. The distribution of hash features for the DWT method is relatively invariant to the secret-key. This is due to the absence of random projection in the DWT method which relies only on random blocking.

3.4 Effects of Image Data on Feature Component pdfs

In the second set of experiments a constant secret-key ($K_1$) is used to produce feature component pdfs from three different image databases. The three databases are MIRFLICKR database (first 1000 images), UCID database of 1320 images and XM2VTS database of faces. The MIRFLICKR database and the UCID database contain diverse images of people, scenery and objects. While the XM2VTS database contains only close-up face images from 145 different individuals. The histograms and cumulative histograms for the 1st, 24th, 48th and 64th feature components are shown in Figures 3.4-3.6.

The feature component pdfs of all three methods show sensitivity to image
CHAPTER 3. EFFECTS OF QUANTIZER TRAINING ON HASHING PERFORMANCE

Figure 3.1: Probability density functions of the 1st, 24th, 48th and 64th randomized features generated using DWT method on the MIRFLICKR image set for different secret-keys.
3.4 Effects of Image Data on Feature Component pdfs

Figure 3.2: Probability density functions of the 1st, 24th, 48th and 64th randomized features generated using Fourier Mellin method on the MIRFLICKR image set for different secret-keys.
CHAPTER 3. EFFECTS OF QUANTIZER TRAINING ON HASHING PERFORMANCE

Figure 3.3: Probability density functions of the 1st, 24th, 48th and 64th randomized features generated using NMF method on the MIRFLICKR image set for different secret-keys.

Figure 3.3: Probability density functions of the 1st, 24th, 48th and 64th randomized features generated using NMF method on the MIRFLICKR image set for different secret-keys.
3.4 Effects of Image Data on Feature Component pdfs

Figure 3.4: Probability density functions of the 1st, 24th, 48th and 64th randomized features generated using DWT method and three different image sets (MIRFLICKR, UCID and XM2VTS) using same secret-key.
CHAPTER 3. EFFECTS OF QUANTIZER TRAINING ON HASHING PERFORMANCE

Figure 3.5: Probability density functions of the 1st, 24th, 48th and 64th randomized features generated using Fourier Mellin method and three different image sets (MIRFLICKR, UCID and XM2VTS) using same secret-key.

58
3.4 Effects of Image Data on Feature Component pdfs

Figure 3.6: Probability density functions of the 1st, 24th, 48th and 64th randomized features generated using the NMF method and three different image sets (MIRFLICKR, UCID and XM2VTS) using same secret-key.
data. The differences between the MIRFLICKR and UCID databases are minor but the XM2VTS database produces a significantly different feature component pdf in all three methods. This is because the images in the XM2VTS database share a common underlying structure. It is well known in face recognition that the visually discriminating facial information does not reside in the lower frequency bands \[46\]. However, both Fourier Mellin and DWT are frequency decomposition techniques and image hashing features are taken from the lower frequencies due to their invariance to content preserving modifications.

The NMF method uses only low-rank decompositions and also retains only the lowest frequency information. The sub-block extraction in the NMF and DWT methods further compounds this because many regions of the face are uniform regions of skin lacking any discriminating features. As a result the variance of features from the XM2VTS database are significantly lower than those derived from the UCID or MIRFLICKR databases. Although the XM2VTS database contain only face images, these results could extend to any set of images that share an underlying structure. If the visually discriminating information of a set of images is restricted to high frequency bands or limited to only certain spatial regions, which faces are a good example of, then the extracted image features may be insensitive to this information and follow a distribution of similarly low variance.

These experimental results indicate that estimated quantization thresholds by adaptive deterministic quantization are sensitive and can be very different depending on the input data used. This is a problem for quantization based image hashing because the set of all possible input images and all possible secret-keys are infinite in size. It is therefore impossible to train a quantizer for all possible images and secret-keys. Furthermore, even if the quantizer is trained using a very large set of data, the feature component pdf for a single secret-key or a subset of images may be very different then that of the entire set. This exposes a weakness of image hashing evaluation methodology, unlike other fields of pattern classification, it is not common practice in image hashing to separate training and testing data. As a result the existing analysis of image hashing fragility, entropy and false-alarm/miss performance show an upper limit that may not be reached in a practical system.
3.5 Performance Evaluation under Different Quantizer Training Scenarios

The previous sections have shown that both secret-keys and input data can have significant impacts on randomized feature distributions for all three of the baseline systems. As a result, quantization thresholds produced from one set of data may be unsuited for quantization of unseen data. It is the purpose of this section to identify the impacts on hashing performance for different quantization training scenarios.

The five different quantization scenarios are:

- **Scenario 1** - Uses first 1000 images (based on filename order) of the MIRFLICKR dataset and secret-key \( K_1 = 1337 \).

- **Scenario 2** - Uses the UCID image set and secret-key \( K_1 \).

- **Scenario 3** - Uses the XM2VTS image set and secret-key \( K_1 \).

- **Scenario 4** - Uses the MIRFLICKR image set but a different secret-key \( K_2 = 80087355 \).

- **Scenario 5** - Uses the UCID image set but uses a different secret-key for each image. The secret-keys are integer numbers uniformly distributed between 0 and 1,000,000 generated from a Mersenne twister PRNG with a seed value of 0.

It is important to point out that the performance evaluation is carried out using the same set of data for all quantizer training scenarios. This evaluation data set is the same as the scenario 1 quantizer training data (MIRFLICKR image set and secret-key \( K_1 \)). Therefore, scenario 1 trains the quantizer using the same image set and secret-key that is used in the performance evaluation and is representative of the best-case scenario prominent in existing literature. Scenarios 2 and 3 use the same secret-key but different image sets of increasing variation to the image set used in quantizer training. Scenario 4 uses a different secret-key but the same image set. Scenario 5 uses a different set of images and a wide range of secret-keys.

The scenarios are chosen in order to isolate the affects of varying secret-key and varying image data on quantizer training. Scenarios 2 and 3 explore the...
CHAPTER 3. EFFECTS OF QUANTIZER TRAINING ON HASHING PERFORMANCE

affects of varying image data. Scenario 2 uses the UCID database which is similar in nature to the MIRFLICKR database both in terms of number of images and range of content. On the other hand, scenario 3 uses a relatively small subset of the XM2VTS database of face images, this shows the behaviour of hashing system when given a set of structurally similar images (either a subset of related images from the original training set or an unseen dataset from a different source or application). Scenario 4 uses the identical image dataset but alters only the choice of secret-key. Scenario 5 is designed to be a realistic practical training set, where the quantizer is trained on a diverse set of images using a number of different secret-keys. In this scenario there is no overlap between quantizer training data and evaluation data.

In each scenario, adaptive deterministic quantization is applied to different image data and/or secret-keys in order to learn the quantization thresholds that are then applied to evaluation data. The evaluation data contains the first 1000 images of the MIRFLICKR database and a fixed secret-key ($K_1$).

3.5.1 Fragility of Hashes

The fragility curves in figure 3.7-3.9 shows the expected discriminability of hashes measured using equation 2.11. Scenario 1 reflects the methodology adopted in the existing literature and the expected discriminability measured is equal to the maximum fragility of 0.5. This reflects the results obtained by Zhu et al. in [14] and is the best case scenario.

The expected discriminability for scenario 2 drops only slightly to around 0.48 for all three algorithms. This is close to the theoretical maximum and is not surprising considering scenario 2 uses the same secret-key and a similar image dataset for both quantizer training and performance evaluation. However, in scenario 3 the expected discriminability drops sharply to around 0.25 for all algorithms. This scenario uses only face images for quantizer training and such a dataset would produce a tightly clustered feature pdf in all three algorithms. This is because the discriminating information in face images reside in the medium frequency bands and this information is lost when using low-frequency based robust image features. Although face images have been used in this experiment similar results would be true for any subset of images that share a similar underlying structure. This result highlights the importance of appropriate image selection for quantizer training but also indicates that low-frequency based hashing methods are suitable
only for diverse image databases.

In both scenario 4 and 5 the expected discrimability for the NMF and Fourier-Mellin methods drop substantially to less than 0.1. This is a severe drop and can be directly attributed to the use of random projection in the randomization stage of robust hashing. This sharp decline is not seen in the DWT scheme where random projection is not used. This result confirms the theoretic analysis of section 3.3 random projection produces randomized features that have distributions highly sensitive to the secret-key used. This is a major flaw in random projection based robust hashing and raises concerns about how quantizer training should be carried out in a practical application. The low expected discrimability for scenario 4 indicates the trained quantizer thresholds obtained for a given secret-key should not be applied to quantize hash features produced with a different secret-key. Similarly, scenario 5 where training is carried out with multiple secret-keys fares only slightly better. This is not a problem for applications of robust hashing where no (or only one) secret-key is required. But for applications that require multiple secret-keys or user-dependent secret-keys then quantizer training would need to be performed for every secret-key in the system.

Furthermore, these results indicate that fragility is affected not only by choice of quantization scheme but more so by quantization training. Each of the three algorithms, using the same evaluation data, produced a wide range of possible expected discrimabilities when quantizer training data is altered. All three hashing algorithms produced the maximum expected discrimability of 0.5 for the best-case scenario (scenario 1). Therefore expected discriminability is not a deterministic measure of fragility, nor is it a good metric for comparing robust hashing schemes. It is instead only a measure of the goodness-of-fit of the feature distribution pdf used in quantizer training to the feature distribution of the evaluation data. The maximum fragility of 0.5 which corresponds to optimal hashing performance [14] can be achieved by any system using any quantization scheme as long as quantization training data (both images and secret-key) perfectly matches the evaluation data. Even a high fragility would be difficult to obtain in a practical situation, especially for random projection based methods which require a good estimate of images to be hashed and also prior knowledge of each and every secret-key. Even if this were possible, the need to store different quantizer training data for each secret-key introduces a number of security flaws. These are discussed in section 3.6.
CHAPTER 3. EFFECTS OF QUANTIZER TRAINING ON HASHING PERFORMANCE

Figure 3.7: Fragility curves and expected discriminability (ED) of DWT scheme for different quantizer training scenarios using 1, 2, 3 and 4 bit quantization. Scenario 3 uses the XM2VTS data set of only facial images and presents a significantly different fragility curve compared to the others. The DWT method is sensitive to input data but not to the key.
3.5 Performance Evaluation under Different Quantizer Training Scenarios

Figure 3.8: Fragility curves and expected discriminability (ED) of NMF hash for different quantizer training scenarios using 1, 2, 3 and 4 bit quantization. The NMF method is sensitive to the key. It is not sensitive to differences between UCID and MIRFLICKR images but when all images are faces as in XM2VTS there is a difference as in scenario 3.
CHAPTER 3. EFFECTS OF QUANTIZER TRAINING ON HASHING PERFORMANCE

Figure 3.9: Fragility curves and expected discriminability (ED) of Fourier Mellin scheme for different quantizer training scenarios using 1, 2, 3 and 4 bit quantization. The sensitivity here is similar to that of the NMF method.
3.5.2 Hash Entropy

Entropy and fragility are directly proportional, therefore quantization scenarios that produce low fragility will produce binary hashes of low entropy. The entropies obtained using equation 2.14 are presented in table 3.1. We would like to point out that these entropies are for a set of hashes $G_i$ obtained from different images $I_i$ given a fixed secret key $K$ and not the entropy discussed in earlier literature which is for a set of hashes $G_i$ obtained from a fixed image $I$ but different secret-keys $K_i$.

A hashing algorithm that produces low entropy hashes is susceptible to attack. One such attack allows the attacker to correctly estimate the hashes of unknown images when the secret-key is known. By estimating the hash closely enough it is possible for an attacker to pass a forged copy as an authentic one. Low entropy bits in the hash are those that tend to be the same even for different images. If an attacker has knowledge of these low entropy bits than a brute force attack becomes feasible. Such an attack can be carried out by generating binary hashes for a large set of images using the known secret-key and trained quantization thresholds. Low entropy bits can then be identified from this set of binary hashes.

Table 3.1 shows only scenario 1 achieves the maximum possible entropy and scenario 2 has close to the maximum. However, scenario 4 has much lower entropies for the NMF and Fourier Mellin schemes with entropies of 5.79 and 10.85 respectively. The entropies in Table 3.1 are directly related to the expected discriminabilities shown in figures 3.7–3.9. Low fragility correspond to low entropy and vice versa. Since entropy and fragility are directly related, the cause of low entropies is the same as the causes of low fragility discussed in the previous subsection. The DWT scheme, which does not use random projection displays high entropy for all but scenario 3, whilst both NMF and Fourier-Mellin have very low entropy for scenarios 4 and 5. Entropy, like fragility, is therefore highly sensitive to the choice of secret-key used in quantizer training for random projection based methods.

Such low entropies make it very easy for an attacker to produce an acceptable hash, especially when considering the acceptance threshold $\tau$ (such that a genuine hash is accepted if $d_H(G_1, G_2) < t$), is ususally greater than zero in order to satisfy robustness. For the NMF scheme scenario 4 produces hashes of only 5.79 bits of entropy. This is the equivalent of only $2^{5.79} = 55$ unique hashes, in such a
system finding collisions and estimating hashes become trivial.

Table 3.1: Entropy of binary hashes for different quantizer training scenarios measured in bits. Note that the maximum entropy for the 1, 2, 3 and 4-bit quantizers are 64, 128, 192 and 256 bits respectively.

<table>
<thead>
<tr>
<th>Quantizer Training Scenario</th>
<th>1-bit</th>
<th>2-bit</th>
<th>3-bit</th>
<th>4-bit</th>
<th>5-bit</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fourier Mellin Scheme</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1-bit</td>
<td>63.99</td>
<td>62.37</td>
<td>35.42</td>
<td>10.85</td>
<td>17.36</td>
</tr>
<tr>
<td>2-bit</td>
<td>127.99</td>
<td>124.56</td>
<td>63.15</td>
<td>24.66</td>
<td>49.99</td>
</tr>
<tr>
<td>3-bit</td>
<td>191.99</td>
<td>187.73</td>
<td>94.77</td>
<td>42.32</td>
<td>100.42</td>
</tr>
<tr>
<td>4-bit</td>
<td>255.99</td>
<td>251.41</td>
<td>129.50</td>
<td>63.03</td>
<td>162.14</td>
</tr>
<tr>
<td>NMF Scheme</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1-bit</td>
<td>63.99</td>
<td>62.64</td>
<td>50.51</td>
<td>5.79</td>
<td>11.89</td>
</tr>
<tr>
<td>2-bit</td>
<td>127.99</td>
<td>125.43</td>
<td>83.89</td>
<td>11.24</td>
<td>36.51</td>
</tr>
<tr>
<td>3-bit</td>
<td>191.99</td>
<td>189.04</td>
<td>122.03</td>
<td>18.21</td>
<td>70.50</td>
</tr>
<tr>
<td>4-bit</td>
<td>255.99</td>
<td>252.83</td>
<td>163.92</td>
<td>26.49</td>
<td>116.25</td>
</tr>
<tr>
<td>DWT Scheme</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1-bit</td>
<td>63.99</td>
<td>62.67</td>
<td>40.24</td>
<td>62.76</td>
<td>63.26</td>
</tr>
<tr>
<td>2-bit</td>
<td>127.99</td>
<td>124.23</td>
<td>67.49</td>
<td>124.48</td>
<td>125.23</td>
</tr>
<tr>
<td>3-bit</td>
<td>191.99</td>
<td>187.78</td>
<td>101.30</td>
<td>187.87</td>
<td>188.49</td>
</tr>
<tr>
<td>4-bit</td>
<td>255.99</td>
<td>251.33</td>
<td>140.05</td>
<td>251.59</td>
<td>252.12</td>
</tr>
</tbody>
</table>

3.5.3 Hashing Accuracy

Zhu showed the choice of quantization scheme can effect the trade-off between false alarm and miss. The ROC curves in Fig. 3.10-3.11 show quantization training can have even larger effects on performance. They are obtained using a set of modified images generated from the 1000 original images. For each original image, five different content preserving modifications (denoted as $k = 1, \ldots, 5$) are applied to create a set of $1000 \times 5$ modified images. The modifications are JPEG compression at 50% quality, Gaussian noise of 0.01 variance, mean filtering using $3 \times 3$ kernel, cropping by 5 pixels and histogram equalization.

ROC curves are drawn using the false alarm and miss probabilities obtained by classifying the modified images using the normalized Hamming distance (see Table 2.1) where the threshold $\tau$ is varied between the two extremes of 0 (total mismatch) and 1 (perfect match). Miss probability is measured by comparing the hashes of each modified images to the hash of the corresponding original image, a miss is when

$$d_H(Y_{i,k}^{\text{modified}}, Y_i^{\text{original}}) > \tau \text{ where } i = 1, \ldots, 1000 \text{ and } k = 1, \ldots, 5 \quad (3.17)$$
3.5 Performance Evaluation under Different Quantizer Training Scenarios

Figure 3.10: ROC curves for DWT scheme for different quantizer training scenarios using 1, 2, 3 and 4 bit quantization.

False alarms are measured by comparing each modified image hash with the hashes generated from the 999 other original images, a false alarm is when

$$d_H(Y_{i,k}^{\text{modified}}, Y_j^{\text{original}}) > \tau$$  where $i, j = 1, \ldots, 1000$ but $i \neq j$ and $k = 1, \ldots, 5$  

(3.18)

Existing image hashing literature often produce ROC curves for each different types of modification, we instead combine all the modified images into a single test set as was done in [15]. This is a better representation of real-world performance where there is no prior knowledge about the modifications in a given image, thus the threshold $\tau$ of the hashing system must provide accurate classification for all types of image modifications. ROC curves are presented for each of the three baseline systems using 1, 2, 3 and 4 bit quantization ($b = 1, 2, 3, 4$).
CHAPTER 3. EFFECTS OF QUANTIZER TRAINING ON HASHING PERFORMANCE

Figure 3.11: ROC curves for Fourier Mellin scheme for different quantizer training scenarios using 1, 2, 3 and 4 bit quantization.

...
3.5 Performance Evaluation under Different Quantizer Training Scenarios

Figure 3.12: ROC curves for NMF scheme for different quantizer training scenarios using 1, 2, 3 and 4 bit quantization.

Figure 3.12: ROC curves for NMF scheme for different quantizer training scenarios using 1, 2, 3 and 4 bit quantization.
CHAPTER 3. EFFECTS OF QUANTIZER TRAINING ON HASHING PERFORMANCE

Scenario 1 generally produces the best results in all three hashing schemes. Although for the DWT scheme the performance is quite similar for scenarios 1, 2, 4 and 5. This is because the feature distributions for the DWT scheme remains mostly the same for scenarios 1, 2, 4 and 5 as was shown in sections 3.3 and 3.4. In scenario 1, all three methods display the worst EER for a 1-bit quantizer. This improves slightly when a 2-bit quantizer is used, but surprisingly begins to degrade for the 3 and 4-bit quantizers. This seems like an strange result, one would expect improved performance as the number of quantization levels increases due to increased feature space resolution and hence better discriminat- ing power. This surprising behaviour can be attributed to distance distortion which degrades EER performance [14]. As discussed in Chapter 2, the amount of distance distortion is directly proportional to the number of quantization levels, so the larger \( b \) the larger the distortion suffered. The ROC curves indicate the optimal \( b \) is 2 for Gray encoding, any higher and the performance degradation introduced by distance distortion outweights the performance gains of increased feature space resolution. Distance distortion and its effects on hashing accuracy is further analysed in Chapter 4.

The EER performance of scenario 2 is very similar but slightly worse than those obtained in scenario 1 for all three hashing schemes. This is expected since the UCID and MIRFLICKR databases produce very similar feature distributions in all three schemes.

Scenario 3 exhibits consistent behaviour for all three hashing schemes, where performance is the best for a 1-bit quantizer but degrades as \( b \) increases. This can be understood by observing the feature distributions in figures 3.4-3.6. In all three schemes, the XM2VTS database produces a tightly concetrated pdf of low variance. Whilst the MIRFLICKR database produces features with a much larger range and is of relatively high variance. Scenario 3 trains with the XM2VTS database but is evaluated with the MIRFLICKR database. This is a case where thresholds learned from a low variance pdf is applied to quantize features from a high variance pdf. From figures 3.4-3.6 we can see that the XM2VTS feature distribution has a range much smaller than the MIRFLICKR feature distribution. Thus when the quantization thresholds learnt from the XM2VTS pdf is applied to the MIRFLICKR pdf it has the effect of dividing the MIRFLICKR pdf into two areas. One area, which does not overlap with the XM2VTS pdf will be devoid of any thresholds whilst the other area, which overlaps with the XM2VTS
3.5 Performance Evaluation under Different Quantizer Training Scenarios

pdf, will contain all of the thresholds. The first area is under-quantized and no discrimination occurs for features that fall within the area. The second area is potentially over-quantized and can be too sensitive to even small differences in feature values. It is easy to see when $b$ increases so does the amount of over-quantization and this increasing sensitivity degrades EER performance. This behaviour is illustrated in figure 3.13.

Figure 3.13: A 2-bit adaptive deterministic quantizer is trained with a low variance pdf (red) and the thresholds (blue) are applied to quantize a high variance pdf (green). The learned quantization thresholds will produce a poor partitioning of the high variance pdf. Resulting in an under-quantized area with no thresholds ($x > -3$) and an over-quantized area containing all of the thresholds ($-5 \leq x \leq -3$).

In scenario 4 the quantizer is trained with secret-key $K_2$ but evaluated with secret-key $K_1$. From figure 3.1 we see that the pdf for both secret-keys are very similar in the DWT scheme and the EER performance behaves similarly to scenario 1. However from figure 3.2 and 3.3 the feature pdfs obtained using two different secret-keys can be very different. The pdf for $K_1$ and $K_2$ are both low variance and display different amounts of overlap for different feature compo-
CHAPTER 3. EFFECTS OF QUANTIZER TRAINING ON HASHING PERFORMANCE

...ntents. When the two pdfs do not overlap, thresholds learned from the pdf of \( \mathbf{K}_2 \) may not partition the pdf of \( \mathbf{K}_1 \) at all. In this case altering \( b \) will have no effect on performance. When the two pdfs do overlap, then increasing \( b \) will also increase the amount of partitioning of the pdf of \( \mathbf{K}_1 \). Over-quantization will not occur since the two pdfs have similar variance and good quantization can still be achieved if the two pdfs have significant overlap. An example of this is shown below in figure 3.14.

![PDF of an example feature component](image)

(a) No overlap

![PDF of an example feature component](image)

(b) Significant overlap

Figure 3.14: A 2-bit adaptive deterministic quantizer is trained with a low variance pdf (red) and the thresholds (blue) are applied to quantize another low variance pdf (green). When the two pdfs do not overlap, the resulting quantization and hence EER performance will be poor regardless of \( b \). However, if the two pdfs significantly overlap then good quantization is still achievable.

The NMF and Fourier-Mellin methods produce the worst results for a 1-bit quantizer in Scenario 5. However, performance improves quickly as \( b \) increases and when \( b = 4 \) the scenario 5 EER is better than the scenario 1 EER! This seems like an impossible result but similar behaviour is presented in [15] where the theoretically inferior uniform quantizer is shown to equal or even outperform the superior adaptive deterministic quantizer when \( b \) is sufficiently high. Recall that scenario 5 trains the quantizer using the feature pdf generated from multiple secret-keys (\( \mathbf{K}_{all} \)) but the evaluation set uses only a single secret-key (\( \mathbf{K}_1 \)). Figure 3.5 and 3.6 show the feature distribution for \( \mathbf{K}_{all} \) has a much larger variance than the distribution for a single key \( \mathbf{K}_1 \). Scenario 5 is therefore a case where thresholds learned from a high variance pdf is applied to quantize features from a low variance pdf. When \( b = 1 \) only a single threshold exists and there is a high chance that the feature pdf of \( \mathbf{K}_1 \) lies entirely (or almost entirely) on one side of...
3.5 Performance Evaluation under Different Quantizer Training Scenarios

the threshold. This results in no discriminating information being contributed by that feature component and hence the low EER performance. As $b$ increases, so does the number of quantization thresholds and this makes it less likely that a feature distribution falls entirely with a single (or low) number of quantization levels. This is why the entropy (as a percentage of the maximum) and EER performance for scenario 5 improve as the number of bits increase.

![PDF of an example feature component](image1.png) ![PDF of an example feature component](image2.png)

(a) 1-bit quantizer  (b) 4-bit quantizer

Figure 3.15: A 1-bit (a) and 4-bit (b) adaptive deterministic quantizer is trained with a high variance pdf (red) and the thresholds (blue) are applied to quantize a low variance pdf (green). A 1-bit quantizer may not partition the low variance pdf at all and EER performance will be poor. When $b$ is increased to 4 a reasonable quantization of the low variance pdf can be achieved.

To understand why scenario 5 can equal or outperform scenario 1. Recall for a $b$-bit quantizer there are $L = 2^b$ quantization levels produced using $L - 1$ thresholds. In scenario 5 these thresholds are learned from the pdf of $K_{all}$ but applied to quantize the features generated from $K_1$. The variance of the pdf of $K_1$ is less than the pdf of $K_{all}$, causing the pdf of $K_1$ to be partitioned in to less than $L$ quantization levels. In other words, the pdf of $K_1$ is being quantized with less than $b$ bits. This reduces the amount of distance distortion suffered and in some cases this reduction in distance distortion allows scenario 5 to outperform scenario 1. An excellent example can be seen in figure 3.11 where the scenario 5 EER for a 4-bit quantizer is equal to the scenario 1 EER for a 3-bit quantizer.

Based on these results some insights can be obtained regarding quantizer training for robust hashing. Whilst scenario 1 is indeed the best-case scenario for all schemes, it is not always possible to have prior knowledge regarding which images will be hashed using which secret-keys. In general, the DWT scheme
which does not use random projection is less sensitive to quantizer training. The random projection based methods are more sensitive, particularly to the choice of secret-key. This sensitivity can be somewhat overcome by training the quantizer using a large range of secret-keys and using a $b > 2$. In this scenario, EER performance can approach those in scenario 1, however the entropy of hashes is still relatively low (see Table 3.1).

3.6 Discussion on Security

The results presented in this chapter highlight the significant effects of quantizer training on hashing performance, reductions in hash fragility, entropy and accuracy are suffered when inappropriate quantization thresholds are applied. The sensitivity of feature distributions to the secret-key in methods utilising random projections also highlights a potential need for secret-key specific quantization training. However, training the quantizer and storing the resulting quantization thresholds for every secret-key in the system has significant implications on security.

The storage of quantization thresholds makes it possible for attackers and rogue administrators to tamper with quantization thresholds. In fact the experiments conducted in this chapter can be viewed as an implementation of an attack where an adversary has replaced the original quantizer thresholds with another set obtained using different training data. The evaluation results for each quantization scenario depicts hashing performance for a system attacked in this way. In some scenarios the entropy of hashes can drop to as low as 5-bits, meaning the systems ability to produce secure hashes is almost non-existent.

The security of the system can be completely circumvented if an attacker replaced quantizer thresholds with arbitrarily selected ones. Such an attack can be used to violate the collision resistance property of hashing. Consider a binary hash produced from a 1-bit quantizer $H = [0, 1, 0, 1]$ using an unknown image $I$ with an unknown secret-key $K$. Collision resistance states that it should not be possible to find another $I$ and $K$ that produces the same hash. However, a 1-bit quantizer has a single threshold for each real-valued feature and by setting the quantization thresholds to $[\infty, -\infty, \infty, -\infty]$ then any input using any secret-key will produce the hash $[0, 1, 0, 1]$. Conversely the thresholds can be changed to $[-\infty, \infty, -\infty, \infty]$ and no input (not even the correct one) can produce the
desired hash. It is obvious to see that these two attacks can render a hashing system useless.

Quantization training data cannot be hashed because the actual threshold values are required, they can be encrypted but encryption does not guarantee the prevention of unauthorised access and this circumvents one of the principal advantages of hashing. For example, the primary advantage of biometrics is their superiority to passwords or PINs (non-repudiation, difficult to share or guess). But if Quantization-based robust hashing is employed then the resulting Quantization training data must be protected with a password. The security of the system comes down to the security of a password and the advantages of using biometrics is negated.

3.7 Chapter Summary

In this chapter a thorough analysis on the effects of secret-key $K$ and input data $I$ on randomized feature distributions has been conducted. It has been shown both theoretically and experimentally that feature distributions can be highly sensitive to both $K$ and $I$ especially for methods relying on random projection. This sensitivity is problematic for adaptive deterministic quantization because a mismatch between quantizer training data and evaluation data has been shown to cause significant reductions in fragility, entropy and accuracy of the resulting hashes.

These findings indicate that the improved hashing performance achieved by adaptive deterministic quantization over uniform quantization are only obtained in the trivial case when quantization data perfectly matches evaluation data. The implications of this for many existing hashing algorithms is that the quantizer needs to be trained separately for every secret-key $K$ in order to achieve the hashing performance presented in literature. This however is not always possible for systems with large or frequently updated secret-keys, nor is it a desirable solution since it introduces a need for training-data management. Even if such training-data management is possible, there are security attacks that can be directed at quantization training data and this severely limits the benefits of random projection based robust hashing.
Chapter 4

Robust Hashing using Angle Features

4.1 Introduction

The previous chapter showed the pdf of randomized features obtained using both random projections and random blocking based hashing display sensitivity to input images and secret-keys. The unbounded nature of randomized features coupled with the varying pdf poses a number of problems for adaptive deterministic and uniform quantization. So far, there has been little research directed at producing randomized features that display less sensitivity to quantizer training. Such features could be discretized without the need for adaptive deterministic quantization.

The method proposed by Khelifi et al. [32] is the only method that does not involve quantization training, it instead relies on a priori knowledge regarding the pdf of real-valued features. It is therefore not a general solution and the hashing process cannot be applied to features produced using a different feature extractor.

A more general solution to the problem lies at the randomization stage. If the randomization process produces bounded outputs then uniform quantization can be applied without quantizer training and a single set of quantization thresholds could be used for all inputs and/or secret-keys. One type of feature that satisfies this criterion are angle features. Angles are naturally bounded between 0 and $2\pi$. In addition, angles are cyclic (or modulo) in nature and the upper and lower limit (0 and $2\pi$) are in fact neighbouring in terms of distance. This makes angle features more suited for Gray encoding because Gray code also exhibits this cyclic
CHAPTER 4. ROBUST HASHING USING ANGLE FEATURES

behaviour.

This chapter proposes a modification to the random projection method that allows the generation of angle features. Although the method is similar to the BioPhasor scheme presented in [19] the primary purpose of the chapter is to analyze the effects of angle features on robust hashing quantization and encoding. In section [4.4] the quantization benefits of angle features are experimentally tested. While in section [4.3] it is theoretically proven that angle features suffer less distortion than real-valued features when encoded with Gray code and this is experimentally verified for robust image hashing using the MIRFLICKR database of images.

4.2 Proposed Method

The random projection method described in Chapter 2 produces randomized features by applying matrix multiplication of the feature vector $F$ with the secret-key dependent projection basis $V$ to obtain the real-valued randomized feature vector $R$. From equation [2.3]

$$R = F \times V$$  (4.1)

A simple method of producing angle features is to produce a complex-valued randomized feature vector which can be converted to polar form and expressed as a magnitude and angle component. For the complex number $z = a + bi$ the magnitude is

$$|z| = \sqrt{a^2 + b^2}$$  (4.2)

and the argument ($\phi$) is

$$\phi = \arctan2\left(\frac{b}{a}\right) = \begin{cases} 
\arctan\left(\frac{b}{a}\right) & \text{if } a > 0 \\
\arctan\left(\frac{b}{a}\right) + \pi & \text{if } a < 0 \text{ and } b \geq 0 \\
\arctan\left(\frac{b}{a}\right) - \pi & \text{if } a < 0 \text{ and } b < 0 
\end{cases}$$  (4.3)

Using the above equation, $\phi$ is bounded in the interval $(-\pi, \pi]$ and is usually shifted to the range $(0, 2\pi]$ by adding $2\pi$ to only the negative values. Although for purposes of randomized feature extraction it is valid to shift the range by simply adding $\pi$ to all $\phi$.  

80
4.2 Proposed Method

Producing a complex-valued $R$ is easily achieved by projecting with a complex-valued $V$. By using a complex projection basis $V = V^R + V^3i$ equation (4.1) becomes:

$$R = F \times (V^R + V^3i) = (F \times V^R) + (F \times V^3)i$$  \hspace{1cm} (4.4)

If the $j$-th column of $V$ is denoted as $v_j$ and $\langle \rangle$ denotes the inner product then each feature component $R(j)$ is given by

$$R(j) = \phi_j = atan2 \left( \frac{\langle F, v_j^3 \rangle}{\langle F, v_j^R \rangle} \right) \text{ where } j = 1, 2, \ldots, M$$ \hspace{1cm} (4.5)

Where $M$ is the length of the randomized feature vector. The resulting randomized feature vector $R$ is comprised of angles bounded between 0 and $2\pi$. In the following sections the advantages of using angle features are presented and discussed for both hashing quantization and encoding. These discussions include performance analysis of angle features compared with standard real-valued features.

4.2.1 Comparison to Biophasors

The biophasor scheme is similar in construction to the proposed method. The length $M$ randomized feature vector produced by the biophasor scheme $\alpha = [\alpha_1, \alpha_2, \ldots, \alpha_M]$ is obtained from the length $N$ feature vector $F = [F_1, F_2, \ldots, F_N]$ and a orthonormal basis matrix $T$ of size $N \times M$ where each component is independent and drawn from a zero mean Gaussian distribution of unit variance. Each randomized feature component is produced by

$$R(j) = \alpha_j = \frac{1}{N} \sum_{i=1}^{N} \arctan \left( \frac{F_i}{T_{i,j}} \right) \text{ where } j = 1, 2, \ldots, M$$  \hspace{1cm} (4.6)

This can be compared to the proposed method by rewriting equation (4.5) as
CHAPTER 4. ROBUST HASHING USING ANGLE FEATURES

\[ R(j) = \phi_j = atan2 \left( \frac{\sum_{i=1}^{N} F_{i,v}^{Im}}{\sum_{i=1}^{N} F_{i,v}^{Re}} \right) \text{ where } j = 1, 2, \ldots, M \tag{4.7} \]

The similarities between the two techniques become apparent when comparing equation 4.6 and 4.7. Although similar, the proposed method has an advantage over biophasors. From equation 4.6, the biophasor features are obtained by computing \( \text{arctan} \left( \frac{F_i^T}{T_{i,j}} \right) \). The denominator \( T_{i,j} \) is randomly drawn from a standard normal distribution, as a result 95% of \( T_{i,j} \)'s will have a magnitude less than 2. But the magnitude of the numerator \( F_i \) is unknown and potentially large. Therefore the resulting fraction can produce large numbers and a visual inspection of the arctan function in Figure 4.1 shows for large inputs the arctan function approaches the horizontal asymptotes at \( \pm \pi \). Similarly, if the magnitude of \( F_i \) is small the arctan function will tend toward 0. This is a problem if the magnitude of \( F_i \) is large or small. For example Fourier-Mellin features are order \( 10^6 \) and the resulting randomized feature component will always be \( \pi \).

![Figure 4.1: The function \( y = \text{arctan}(x) \) asymptotes for large values of \( x \).](image)

In order to overcome this a scaling must be applied to \( F_i \), this can be achieved in one of two ways.

1. Scale each feature component \( F_i \) using the mean and standard deviation of that feature component \( \bar{F}_i \) and \( \sigma_{F_i} \).

2. Scale each feature component by the maximum component in the feature vector (\( \text{max}(F) \)).

However, both methods are problematic in a robust hashing framework. The first method requires knowledge of \( \bar{F}_i \) and \( \sigma_{F_i} \) and these are obtainable from
training samples, but the purpose of using angle features is to avoid the need for training. The second method does not work if the range of values in $\mathbf{F}$ is large. For example if $\text{max}(\mathbf{F})$ is three orders of magnitude larger than $\text{min}(\mathbf{F})$ then scaling by $\text{max}(\mathbf{F})$ pushes the smaller magnitude feature components towards 0. The presence of these small feature values will drive $\alpha_j$ towards 0.

In the proposed method, the issue with feature scaling is avoided because the arctan function is applied to the fraction

$$\frac{\sum_{i=1}^{N} F_i v_{i,j}^{lm}}{\sum_{i=1}^{N} F_i v_{i,j}^{Re}}$$

(4.8)

where both the numerator and denominator contain $F_i$. The summation operator does not allow the $F_i$ component to directly cancel (which is undesirable since it eliminates $F_i$ from the equation) but it does prevent large differences in magnitude from occurring except in the highly unlikely scenario that either $v_{i,j}^{Re}$ or $v_{i,j}^{lm}$ are all large or all small.

### 4.3 Encoding Benefits

In this section, the distance distortion introduced when angle features $\phi$ are encoded using Gray code is analyzed. It is theoretically proved that $\phi$ suffers from less distance distortion that other real-valued features used in hashing. This is because the cyclic nature of angles is shared by Gray code.

The concept of distance distortion was introduced in 2.4.2 and is introduced in the encoding stage of robust hashing. During the quantization stage, randomized features are thresholded into $L$ discrete levels where each level is assigned a unique integer between 1 and $L$. The encoding stage than assigns a binary bit-string to each of the $L$ quantization levels. This binarization process converts each real-valued integer into a binary bit-string and a distortion is introduced because the Cityblock distance (or $L_1$ norm) distance between two sets of quantized features is not necessarily preserved in the Hamming distance of their binarized counterparts. In other words, if $\mathbf{Q}_1$ and $\mathbf{Q}_2$ are the quantized feature vectors and $\mathbf{Y}_1$ and $\mathbf{Y}_2$ are the binarized versions then $d_1(\mathbf{Q}_1, \mathbf{Q}_2) \neq d_H(\mathbf{Y}_1, \mathbf{Y}_2)$.

The concept is illustrated by constructing the distance matrices using definitions 1 - 3 from Chapter 2. Distance matrices are constructed for standard
CHAPTER 4. ROBUST HASHING USING ANGLE FEATURES

integers \((\alpha_{i,j}^{(b)})\), standard binary \((\beta_{i,j}^{(b)})\) and GrayCode \((\gamma_{i,j}^{(b)})\). For \(\alpha_{i,j}^{(b)}\), the elements in the \(i\)-th row and \(j\)-th column is the Manhattan distance between \(i\) and \(j\). For the \(\beta_{i,j}^{(b)}\) and \(\gamma_{i,j}^{(b)}\) the elements are the Hamming distances between the binary representations of \(i\) and \(j\) encoded using standard binary and GrayCode respectively.

\[
\alpha_{i,j}^{(1)} = \frac{1}{2} \begin{pmatrix} 1 & 2 \\ 0 & 1 \end{pmatrix}, \quad \alpha_{i,j}^{(2)} = \begin{pmatrix} 1 & 2 & 3 & 4 \\ 0 & 1 & 2 & 3 \end{pmatrix}
\]

\[
\beta_{i,j}^{(1)} = 0 \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \quad \beta_{i,j}^{(2)} = \begin{pmatrix} 0 & 1 & 1 & 2 \\ 0 & 1 & 0 & 2 \end{pmatrix}
\]

\[
\gamma_{i,j}^{(1)} = 0 \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \quad \gamma_{i,j}^{(2)} = \begin{pmatrix} 0 & 1 & 2 & 1 \\ 0 & 1 & 0 & 2 \end{pmatrix}
\]

It can be seen from the above matrices when \(b = 1\) all three distance matrices are identical and therefore no distance distortion is introduced when encoding is applied to a 1-bit quantizer. However, when \(b\) increases to 2, \(\alpha_{i,j}^{(2)} \neq \beta_{i,j}^{(2)} \neq \gamma_{i,j}^{(2)}\) and distance distortion is introduced when a 2-bit quantizer is encoded with either standard binary or GrayCode. The amount of distortion can be quantized by averaging the difference between \(\alpha_{i,j}^{(b)}\) and either \(\beta_{i,j}^{(b)}\) or \(\gamma_{i,j}^{(b)}\) as presented in equation 2.12 and 2.13.

It was shown in [14] that distance distortion has a negative impact on hashing accuracy. As a result, minimizing distance distortion of the encoding phase can improve hashing performance. Angle features are one possible method of achieving this, before continuing a number of additional definitions related to angle features are first presented.
4.3 Encoding Benefits

Definition 12. The matrix $D_A^{(b)} = [\phi_{i,j}^{(b)}]_{L \times L}$ for $b \in \mathbb{Z}^+$ is called the cyclic-distance matrix of bit-order $b$, where:

$$
\phi_{i,j}^{(b)} = \begin{cases} 
|i - j| & \text{if } |i - j| < \frac{L}{2} \\
L - (|i - j|) & \text{otherwise}
\end{cases}
$$

(4.9)

for $i, j = 1, \ldots, L$ and $L = 2^b$.

Definition 13. The distance distortion introduced when a cyclic angle feature is encoded using Gray code can be expressed by:

$$
M(D^{(b)}_{A,G}) = \frac{1}{L^2} \sum_{i=1}^{L} \sum_{j=1}^{L} |\phi_{i,j}^{(b)} - \gamma_{i,j}^{(b)}|
$$

(4.10)

By using definition 12, the cyclic integer distance matrix $(\phi_{i,j}^{(b)})$ for $b = 1, b = 2$ and $b = 3$ are presented below:

$$
\phi_{i,j}^{(1)} = \begin{pmatrix} 1 & 2 \\
2 & 1 \end{pmatrix}, \quad \phi_{i,j}^{(2)} = \begin{pmatrix} 1 & 0 & 1 & 2 \\
2 & 1 & 0 & 1 \\
3 & 2 & 1 & 0 \end{pmatrix}, \quad \phi_{i,j}^{(3)} = \begin{pmatrix} 1 & 2 & 3 & 4 \\
1 & 0 & 1 & 2 \\
2 & 1 & 0 & 1 \\
3 & 2 & 1 & 0 \end{pmatrix}
$$

The advantage of using cyclic integer features (such as angles) is immediately obvious because $\phi_{i,j}^{(2)} = \gamma_{i,j}^{(2)}$ and no distance distortion is introduced when a cyclic integer is encoded using GrayCode when $b = 2$. In the remainder of the section, it is proven that distance distortion of cyclic integers encoded with GrayCode is always less than standard integers encoded with GrayCode for all values of $b$.

Some lemmas needed for the theoretic analysis are first presented.

Lemma 14. (From Zhu et al. [14]). Let $D_G^{(b)} = [\gamma_{i,j}^{(b)}]_{L \times L}$ for $b \in \mathbb{Z}^+$ denote the Gray-distance matrix of $b$ bits, where $L = 2^b$. Then

$$
D_G^{(b+1)} = \begin{bmatrix} D_G^{(b)} & D_G^{(b)} \bar{I}^{(b)} + O^{(b)} \\
D_G^{(b)} I^{(b)} + O^{(b)} & D_G^{(b)} \end{bmatrix}
$$

(4.11)

where $O^{(b)}$ is an $L \times L$ matrix of ones and $\bar{I}^{(b)}$ is the $L \times L$ exchange matrix with ones along the anti-diagonal and zeros elsewhere.
CHAPTER 4. ROBUST HASHING USING ANGLE FEATURES

Lemma 15. Let $D^{(b)}_G = [\gamma^{(b)}_{i,j}]_{L \times L}$ for $b \in \mathbb{Z}^+$ denote the Gray-distance matrix where $L = 2^k$. Then $D^{(b)}_G$ is a symmetric matrix.

Since Hamming distance is symmetric measure of distance and from the definition of $D^{(b)}_G$ in Chapter 2, the above Lemma is obviously true. It’s symmetry has also been discussed in [14] and [48].

Lemma 16. Let $D^{(b)}_I = [\alpha^{(b)}_{i,j}]_{L \times L}$ for $b \in \mathbb{Z}^+$ denote the standard-distance matrix where $L = 2^k$. Then $D^{(b)}_I$ is a symmetric matrix.

Proof. A matrix $A = [a_{i,j}]_{L \times L}$ is symmetric if it satisfies $a_{i,j} = a_{j,i}$. From definition [12] in Chapter 2, $\alpha_{i,j} = \alpha_{j,i}$ because $|i - j| = |j - i|$. Therefore, $D^{(b)}_I$ is a symmetric matrix.

Lemma 17. Let $D^{(b)}_A = [\phi^{(b)}_{i,j}]_{L \times L}$ for $b \in \mathbb{Z}^+$ denote the cyclic-distance matrix where $L = 2^k$. Then $D^{(b)}_A$ is a symmetric matrix.

Proof. A matrix $A = [a_{i,j}]_{L \times L}$ is symmetric if $a_{i,j} = a_{j,i}$. From definition [12] in Chapter 2, $\phi_{i,j} = \phi_{j,i}$ because $|i - j| = |j - i|$. Therefore, $D^{(b)}_A$ is a symmetric matrix.

Lemma 18. Let $D^{(b)}_A = [\phi^{(b)}_{i,j}]_{L \times L}$ and $D^{(b)}_I = [\alpha^{(b)}_{i,j}]_{L \times L}$ for $b \in \mathbb{Z}^+$ denote the cyclic-distance matrix and the standard-distance matrix respectively, where $L = 2^k$. Then $\alpha^{(b)}_{i,j} \geq \phi^{(b)}_{i,j}$ for $1 \leq i, j \leq L$ and $b \in \mathbb{Z}^+$.

Proof. From Definition [12] in Chapter 2 and [12]

\[
\phi^{(b)}_{i,j} = \begin{cases} 
\alpha^{(b)}_{i,j} & \text{if } \alpha^{(b)}_{i,j} < \frac{L}{2} \\
L - \alpha^{(b)}_{i,j} & \text{if } \alpha^{(b)}_{i,j} \geq \frac{L}{2}
\end{cases}
\] (4.12)

For the case when $\alpha^{(b)}_{i,j} < \frac{L}{2}$, it is clear that $\phi^{(b)}_{i,j} = \alpha^{(b)}_{i,j}$ and when $\alpha^{(b)}_{i,j} \geq \frac{L}{2}$ then $\phi^{(b)}_{i,j} = L - \alpha^{(b)}_{i,j} \leq \alpha^{(b)}_{i,j}$ (since $\alpha^{(b)}_{i,j} \geq \frac{L}{2}$). Hence, $\alpha^{(b)}_{i,j} \geq \phi^{(b)}_{i,j}$ for all cases and the Lemma holds.

Lemma 19. From Lemma 11 in [14], let $D^{(b)}_I = [\alpha^{(b)}_{i,j}]_{L \times L}$ and $D^{(b)}_G = [\gamma^{(b)}_{i,j}]_{L \times L}$ for $b \in \mathbb{Z}^+$ denote the ideal-distance matrix and the Gray-distance matrix respectively, where $L = 2^k$. Then $\alpha^{(b)}_{i,j} \geq \gamma^{(b)}_{i,j}$ for $1 \leq i, j \leq L$ and $b \in \mathbb{Z}^+$.

Proof. This lemma is proven by induction. When $b = 1$, $\alpha^{(1)}_{1,1} = 0 = \gamma^{(1)}_{1,1}$, $\alpha^{(1)}_{1,2} = 1 = \gamma^{(1)}_{1,2}$, $\alpha^{(1)}_{2,1} = 1 = \gamma^{(1)}_{2,1}$ and $\alpha^{(1)}_{2,2} = 0 = \gamma^{(1)}_{2,2}$ and the lemma holds for $b = 1$. To prove the lemma is also true for $b = k + 1$, we consider each quadrant of the matrices in turn.
4.3 Encoding Benefits

Case 1: When $1 \leq i, j \leq 2^k$ (the top left corner of the matrix), according to the definition of $D_I^{(b)}$ and lemma 19, we obtain $\alpha_{i,j}^{(k+1)} = \alpha_{i,j}^{(k)} \geq \gamma_{i,j}^{(k)} = \gamma_{i,j}^{(k+1)}$.

Case 2: When $2^k < i, j \leq 2^{k+1}$ (the bottom right corner), it is easy to obtain

$$\alpha_{i,j}^{(k+1)} = \alpha_{i-2^k, j-2^k}^{(k)}$$

(4.13)

also according to lemma 14

$$\gamma_{i,j}^{(k+1)} = \gamma_{i-2^k, j-2^k}^{(k)}$$

(4.14)

Then, combining equation 4.13 and 4.14, we have $\alpha_{i,j}^{(k+1)} \geq \gamma_{i,j}^{(k+1)}$ for $2^k < i, j / e 2^{k+1}$.

Case 3: When $1 \leq i \leq 2^k$ and $2^k < j \leq 2^{k+1}$ (top right corner), according to Lemma 14

$$\gamma_{i,j}^{(k+1)} = \gamma_i^{k,2^k+1-j} + 1 \leq \alpha_i^{k,2^k+1-j} + 1 = |i + j - 2^{k+1} - 1| + 1$$

(4.15)

Rewriting equation 4.15 we obtain

$$\gamma_{i,j}^{(k+1)} \leq j + i - 2^{k+1} \leq j - i \text{ if } i + j - 2^{k+1} - 1 \geq 0$$

$$\gamma_{i,j}^{(k+1)} \leq 2^{k+1} + 2 - j - i \leq j - i \text{ if } i + j - 2^{k+1} - 1 < 0$$

(4.16)

Then, according to equation 4.16 we get $\alpha_{i,j}^{(k+1)} = |i - j| \geq \gamma_{i,j}^{(k+1)}$ for $1 \leq i \leq 2^k$ and $2^k < j \leq 2^{k+1}$.

Case 4: When $2^k < i \leq 2^{k+1}$ and $1 \leq j \leq 2^k$ (bottom left corner). Recall from Lemma 15 and Lemma 16 that both $D_G^{(b)}$ and $D_I^{(b)}$ are symmetric. The bottom left corner of both matrices are a reflection of the top right corner and case 3 applies.

In summary this lemma is true for $b = k$, then it is still true for $b = k + 1$ and this lemma holds.

Lemma 20. Let $D_A^{(b)} = [\phi_{i,j}^{(b)}]_{L \times L}$ and $D_G^{(b)} = [\gamma_{i,j}^{(b)}]_{L \times L}$ for $b \in \mathbb{Z}^+$ denote the cyclic-distance matrix and the Gray-distance matrix respectively, where $L = 2^b$. Then $\phi_{i,j}^{(b)} \geq \gamma_{i,j}^{(b)}$ for $c$ and $b \in \mathbb{Z}^+$.

Proof. This lemma is proven by induction. Consider when $b = 1$, $\phi_{1,1}^{(b)} = 0 = \gamma_{1,1}^{(b)}$. 87
CHAPTER 4. ROBUST HASHING USING ANGLE FEATURES

$\phi_{1,2}^{(b)} = 1 = \gamma_{1,2}^{(b)}$, $\phi_{2,1}^{(b)} = 1 = \gamma_{2,1}^{(b)}$ and $\phi_{2,2}^{(b)} = 0 = \gamma_{2,2}^{(b)}$ and the lemma holds for $b = 1$. Assuming the lemma is true for $b = k$ it is proved that the lemma is also true for $b = k + 1$. Similar to Lemma 19 we consider four cases.

**Case 1:** When $1 \leq i, j \leq 2^k$ it’s easy to see that $|i - j| < 2^k$ and using definition 12 and 1 $\phi_{i,j}^{(k+1)} = |i - j| = \alpha_{i,j}^{(k+1)}$ and according to Lemma 18 $\alpha_{i,j}^{(k+1)} \geq \gamma_{i,j}^{(k+1)}$ so $\phi_{i,j}^{(k+1)} \geq \gamma_{i,j}^{(k+1)}$.

**Case 2:** When $2^k < i, j \leq 2^{k+1}$, then $|i - j| < 2^k$ and from case 1 it is obvious that $\phi_{i,j}^{(k+1)} \geq \gamma_{i,j}^{(k+1)}$.

**Case 3:** When $1 \leq i \leq 2^k$ and $2^k < j \leq 2^{k+1}$ then $\phi_{i,j}^{(k+1)}$ can be rewritten as:

$$
\phi_{i,j}^{(k+1)} = \begin{cases} 
    j - i & \text{if } j - i < 2^k \\
    2^{k+1} - j + i & \text{otherwise}
\end{cases}
$$

(4.17)

According to Lemma 14

$$
\gamma_{i,j}^{(k+1)} = \gamma_{i,2^{k+1}-j+1}^{(k)} + 1
$$

(4.18)

From Lemma 19 it follows that $\alpha_{i,j}^{(k)} \geq \gamma_{i,j}^{(k)}$ so equation (4.18) can be rewritten as

$$
\gamma_{i,j}^{(k+1)} \leq \alpha_{i,2^{k+1}-j+1}^{(k)} + 1
$$

(4.19)

and by definition of $\alpha_{i,j}$, (4.19) becomes

$$
\gamma_{i,j}^{(k+1)} \leq |i - (2^{k+1} - j + 1)| + 1 = |i + j - 2^{k+1} + 1| + 1
$$

(4.20)

which can be rewritten as

$$
\gamma_{i,j}^{(k+1)} \leq \begin{cases} 
    j + i - 2^{k+1} & \text{if } i + j - 2^{k+1} - 1 \geq 0 \\
    2^{k+1} + 2 - j - i & \text{if } i + j - 2^{k+1} - 1 < 0
\end{cases}
$$

(4.21)

It can be shown that $\phi_{i,j}^{(k+1)} \geq \gamma_{i,j}^{(k+1)}$ by showing each case in equation (4.17) is greater than or equal to each case in equation (4.21). By simplifying the following
four inequalities:

\[ j - i \geq j + i - 2^{k+1} \]
\[ j - i \geq 2^{k+1} + 2 - j - i \]
\[ 2^{k+1} - j + i \geq j + i - 2^{k+1} \]
\[ 2^{k+1} - j + i \geq 2^{k+1} + 2 - j - i \]

1. For \( j - i \geq j + i - 2^{k+1} \):

\[ 2^{k+1} \geq 2i \]
\[ 2^k \geq i \text{ (True since } 1 \leq i \leq 2^k \) }

2. For \( j - i \geq 2^{k+1} + 2 - j - i \):

\[ 2j \geq 2^{k+1} + 2 \]
\[ j \geq 2^k + 1 \text{ (True since } 2^k < j \leq 2^{k+1} \) }

3. For \( 2^{k+1} - j + i \geq j + i - 2^{k+1} \):

\[ 2^{k+1} + 2^{k+1} \geq 2j \]
\[ 2^{k+1} \geq j \text{ (True since } 2^k < j \leq 2^{k+1} \) }

4. For \( 2^{k+1} - j + i \geq 2^{k+1} + 2 - j - i \):

\[ 2^{k+1} + i \geq 2^{k+1} + 2 - i \]
\[ 2i \geq 2 \]
\[ i \geq 1 \text{ (True since } 1 \leq i \leq 2^k \) }

**Case 4:** When \( 2^k < i \leq 2^{k+1} \) and \( 1 \leq j \leq 2^k \), due to the symmetry of \( D_G^{(b)} \) and \( D_A^{(b)} \) (Lemma 15 and Lemma 17) case 3 applies.

Thus this Lemma is true for \( b = k \) and \( b = k + 1 \) hence by induction this lemma holds.

The above Lemmas highlight the properties of real-valued cyclic features, real-valued non-cyclic features and Gray encoding necessary to understand the
CHAPTER 4. ROBUST HASHING USING ANGLE FEATURES

following theorems.

Theorem 21. Let $D_A(b) = [\phi_{i,j}]_{L \times L}$, $D_I(b) = [\alpha_{i,j}]_{L \times L}$ and $D_G(b) = [\gamma_{i,j}]_{L \times L}$ for $n \in \mathbb{Z}^+$ denote the cyclic-distance matrix, the standard-distance matrix and the Gray-distance matrix respectively, where $L = 2^b$. Then $M(D_I,G) > M(D_A,G)$ for $b \geq 2$.

Proof. From Lemma 18 and 20 it can be seen that $\alpha_{i,j} \geq \phi_{i,j} \geq \gamma_{i,j}$ and equation (2.13) can be rewritten as

$$M(D_{I,G}) = \frac{1}{L^2} \sum_{i=1}^{L} \sum_{j=1}^{L} |\alpha_{i,j} - \gamma_{i,j}|$$

$$= \frac{1}{L^2} \left( \sum_{i=1}^{L} \sum_{j=1}^{L} \alpha_{i,j} - \sum_{i=1}^{L} \sum_{j=1}^{L} \gamma_{i,j} \right)$$

(4.22)

Similarly, equation (4.10) in definition 13 becomes

$$M(D_{A,G}) = \frac{1}{L^2} \sum_{i=1}^{L} \sum_{j=1}^{L} |\phi_{i,j} - \gamma_{i,j}|$$

$$= \frac{1}{L^2} \left( \sum_{i=1}^{L} \sum_{j=1}^{L} \phi_{i,j} - \sum_{i=1}^{L} \sum_{j=1}^{L} \gamma_{i,j} \right)$$

(4.23)

Using Lemma 20 and equations (4.22) and (4.23) it is easy to see that

$$\left( \sum_{i=1}^{L} \sum_{j=1}^{L} \alpha_{i,j} - \sum_{i=1}^{L} \sum_{j=1}^{L} \gamma_{i,j} \right) > \left( \sum_{i=1}^{L} \sum_{j=1}^{L} \phi_{i,j} - \sum_{i=1}^{L} \sum_{j=1}^{L} \gamma_{i,j} \right)$$

(4.24)

and therefore $M(D_{I,G}) > M(D_{A,G})$. □

Theorem 22. Let $[\phi_{i,j}]_{L \times L}$ and $[\gamma_{i,j}]_{L \times L}$ for $n \in \mathbb{Z}^+$ denote the cyclic-distance matrix and the Gray-distance matrix respectively, where $L = 2^b$. Then $M(D_{A,G}) = 0$ for $b \leq 2$ and $M(D_{A,G}) > 0$ for $b \geq 3$.

Proof. Using equation (2.9) and Definition 12

$$\phi_{i,j}^{(1)} = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \quad \text{and} \quad \gamma_{i,j}^{(1)} = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$
4.3 Encoding Benefits

\[ \phi^{(2)}_{i,j} = \begin{bmatrix} 0 & 1 & 2 & 1 \\ 1 & 0 & 1 & 2 \\ 2 & 1 & 0 & 1 \\ 1 & 2 & 1 & 0 \end{bmatrix} \quad \text{and} \quad \gamma^{(2)}_{i,j} = \begin{bmatrix} 0 & 1 & 2 & 1 \\ 1 & 0 & 1 & 2 \\ 2 & 1 & 0 & 1 \\ 1 & 2 & 1 & 0 \end{bmatrix} \]

It can be seen that \( D^{(1)}_A = D^{(1)}_G \) and \( D^{(2)}_A = D^{(2)}_G \) thus \( M(D^{(b)}_{A,G}) = 0 \) for \( b \leq 2 \).

The maximum distance in \( D^{(b)}_G = b \) and the maximum distance in \( D^{(b)}_A = \frac{2^b}{2} \), when \( b \geq 3 \) it is seen that \( \frac{2^b}{2} > b \) (where as for \( b < 3, \frac{2^b}{2} = b \)). Recall from Lemma 20 that \( \theta^{(b)}_{i,j} \geq \gamma^{(b)}_{i,j} \) and since \( \frac{2^b}{2} > b \) then \( D^{(b)}_A > D^{(b)}_G \) and \( M(D^{(b)}_{A,G}) > 0 \) for \( b \geq 3 \).

An example is shown below for \( b = 3 \).

\[ \phi^{(3)}_{i,j} = \begin{bmatrix} 0 & 1 & 2 & 3 & 4 & 3 & 2 & 1 \\ 1 & 0 & 1 & 2 & 3 & 4 & 3 & 2 \\ 2 & 1 & 0 & 1 & 2 & 3 & 4 & 3 \\ 3 & 2 & 1 & 0 & 1 & 2 & 3 & 4 \\ 4 & 3 & 2 & 1 & 0 & 1 & 2 & 3 \\ 3 & 4 & 3 & 2 & 1 & 0 & 1 & 2 \\ 2 & 3 & 4 & 3 & 2 & 1 & 0 & 1 \\ 1 & 2 & 3 & 4 & 3 & 2 & 1 & 0 \end{bmatrix} \quad \text{and} \quad \gamma^{(3)}_{i,j} = \begin{bmatrix} 0 & 1 & 2 & 1 & 2 & 3 & 2 & 1 \\ 1 & 0 & 1 & 2 & 3 & 2 & 1 & 2 \\ 2 & 1 & 0 & 1 & 2 & 3 & 2 & 1 \\ 1 & 2 & 1 & 0 & 1 & 2 & 3 & 2 \\ 2 & 3 & 2 & 1 & 0 & 1 & 2 & 1 \\ 3 & 2 & 1 & 2 & 1 & 0 & 1 & 2 \\ 2 & 1 & 2 & 3 & 2 & 1 & 0 & 1 \\ 1 & 2 & 3 & 2 & 1 & 2 & 1 & 0 \end{bmatrix} \]

From Theorems 21 and 22 it can be concluded that angle features suffer less distance distortion than other real-valued features that are not cyclic, when encoded using Gray code. Further angle features suffer no distance distortion when \( b \leq 2 \). The reduced distance distortion makes such features attractive for robust hashing. A comparison of distance distortion for standard integer features and cyclic integer features is summarised in Table 4.1 for \( b = 1, \ldots, 8 \). This table is generated using the distance distortion equations presented above and in Chapter 2. It summarises the theoretical analysis of this section and shows that cyclic integer features encoded with GrayCode suffers the least distance distortion. In the following section, experimental verification of these theoretical findings are conducted.
CHAPTER 4. ROBUST HASHING USING ANGLE FEATURES

<table>
<thead>
<tr>
<th>Distance Distortions</th>
<th>Standard Integer</th>
<th>Cyclic Integer</th>
</tr>
</thead>
<tbody>
<tr>
<td>bits ( (b) )</td>
<td>Binary ( M(D_{I,B}^{(b)}) )</td>
<td>GrayCode ( M(D_{I,G}^{(b)}) )</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>0.5</td>
<td>0.25</td>
</tr>
<tr>
<td>3</td>
<td>1.375</td>
<td>1.125</td>
</tr>
<tr>
<td>4</td>
<td>3.531</td>
<td>3.313</td>
</tr>
<tr>
<td>5</td>
<td>8.313</td>
<td>8.156</td>
</tr>
<tr>
<td>6</td>
<td>18.425</td>
<td>18.328</td>
</tr>
<tr>
<td>8</td>
<td>81.364</td>
<td>81.332</td>
</tr>
</tbody>
</table>

Table 4.1: Distance distortions for standard and cyclic integers encoded using standard binary and GrayCode for \( b = 1, \ldots, 8 \).

4.3.1 Experimental Results

A number of experiments are carried out to compare the hashing accuracy of cyclic features compared to standard integer features. Randomized features are produced using four algorithms, the first two being the baseline NMF and Fourier-Mellin schemes described in Chapter 2 and the remaining two methods use NMF and Fourier-Mellin features in conjunction with the proposed method of complex projection described in this Chapter. The encoding schemes used are standard binary, Gray code and no encoding. For the case of no encoding, the quantized integer-valued feature vectors are used directly without binarization and the Manhattan distance (\( L_1 \) norm) is used to compare similarity between two hashes. Quantization is restricted to adaptive deterministic quantization using quantizer training scenario 1. The limitations of such quantization was discussed in Chapter 3, but its use in these experiments is justified because the aim of the experiment is not to measure the absolute performance but instead compare the relative performance difference between different encoding schemes and feature types.

These experiments are carried out using 1000 images from the MIRFLICKR database. Each of these 1000 original images is modified using a combination of different image transformations including JPEG compression at 50% quality, additive Gaussian noise, mean filtering with 3x3 kernal, cropping by 5 pixels, rotation by 4 degrees and histogram equalisation. The methodology for gener-
4.3 Encoding Benefits

ating modified images is slightly varied from that in Chapter 3. In the previous chapter, each of the 5 modifications was applied separately so that each modified image contains only one type of transformation. In this chapter, each modified image contains all 5 image transformations. With this approach each image displays more variation, although the pool of modified images is reduced from 5000 images to only 1000 images.

Hashing accuracy is quantifiably measured using ROC curves and different feature types and encoding schemes can be compared. The ROC curves are generated in exactly the same manner as described in Chapter 3.

![ROC curves for encoding benefits](image)

Figure 4.2: Performance comparisons for no encoding, standard binary and Gray-Code for Fourier-Mellin features and random projection.

The ROC curves in Figures 4.2 and 4.3 are for the baseline methods, Fourier-Mellin with random projection and NMF with random projection respectively. When no encoding is applied the EER performance of the Fourier-Mellin method
Figure 4.3: Performance comparisons for no encoding, standard binary and Gray-Code for NMF features and random projection.
4.3 Encoding Benefits

Figure 4.4: Performance comparisons for no encoding, standard binary and Gray-Code for Fourier-Mellin features and Complex random projection.
CHAPTER 4. ROBUST HASHING USING ANGLE FEATURES

Figure 4.5: Performance comparisons for no encoding, standard binary and Gray-Code for NMF features and Complex random projection.
4.3 Encoding Benefits

gradually improves as \( b \) increases. This decrease in EER is attributed to the absence of distance distortion which preserves the recognitions gains provided by increased feature space resolution as the number of quantization levels increases. Interestingly, the EER performance of the NMF method with no encoding, does not monotonically decrease with \( b \). Recognition accuracy drops slightly then plateaus as \( b \) is increased above 2. This behaviour is because the increased feature space resolution has made the method too sensitive to the transformations present in the modified images.

For standard binary encoding which has the largest distance distortion (see Table 4.1) EER performance degrades as soon as \( b \) increases above 1. Whilst for Gray encoding, which has less distance distortion this degradation occurs only when \( b > 2 \). These results demonstrate the behaviour between \( b \) and EER performance. Intuitively, increasing \( b \) should increase performance which is the case when no encoding is applied. But distance distortion also increases with \( b \), the amount of distortion is dependent on both \( b \) and the choice of encoding. The presence of distortion decreases performance and this produces a trade-off scenario between distance distortion and increased feature space resolution. For random projection based methods this occurs when \( b > 1 \). The optimal choice of \( b \) is 1 for standard binary and 2 for Gray code.

When using complex projections to produce angle features distance distortion is reduced compared to standard integer features. From Theorem 22 angle features display no distance distortion with Gray code for \( b = 2 \) and this can be seen in Figures 4.4 and 4.5 where the ROC curves for no encoding and Gray code are identical when \( b = 2 \). It is also evident when \( b > 2 \) performance degrades more gradually than the baseline systems which is consistent with Theorem 21. These results confirm the theoretical proofs obtained in the previous section, angle feature are better suited for encoding than standard integers due to the decreased distance distortion.

Another point to note is the minor difference in EER performance between random projection and complex projection. This difference arises due to the slight difference in feature randomization, namely the conversion from a complex valued integer into an angle feature that is present in the complex projection. The difference is minor, roughly a 0.5% change in EER performance. For the Fourier-Mellin based method the complex projection is better than the random projection. Whilst for the NMF method the reverse is true. Only a small change
exists because both methods rely on the same underlying image features.

4.4 Quantization Benefits

Chapter 2 introduced uniform quantization where quantization levels are uniformly spaced between the minimum and maximum feature values learnt from training observations. However, with angle features these bounds are known a priori and equation (2.6) used to determine quantization thresholds becomes

$$\frac{\max(R) - \min(R)}{L} = \frac{2\pi - 0}{L} = \frac{2\pi}{L}$$

Where $L$ is the number of discrete quantization levels. This simple modification produces a significant advantage, quantization thresholds for uniform quantization can be obtained without training data. They are also independent of the original feature vector $F$ and would be the same for all $M$ feature components.

Tables 4.2 and 4.3 shows the EER and fragility of hashing schemes using different hashing schemes and quantization training scenarios. The presented results are obtained from the experiments carried out in the previous section but using uniform quantization instead of adaptive quantization. Consequently the results presented below are different from those obtained in previous sections which were for adaptive quantization only. The tables show the EER and fragility for both NMF-Complex and FM-Complex are identical for all quantization scenarios when uniform quantization with known limits is applied.

Table 4.2: EER of hashing schemes for different quantizer training scenarios using Uniform Quantization.

<table>
<thead>
<tr>
<th>Quantizer Training Scenario</th>
<th>1-bit</th>
<th>2-bit</th>
<th>3-bit</th>
<th>4-bit</th>
<th>5-bit</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fourier Mellin-RP Scheme</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1-bit</td>
<td>18.42%</td>
<td>19.05%</td>
<td>19.15%</td>
<td>18.79%</td>
<td>16.95%</td>
</tr>
<tr>
<td>4-bit</td>
<td>12.79%</td>
<td>12.63%</td>
<td>11.95%</td>
<td>12.66%</td>
<td>12.98%</td>
</tr>
<tr>
<td>NMF-RP Scheme</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1-bit</td>
<td>14.75%</td>
<td>14.48%</td>
<td>19.35%</td>
<td>49.69%</td>
<td>13.80%</td>
</tr>
<tr>
<td>4-bit</td>
<td>10.33%</td>
<td>10.48%</td>
<td>10.58%</td>
<td>12.79%</td>
<td>11.43%</td>
</tr>
<tr>
<td>Fourier-Mellin Complex</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1-bit</td>
<td>14.95%</td>
<td>14.95%</td>
<td>14.95%</td>
<td>14.95%</td>
<td>14.95%</td>
</tr>
<tr>
<td>4-bit</td>
<td>12.76%</td>
<td>12.76%</td>
<td>12.76%</td>
<td>12.76%</td>
<td>12.76%</td>
</tr>
<tr>
<td>NMF Complex</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1-bit</td>
<td>12.84%</td>
<td>12.84%</td>
<td>12.84%</td>
<td>12.84%</td>
<td>12.84%</td>
</tr>
<tr>
<td>4-bit</td>
<td>10.60%</td>
<td>10.60%</td>
<td>10.60%</td>
<td>10.60%</td>
<td>10.60%</td>
</tr>
</tbody>
</table>

In most cases for uniform quantization, the complex projection outperforms the random projection counterpart in both EER and fragility. Furthermore, both
4.4 Quantization Benefits

Table 4.3: Fragility of hashing schemes for different quantizer training scenarios using Uniform Quantization.

<table>
<thead>
<tr>
<th>Quantizer Training Scenario</th>
<th>1-bit</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fourier Mellin-RP Scheme</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1-bit</td>
<td>0.10</td>
<td>0.13</td>
<td>0.11</td>
<td>0.10</td>
<td>0.07</td>
</tr>
<tr>
<td>4-bit</td>
<td>0.17</td>
<td>0.17</td>
<td>0.27</td>
<td>0.16</td>
<td>0.13</td>
</tr>
<tr>
<td>NMF-RP Scheme</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1-bit</td>
<td>0.07</td>
<td>0.07</td>
<td>0.05</td>
<td>0</td>
<td>0.07</td>
</tr>
<tr>
<td>4-bit</td>
<td>0.10</td>
<td>0.14</td>
<td>0.15</td>
<td>0.03</td>
<td>0.08</td>
</tr>
<tr>
<td>Fourier-Mellin Complex</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1-bit</td>
<td>0.14</td>
<td>0.14</td>
<td>0.14</td>
<td>0.14</td>
<td>0.14</td>
</tr>
<tr>
<td>4-bit</td>
<td>0.21</td>
<td>0.21</td>
<td>0.21</td>
<td>0.21</td>
<td>0.21</td>
</tr>
<tr>
<td>NMF Complex</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1-bit</td>
<td>0.08</td>
<td>0.08</td>
<td>0.08</td>
<td>0.08</td>
<td>0.08</td>
</tr>
<tr>
<td>4-bit</td>
<td>0.13</td>
<td>0.13</td>
<td>0.13</td>
<td>0.13</td>
<td>0.13</td>
</tr>
</tbody>
</table>

EER performance and fragility are independent of quantization training. This is a significant advantage of using complex projection, quantization training is not required for the uniform quantizer. However, hashing fragility is still low for the NMF-Complex and FM-Complex methods. This is because the randomized feature components follow a non-uniform distribution, recall from Chapter 2 that uniform quantization is inferior to adaptive deterministic quantization for non-uniformly distributed features.

Both complex and random projection do not produce uniform features because for a single secret-key the projection basis $v_{i,j}^{Re}$ and $v_{i,j}^{Im}$ are constants. Therefore both the numerator $(\sum_{i=1}^{N} F_i v_{i,j}^{Im})$ and denominator $(\sum_{i=1}^{N} F_i v_{i,j}^{Im})$ in equation 4.8 vary only due to the feature component $F_i$. There are two weaknesses associated with this dependency on $F_i$

1. If the magnitude of $F_i$ is dependent on $i$, in other words each feature component is scaled diferently which is the case for Fourier-Mellin features. Then the resulting sum becomes dominated by the larger feature components and the distribution of randomized features becomes non-uniform.

2. If the distribution of $F_i$ for different images is tightly clustered (has low variance) then the ratio of $\sum_{i=1}^{N} F_i v_{i,j}^{Re}$ and $\sum_{i=1}^{N} F_i v_{i,j}^{Im}$ will tend to the ratio of $\sum_{i=1}^{N} v_{i,j}^{Re}$ and $\sum_{i=1}^{N} v_{i,j}^{Im}$ which is a constant since $v^{Re}$ and $v^{Im}$ are constants.

Both of these weaknesses are inherent in random and complex projection and results in randomized features of non-uniform distribution. This problem can
be overcome in one of two ways. Use features where the scale of each feature component $F_i$ is consistent for all $i$. Or develop a projection scheme where $v_{i,j}^{Re}$ and $v_{i,j}^{Im}$ are also functions of the image. If this can be achieved, the performance of a uniform quantizer will approach that of the adaptive quantizer. If angle features are used than high fragility and performance can be achieved without the need for quantizer training. A method of doing this is discussed in Chapter 5.

4.5 Chapter Summary

This chapter has presented a modification to the standard random projection that can produce bounded angle features. Angles are obtained by projecting the feature vector $F$ with a complex-valued basis and retaining only the argument of the resulting complex-valued vector. This complex-projection is applied to both Fourier-Mellin and NMF image features and is referred to as FM-complex and NMF-complex.

The use of angle features is motivated for two reasons, firstly angles are cyclic and they suffer less distance distortion because they better match the cyclic behaviour of Gray code. The reduced distance distortion is theoretically proven for all $b$. Secondly, angles are bounded between 0 and $2\pi$ and this allows uniform quantization to be carried out without quantizer training. Eliminating quantizer training is important part of improving hashing security as discussed in Chapter 3.
Chapter 5

Robust Hashing Using Bispectra

5.1 Introduction

Chapters 2 and 3 described a number of weaknesses that effect random projection based robust hash functions. This chapter introduces a new randomization technique for image hashing based on the bispectrum. The method produces a set of projection basis that are a non-linear function of both the input image and secret-key. This is achieved by extracting bispectral invariants from the 1D Radon projection of the input image. These Radon projections are first permuted using the secret-key prior to extracting invariants. This method has a number of advantages:

1. In random projection based randomization, one factor of the inner product is solely dependent on the secret-key and the other is solely dependent on the input image. This is evident from inspection of equation (2.3). In the proposed technique both both factors are a function of the secret-key and input image, this prevents the use of unicity distance attacks proposed in [10] and is the first method (to our knowledge) that uses an input dependent secret-key as suggested by Khelifi [9].

2. Furthermore, image dependent projection basis also contain image information and are themselves image features. This essentially makes the randomization stage a form of feature level fusion and this has the potential to increase hashing performance.

3. HOS invariants are complex-numbers and the resulting randomized features
CHAPTER 5. ROBUST HASHING USING BISPECTRA

retain the advantages such as bounded outputs and reduced distance distortion described in Chapter 4.

This chapter begins with an exploration of HOS and the motivations for using it in robust image hashing. A description of the proposed technique is given in Section and is followed by an analysis of the invariance properties of HOS invariants for common image transformations. A thorough performance comparison of the proposed system to the baseline systems described in section is also provided.

5.2 Higher Order Spectra

Higher order spectra were originally defined and developed in analysing random sequences. A random sequence or random signal is represented in the frequency domain through the use of spectral representations of the cumulants of the joint probability density functions. The first order cumulant is the expected value or mean value of the signal. For wide sense stationary random signals, the second order cumulant is the autocorrelation. Its Fourier spectrum is the well known power spectrum. The autocorrelation is a function of one lag variable and the power spectrum is a function of the one corresponding frequency variable for a one-dimensional signal. For a two-dimensional signal, the lag is a vector with $x$ and $y$ spatial components and the frequency is also a corresponding two-dimensional vector of frequency components. The autocorrelation can be defined over an ensemble of realizations of the random signal or over the time (or space) domain. For ergodic signals, the ensemble autocorrelation and time autocorrelation are equal. The first and second cumulants (mean value and covariance) are sufficient to describe a Gaussian probability density function. For a Gaussian random signal higher order cumulants are all zero. The Power spectrum is sufficient for its analysis.

For a zero-mean, real, ergodic random process $x(n)$ the second order cumulant $c_2(\tau_1)$ is:

$$c_2(\tau_1) = E[x(n).x(n + \tau_1)]$$ (5.1)

Where $E[\cdot]$ denotes the expectation operator and the Power spectrum of $x(n)$ is therefore:

102
5.2 Higher Order Spectra

\[
S_2(f_1) = \sum_{\tau_1 = -\infty}^{+\infty} c_2(\tau_1)e^{-j2\pi(f_1 \tau_1)} \tag{5.2}
\]

However, many real signals are non-Gaussian in nature and require higher-than-second order cumulants to fully describe them. These higher order cumulants are functions of more than one lag variable even for one dimensional signals. The third cumulant is a function of two lag variables. A spectrum can be defined for it provided it satisfies Dirichlet conditions for the existence of a Fourier transform. Unlike the power spectrum, the Fourier transform of the third cumulant, called the bispectrum, is complex-valued. Further, unlike the power spectrum, it contains phase information from Fourier components in the random signal. This is a particularly useful property and the bispectrum has been used in investigating phase coupling between harmonics that arise when a random signal passes through a non-linear system.

The bispectrum, \( B(f_1, f_2) \), of \( x(n) \) is defined as the Fourier transform of the third order cumulant sequence:

\[
B(f_1, f_2) = \sum_{\tau_1 = -\infty}^{+\infty} \sum_{\tau_2 = -\infty}^{+\infty} c_3(\tau_1, \tau_2)e^{-j2\pi(f_1 \tau_1 + f_2 \tau_2)} \tag{5.3}
\]

where the third order cumulant sequence is:

\[
c_3(\tau_1, \tau_2) = E[x(n).x(n + \tau_1).x(n + \tau_2)] \tag{5.4}
\]

A reader interested in an in-depth treatment of higher order spectral analysis is referred to the book on this topic by Nikias and Petropulu\[49\]. Different methods of estimating the bispectrum, indirectly from estimated triple correlations and directly from averaged products of Fourier coefficients, are explained. A good review article on the topic is \[50\].

Definitions of higher order spectra have been extended to deterministic signals where the expectation operation is replaced by a time average. In the deterministic framework, the bispectrum can be estimated (or even defined) as a triple product of Fourier coefficients,

\[
B(f_1, f_2) = X(f_1)X(f_2)X^*(f_1 + f_2) \tag{5.5}
\]

where \( X(f) \) is the discrete-time Fourier transform (DTFT) of \( x(n) \) at fre-
CHAPTER 5. ROBUST HASHING USING BISECTRA

quency $f$. Although frequencies $f_1$ and $f_2$ have a range $[-f_s/2, f_s/2]$ where $f_s$ is the sampling frequency, it need not be computed over this entire region in bi-

frequency space. The bispectrum of a real-valued input signal exhibits symmetry properties arising from the complex-conjugate symmetry of the Fourier spectrum of a real-valued input and the inter-changeability of the two frequency variables in the above equation. If the signal can be assumed band-limited to the Nyquist frequency, $f_s/2$, the non-redundant region of computation of the bispectrum re-

duces to a triangular region bounded by $f_1 > 0$, $f_1 > f_2$, and $f_1 + f_2 < f_s/2$. In this region, shown in Figure 5.1, the bispectrum is complex-valued and retains Fourier phase information. It has been shown that the original deterministic sig-

nal can be reconstructed from the bispectrum except for an unknown shift\[51\]. The bispectrum, like the power spectrum, is shift invariant.

![Figure 5.1: Region of computation of the bispectrum. Feature are obtained by integrating (summing) the bispectrum along radial lines of varying gradients $a$](image)

Chandran and Elgar [52] proposed a method of extracting features from the bispectrum that exhibit additional invariance to changes in time scale and amplit-

de scale. They compute phases of the integrated Bispectrum along radial lines of varying gradients $a_1, \ldots, a_n$ for $0 < a_n \leq 1$. These phases serve as a set of invariant features. The number of gradients used $(n)$ is equal to the length of the input signal $(x(n))$. The integrated bispectrum is represented as a complex-valued vector $V$ and each element is defined as:

$$V(n) = V^R(n) + V^I(n)i = \int_{f_1=0+}^{1/(1+a_n)} B(f_1, a_n f_1) df_1$$

(5.6)

where $V^R(n)$ and $V^I(n)$ are the real and imaginary components of $V(n)$ and
5.2 Higher Order Spectra

\[ i = \sqrt{-1} \]. The features, \( P(a) \) are given by

\[ P(a) = \arctan \left( \frac{V^3(n)}{V^\Re(n)} \right) \]  

(5.7)

This is known as the direct method of computing the bispectrum. Alternatively, the bispectrum can be computed from the power spectrum of the input signal and this is known as the indirect approach. The indirect method discards all the phase information of the original signal and the resulting features have improved scale invariance [53]. Figure 5.2 shows the steps of both the direct and indirect approach of bispectrum estimation. Since scale invariance can be trivially obtained with images by resizing all images to a predetermined size we use the direct method of bispectrum estimation in this thesis.

![Diagram of Direct and Indirect Methods](image)

Figure 5.2: Direct and indirect methods of computing invariant features from the bispectrum of a 1D input signal.

The bispectral invariant feature extraction can be applied to two-dimensional data such as images by first reducing the 2D input into a set one-dimensional components. In [53], the task of converting 2D images into 1D components is achieved using the Radon transform (also known as parallel beam projection).
5.3 The Radon Transform

If a 2D image of dimensions $X \times Y$ is denoted as a matrix of pixel values $I(n, m)$, where $n = 1, \ldots, X$ and $m = 1, \ldots, Y$, then the Radon transform is described as the integral of $I(n, m)$ along straight lines oriented at angle $\theta$ with offset $p$. The Radon projection of $I(n, m)$ at angle $\theta$ can be expressed as [54]:

$$x_{\theta}(p) = \int \int I(n, m) \delta(p - n \cos(\theta) - m \sin(\theta)) \, dn \, dm \quad (5.8)$$

The 2D image can be reduced to a set of $N$ 1D projections by applying $5.8$ using $N$ different angles $\theta$ equally spaced between 0 and $\pi$. Each 1D projection vector $(x_{\theta}(p))$ has a different length dependent on $\theta$. In other words, $p = 1, \ldots, P$ where $P = X$ (the horizontal width of the image) when $\theta = 0$, $P = Y$ when $\theta = 90$ and ranges up to a maximum length of $P = \sqrt{X^2 + Y^2}$ (the diagonal length of the image) when $\theta = 45$.

The value of $N$ is chosen to be equal to the length of the randomized feature vector, which is $N = 64$ in our experiments. An example is shown in Figure 5.3. It is possible to reconstruct the original image $I(n, m)$ from the set of Radon projections $x_{\theta}(p)$ using the inverse Radon transform. But, for purposes of Bispectral invariant feature extraction the Radon transform serves as a necessary dimensionality reduction that reduces a 2D input of size $X \times Y$ into a set of $N$ 1D vectors. For a digital (discrete) image the projection can be produced by rotating the image about its centre by $\theta$ degrees and summing pixel values along vertical beams.

This technique was first applied to 2D inputs in [53] and subsequently used for digit recognition [55], automatic sea mine detection [54], virus recognition [57], biometric template security [58] and biometric cryptography [59, 60]. In many of these applications the 2D inputs are binary image blocks containing the mask of the query object (see Figure 5.4). The Radon projections serve as 1D profiles of the object from which bispectral invariants are extracted. The bispectral invariants are robust to:

- Translation - Shifts the object within the image block resulting in a corresponding shift of the Radon projection.

- Scaling - Scaling of the object size results in either amplitude scaling, length scaling or a combination of the two in the Radon projections.
5.4 Normalisation of Radon Projections

Figure 5.3: Example of the Radon projection obtained from an image $I(n, m)$ at angle $\theta$.

- Rotation - Rotation of the object results in a circular shift of the Radon projections.

Feature extraction for image hashing is fundamentally different than feature extraction for object recognition and some of the robustness properties useful in recognition tasks are unnecessary in image hashing. For example, translation and scale invariance are necessary in object recognition because the object may be translated and scaled within the image block. In image hashing the entire image serves as the image block, it is therefore not possible for resulting Radon projections to be translated. Instead, if there is a translation of the scene depicted in the image the resulting Radon projections will be altered in a non-trivial manner.

5.4 Normalisation of Radon Projections

When the entire image is used as the image block the resulting Radon projections will have an underlying shape based on $\theta$. When $\theta$ is 0, 90, 180 or 270 the image is rotated at right angles and remains rectangular (or square). However when $\theta$ is not a multiple of 90 the image can appear diamond in shape with tapered ends. The resulting Radon projection will have similarly tapered ends and appear triangular. This underlying shape would be detrimental to bispectral analysis.
which is sensitive to signal shape. In Figure 5.5 the Lena image along with its Radon projection ($x_\theta(p)$) is shown for both $\theta = 0$ and $\theta = 45$.

It is easy to see that this triangular shape would be present in the Radon projection of any image rotated at $\theta = 45$. In order to remove this dependency of $x_\theta(p)$ on $\theta$ a normalization is applied. Each Radon projection $x_\theta(p)$ can be normalized by dividing with the Radon projection obtained from a same sized image made up entirely of ones. We denote the Radon projection of such an image as $o_\theta(p)$. The normalized vector can be expressed as:

$$x_\theta'(p) = \frac{x_\theta(p)}{o_\theta(p)} \quad (5.9)$$

Finally, the Radon projection is made zero-mean with maximum value of 1 by subtracting the mean and dividing by the largest value:

$$\hat{x}_\theta(p) = \frac{x_\theta'(p) - \bar{x}_\theta}{max(x_\theta'(p))} \quad (5.10)$$

An example of this normalization process is demonstrated on the Lena image.
5.5 Properties of Radon Projections and their Bispectral Invariants

In the field of image hashing a number of techniques have used Radon projection and these were discussed in Chapter 2. Of these methods, none have presented analysis into the robustness properties of Radon projections for common image transformations. Furthermore, HOS based techniques such as the bispectral invariants method have not be applied for image hashing despite a number of properties that make them suited for the task. Bispectral invariants can be used as a set of complex-valued projection basis that are a non-linear function of both the input image and secret-key. Such a randomization scheme would retain the benefits of complex projection (Chapter 4), be immune to linear inversion techniques described in Chapter 2 and produce randomized features that are a fusion of two types of image features. This is possible because bispectral invariants
CHAPTER 5. ROBUST HASHING USING BISPECTRA

(a) Lena image rotated using $\theta = 45$
(b) Image of ones rotated using $\theta = 45$

Figure 5.6: The Radon projection of the Lena image with $\theta = 45$ is normalized using the Radon projection of the ones image with $\theta = 45$.

extracted from the Radon projection of an image are:

1. Themselves image features that are robust to many common image transformations.

2. Sensitive to the shape of the Radon projection which is dependent on the input image but can also be altered by applying a secret-key dependent permutation.

These properties are analysed in detail in the following sections.

5.5.1 Robustness Analysis

In existing robust hashing algorithms the randomization is independent of the input image. For example, in random projection the projection basis are derived solely from the secret-key. They are therefore deterministic and independent of the input image. The purpose of this chapter is to investigate the use of bispectral invariants as an image-dependent projection basis. Equation 2.3 shows
5.5 Properties of Radon Projections and their Bispectral Invariants

The randomized features are a combination of the image features and projection basis. In order for randomized features to be consistent the bispectral invariants must be robust to common image transformations so as to produce similar basis for transformed versions of the same image. This section analyses the robustness properties of Radon projections and the corresponding bispectral invariants to common perceptual content preserving image transformations. Related analysis can also be obtained from [50] and [49].

Amplitude Transformations

It is proved in [52] that phases of the integrated bispectrum along radial slices are invariant to amplitude scaling. Amplitude change in a two dimensional image result in corresponding amplitude change in the Radon projections. Therefore, the feature values extracted by the proposed method are robust to these content preserving transformation even in a 2D image.

Amplitude changes in images are usually due to brightness adjustments. A simple normalisation or the Radon projection, by making projections zero mean and unit $L_\infty$ norm, can help make bispectral invariant features robust such adjustments. This robustness is illustrated in figure 5.7 where the Lena image has been brightened by adding a constant to each pixel value. The figures depict the original and modified Lena images, their corresponding $\theta = 0$ Radon projection, along with the real and imaginary components ($V^R(n)$ and $V^I(n)$) of the complex bispectral invariants extracted from the Radon projection. The projection and invariants extracted from the Mandrill image are also shown for reference. This type of brightness adjustment manifests as an offset of the Radon projections and Bispectral invariants are robust to offset changes on the input signal.

Another common brightness adjustment is contrast stretching, this is a linear process achieved by the equation:

$$Q = (P - P_{\text{min}}) \times \frac{255}{P_{\text{max}} - P_{\text{min}}} \quad (5.11)$$

where $Q$ is the new pixel value, $P$ the current pixel value and $P_{\text{min}}, P_{\text{max}}$ denotes the minimum and maximum pixel values present in the image. From equation (5.11) it is easy to see contrast stretching involves an offset followed by a scaling. Hence the normalised Radon projections would be immune to such changes as shown in Figure 5.8.
CHAPTER 5. ROBUST HASHING USING BISPECTRA

More complicated contrast adjustments such as histogram equalisation, gamma transforms and piecewise contrast stretching involve a non-linear mapping of pixel values, the effects of non-linear transformations are not so easily modeled. Figure 5.9 shows the effects of histogram equalisation on Radon projections and bispectral invariants, although the transformation is non-linear its effects on the Radon projections and resulting bispectral invariants are minimal.

It can be seen from Figures 5.7, 5.8 and 5.9 that both the Radon projections and bispectral invariants are robust to common brightness and contrast adjustments of the image.

Geometric Transformations

Bispectral invariant features are invariant to translation and scaling of 1D input signals. In an image hashing context, translation invariance is not considered and scaling of images can be overcome by resizing images to a fixed size. Image resizing involves the use of interpolation and this introduces interpolation error. As a result, two identical images of different scale can be normalised to be the
5.5 Properties of Radon Projections and their Bispectral Invariants

![Radon Projections and Bispectral Invariants](image)

(a) Original  (b) Contrast Stretched

(c) Left - Radon projection at 0 degrees, Middle - Real component of $V$, Right - imaginary component of $V$

Figure 5.8: Behaviour of Radon projections and the integrated bispectrum ($V$) to Contrast Stretching

same size (in terms of pixel dimensions) but will not have the same pixel values. This changes a scaling transformation into noise, where the noise component is the interpolation error.

Figure 5.10 shows the effects of image noise due to errors introduced by bilinear interpolation. The resulting image looks noticeably smoothed, this is because the process of bilinear interpolation uses spatial averaging. This is can be seen by comparing Figure 5.10 to Figure 5.11 which depict the effects of mean filtering with a $9 \times 9$ kernel. Both figures show that bispectral invariant features are highly robust to both image resizing and mean filtering.

Rotations of the two-dimensional image result in a Radon projection moving to a new angle of projection or a cyclic shift in the angle axis [53] by $\phi$ where $\phi$ is the angle of rotation. For robust hashing, large rotations are not considered because when $\phi > 10^\circ$ large portions of the resulting image will exceed the original image boundaries. This can be addressed by cropping the image to its original size [4] but doing so causes significant perceptual changes and therefore only minor rotations that preserve perceptual quality ($\phi < 5^\circ$) are considered. A rotation of
Figure 5.9: Behaviour of Radon projections and the integrated bispectrum ($V$) to histogram equalisation

the image can be formally described by letting $x(a)$ denote the Radon projections obtained from the input image where $a$ is the angle of projection and $x'(a)$ denote the set of Radon projections obtained from a rotated version of the original image. It is easy to see that

$$x'(a) = x(a + \theta)$$

and $x(a) \approx x(a + \phi)$ if $\phi$ is small

then $x(a) \approx x'(a)$

Figure 5.12 shows the effects of image rotation where $\phi = 2^\circ$. The rotated image is also cropped to remove black regions (due to rotation) and resized to original size.

These examples show the robustness of Bispectral invariant features to common perceptual content preserving geometric transformations.
5.5 Properties of Radon Projections and their Bispectral Invariants

(a) Original  (b) Scaled

(c) Left - Radon projection at 0 degrees, Middle - Real component of V, Right - imaginary component of V

Figure 5.10: Behaviour of Radon projections and the integrated bispectrum (V) to image scaling where image down sampled by factor of 2 then scaled up to original size (using Bilinear interpolation)

Image Noise

Uncorrelated noise in image pixel values ends up being nearly Gaussian distributed in the Radon projections and bispectra are robust to additive Gaussian noise. This is analytically shown here.

Representing an image (I) of dimensions $X \times Y$ as a matrix of pixel values $P$:

$$I = [P_{n,m}]_{X \times Y} \quad (5.12)$$

Additive noise results in the transformed image $I'$ effected with pixel errors denoted as $\epsilon$, thus:

$$I' = I + \epsilon$$

$$= [P_{n,m}]_{X \times Y} + [\epsilon_{n,m}]_{X \times Y} \quad (5.13)$$

The Radon projection is the column summation of the image $I$ after rotation at its centre by $\theta$ degrees. For simplicity it is assumed that $\theta = 0^\circ$ but the analysis
discussed here easily extends to all values of \( \theta \). The Radon projection \( x_\theta \) of image \( I \) becomes:

\[
x_\theta (n) = \sum_{m=1}^{Y} [P_{n,m}]_{X \times Y} \text{ where } n = 1, 2, \ldots, X
\]  

(5.14)

The Radon projection of the transformed image \( I' \) becomes \( x'_\theta \):

\[
x'_\theta (n) = \sum_{m=1}^{Y} ([P_{n,m}]_{X \times Y} + [\epsilon_{n,m}]_{X \times Y})
\]

\[
= \sum_{m=1}^{Y} [P_{n,m}]_{X \times Y} + \sum_{m=1}^{Y} [\epsilon_{n,m}]_{X \times Y}
\]

(5.15)

\[
= x_\theta (n) + \sum_{m=1}^{Y} [\epsilon_{n,m}]_{X \times Y}
\]

A Radon projection of the transformed image is equal to the Radon projection of the original image plus the summation of the error component \( \epsilon_{n,m} \). Let this
5.5 Properties of Radon Projections and their Bispectral Invariants

Figure 5.12: Behaviour of Radon projections and the integrated bispectrum (V) to image rotation of 2 degrees

column sum of $\epsilon_{n,m}$ be the projection error vector ($\epsilon$) where each element:

$$\epsilon (n) = \sum_{m=1}^{Y} [\epsilon_{n,m}]_{X \times Y}$$  \hspace{1cm} (5.16)

For an image affected by noise, the error elements ($\epsilon_{n,m}$) are randomly drawn from a noise distribution. For any noise distribution whose elements are i.i.d the resulting vector $\epsilon$ will tend towards Gaussian due to central limit theorem.

If the noise probability density distribution is $\epsilon \sim N(0, \sigma^2)$, then the Radon projection of an image effected by such noise is:

$$x'_\theta (n) = x_\theta (n) + N(0, \sigma^2)$$  \hspace{1cm} (5.17)

Bispectral invariant features derived from $x_\theta$ and $x'_\theta$ will be approximately equal due to the Gaussian noise immunity property of the bispectrum. This immunity holds only when bispectra are averaged but the integration step in the computation of the invariants serves to satisfy this condition even though the
input is deterministic.

If the noise is signal-dependent or if noise is multiplicative rather than additive, the bispectrum will change. For example, for salt and pepper noise where \( \epsilon_{n,m} \) is a function of the original pixel value \( P_{n,m} \). Most pixels are unchanged for salt and pepper noise and thus the majority of \( \epsilon_{n,m} = 0 \) but for pixels affected by pepper noise \( \epsilon_{n,m} = P_{n,m} \) while for salt noise \( \epsilon_{n,m} = 1 - P_{n,m} \), where the minimum and maximum allowable pixel value are normalized to 0 and 1.

![Original and Gaussian Noise](image)

(a) Original (b) Gaussian Noise

![Graphs](image)

(c) Left - Radon projection at 0 degrees, Middle - Real component of \( V \), Right - imaginary component of \( V \)

Figure 5.13: Behaviour of Radon projections and the integrated bispectrum (\( V \)) to Gaussian noise

Image compression algorithms such as JPEG often discard high frequency information even though the image content is not disturbed perceptibly. Compression algorithms usually work on image blocks and therefore compression error is dependent on the pixels in the region. Consequently an analytical treatment of changes in the bispectral invariants is difficult. Furthermore, the robustness analysis so far has involved only a single image, the next section carries out experimental measurements of robustness and fragility for a large image set.
5.5 Properties of Radon Projections and their Bispectral Invariants

![Images of original and JPEG compressed images]

(a) Original  (b) JPEG compressed

(c) Left - Radon projection at 0 degrees, Middle - Real component of \( V \), Right - imaginary component of \( V \)

Figure 5.14: Behaviour of Radon projections and the integrated bispectrum (\( V \)) to JPEG compression at 10% Quality

Experimental Results

The MIRFLICKR database is used to experimentally measure the projection error \( \epsilon \) for 25,000 images under different image transformations. Radon projections of the \( c \)-th image \( (I_c) \) at \( \theta = 0^\circ \) to obtain \( x_c \) are used. Different content preserving transformations (listed in Table 5.1) are applied to each image to produce the modified image \( (I'_c) \) from which \( x'_c \) is obtained. It can be seen from (5.15) \( \epsilon_c = x_c - x'_c \) and this is used to obtain the Euclidean distance between two real-valued Radon projections:

\[
d_E(x_c, x'_c) = \sqrt{\sum_{n=1}^{X} [x_c(n) - x'_c(n)]^2} = \sqrt{\sum_{n=1}^{X} [\epsilon_c(n)]^2} \tag{5.18}
\]

The robustness of Radon projections is the average Euclidean distance between \( x_c \) and \( x'_c \), we let \( C = 25,000 \) the total number of images in the dataset:

\[
\text{mean} [d_E(x_c, x'_c)] = \frac{1}{C} \sum_{c=1}^{C} d_E(x_c, x'_c)
\]
CHAPTER 5. ROBUST HASHING USING BISPECTRA

Bispectral invariants $V_c$ and $V'_c$ are then computed from $x_c$ and $x'_c$. Since $V$ is a complex vector its robustness is measured by comparing the average Euclidean distance of the real ($V^R$) and imaginary ($V^I$) components separately using

$$mean \left[ d_E \left( V^R_c, V'^R_c \right) \right] = \frac{1}{C} \sum_{c=1}^{C} d_E \left( V^R_c, V'^R_c \right)$$

and

$$mean \left[ d_E \left( V^I_c, V'^I_c \right) \right] = \frac{1}{C} \sum_{c=1}^{C} d_E \left( V^I_c, V'^I_c \right)$$

The fragility of $x$ is the average Euclidean distance between Radon projection vectors obtained from two different images $x_c$ and $x_d$ and the fragility of $V$ is the average Euclidean distance between the real and imaginary components. The image dataset contains $C = 25,000$ images and the commutative nature of the distance metrics used means there will be $\frac{C^2 - C}{2}$ unique pairwise distances.

$$mean \left[ d_E \left( x_c, x_d \right) \right] = \frac{2}{C^2 - C} \sum_{c=1}^{C-1} \sum_{d=c+1}^{C} d_E \left( x_c, x_d \right)$$

$$mean \left[ d_E \left( V^R_c, V^R_d \right) \right] = \frac{2}{C^2 - C} \sum_{c=1}^{C-1} \sum_{d=c+1}^{C} d_E \left( V^R_c, V^R_d \right)$$

$$mean \left[ d_E \left( V^I_c, V^I_d \right) \right] = \frac{2}{C^2 - C} \sum_{c=1}^{C-1} \sum_{d=c+1}^{C} d_E \left( V^I_c, V^I_d \right)$$

Table 5.1 shows the average difference between transformed versions of the same image (intra-class difference) for both Radon projections and the corresponding bispectral invariants. Table 5.2 shows the fragility (or average inter-class difference) for Radon projections and bispectral invariants. For most transformations the average intra-class difference in Table 5.1 is at least an order of magnitude smaller than the average inter-class distance in Table 5.2. These results lend support to the use of bispectral invariants for robust hashing randomization. The bispectral invariants are robust to common image transformations meaning they can serve as a reliable set of basis for a given image even in the presence of modification. They also display fragility and this allows them to produce a different set of basis for different images.
Table 5.1: Robustness of the Radon projections $\mathbf{x}$ and their derived HOS invariants $\mathbf{V}$ for common image transformations. Both the mean and standard deviation is shown.

<table>
<thead>
<tr>
<th>Image Transformations</th>
<th>Robustness Measures</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$d_E (\mathbf{x}_c, \mathbf{x}'_c)$</td>
<td>$d_E (\mathbf{V}_c^R, \mathbf{V}'_c^R) \times 10^6$</td>
</tr>
<tr>
<td></td>
<td>mean</td>
<td>std</td>
</tr>
<tr>
<td>1. JPEG 10% Quality</td>
<td>0.452</td>
<td>0.274</td>
</tr>
<tr>
<td>2. Mean filtering (15x15)</td>
<td>1.695</td>
<td>1.291</td>
</tr>
<tr>
<td>3. Median filtering (15x15)</td>
<td>1.681</td>
<td>1.374</td>
</tr>
<tr>
<td>4. Gaussian noise ($\sigma = 0.05$)</td>
<td>1.44</td>
<td>0.799</td>
</tr>
<tr>
<td>5. Salt and pepper noise (10%)</td>
<td>1.2864</td>
<td>0.732</td>
</tr>
<tr>
<td>6. Cropping by 5 pixels</td>
<td>1.962</td>
<td>1.321</td>
</tr>
<tr>
<td>7. Rotation by 4 degrees</td>
<td>3.652</td>
<td>1.620</td>
</tr>
<tr>
<td>8. Contrast stretching ($\times 10^{-14}$)</td>
<td>2.747</td>
<td>6.692</td>
</tr>
<tr>
<td>9. Histogram equalisation</td>
<td>2.511</td>
<td>1.774</td>
</tr>
<tr>
<td>10. Sharpening</td>
<td>1.335</td>
<td>0.841</td>
</tr>
<tr>
<td>Combination of 1, 2, 4, 6, 7 and 9</td>
<td>4.822</td>
<td>1.767</td>
</tr>
</tbody>
</table>
### 5.5.2 Sensitivity Analysis

The unpredictability property of robust hashing was introduced in Chapter 2. This property relates to the sensitivity of robust hash outputs to changes in the secret-key. Secret-key dependence is a crucial part of robust hashing security because it ensures the correct hash cannot be obtained without knowledge of the correct secret-key. However, Bispectral invariants are not designed to function with a secret-key and this feature must be specifically incorporated. This can be achieved by exploiting the sensitivity of Bispectral invariants to the shape of the input signal.

One way to alter signal shape is to apply a random permutation to the Radon projections using the secret-key as the seed to the PRNG. This makes the permutation repeatable and deterministic since the sequence of random numbers generated by a PRNG is a deterministic function of the seed. But at the same time the permutation is secure because it cannot be obtained without the correct secret-key.

Equation 5.5 shows the Bispectrum is a function of Fourier coefficients obtained from the DTFT. Altering the shape of a signal will change its frequency profile. Furthermore, the change in frequency is a function of both the permutation and the input signal itself. In other words, the same permutation applied to two different signals will not produce the same frequency domain changes. This concept is illustrated using the $\theta = 0$ Radon projection of the Lena and Mandrill images. Three different permutations are applied to the $\theta = 0$ Radon projection of each image (see Figure 5.15). The corresponding DTFT’s can be compared in Figure 5.16. Since the frequency domain is symmetric around the 0-Hz component only the positive half of the frequency domain is shown. The real and imaginary components of the Bispectral invariants extracted from the three permuted Radon projections are shown in Figures 5.17 and 5.18.

These figures illustrate the sensitivity of Bispectral invariants to permutations.
5.5 Properties of Radon Projections and their Bispectral Invariants

Figure 5.15: The $\theta = 0$ Radon Projection of the Lena (left) and Mandrill (right) images for three different permutations.
Figure 5.16: The positive half of the Fourier domain obtained from the DTFT of the $\theta = 0$ Radon Projection of the Lena (left) and Mandrill (right) images for three different permutations.
5.5 Properties of Radon Projections and their Bispectral Invariants

Figure 5.17: The real component of the Bispectral invariants derived from the \( \theta = 0 \) Radon Projection of the Lena (left) and Mandrill (right) images for three different permutations.
CHAPTER 5. ROBUST HASHING USING BISPECTRA

Figure 5.18: The imaginary component of the Bispectral invariants derived from the $\theta = 0$ Radon Projection of the Lena (left) and Mandrill (right) images for three different permutations.
in the input signal and this indicates that Bispectral invariants could satisfy the unpredictability property of robust hashing. The non-linear relationship between signal shape and frequency makes it difficult if not impossible to build a theoretic model that describes how Bispectral invariants vary with permutations of the input signal. Instead an experimental approach can be used to quantify unpredictability by calculating the average euclidean distance between Radon projections and the real and imaginary components of Bispectral invariants produced for a constant image but varying secret-keys.

These experiments are conducted with image 1 of the MIRFLICKR database for 25,000 different secret-keys. In this section, $\tilde{x}_k$ is the normalized Radon projection of the image for $\theta = 0$ that has been permuted with the $k$-th secret-key and $V^R_k$, $V^I_k$ are used to denote the real and imaginary components of the complex bispectrum obtained from $\tilde{x}_k$ where $k = 1, 2, \ldots, 25,000$. For simplicity the $k$-th secret-key is equal to $k$ that is $K_k = k$. These distances can be obtained with equations similar to those in 5.19 to 5.21 we let $C = 25,000$:

$$\text{mean} \left[ d_E (\tilde{x}_k, \tilde{x}_l) \right] = \frac{2}{C^2 - C} \sum_{k=1}^{C-1} \sum_{l=k+1}^{C} d_E (\tilde{x}_k, \tilde{x}_l) \quad (5.22)$$

$$\text{mean} \left[ d_E \left( V^R_k, V^R_l \right) \right] = \frac{2}{C^2 - C} \sum_{k=1}^{C-1} \sum_{l=k+1}^{C} d_E \left( V^R_k, V^R_l \right) \quad (5.23)$$

$$\text{mean} \left[ d_E \left( V^I_k, V^I_l \right) \right] = \frac{2}{C^2 - C} \sum_{k=1}^{C-1} \sum_{l=k+1}^{C} d_E \left( V^I_k, V^I_l \right) \quad (5.24)$$

The experimentally measured unpredictability is shown in table 5.3.

<table>
<thead>
<tr>
<th>Unpredictability</th>
<th>$\text{mean} \left[ d_E (\tilde{x}_k, \tilde{x}_l) \right]$</th>
<th>$\text{mean} \left[ d_E \left( V^R_k, V^R_l \right) \right] \times 10^6$</th>
<th>$\text{mean} \left[ d_E \left( V^I_k, V^I_l \right) \right] \times 10^6$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$14.03$</td>
<td>$1.159$</td>
<td>$0.962$</td>
<td></td>
</tr>
<tr>
<td>$0.314$</td>
<td>$0.092$</td>
<td>$0.273$</td>
<td></td>
</tr>
</tbody>
</table>

The robustness, fragility and unpredictability results in Tables 5.1, 5.2 and 5.3 are graphically summarised using box-and-whisker plots. In this representation the box represents the central 50% of the distribution with the red line denoting the median value. The upper and lower whiskers represent the highest and lowest
value which fall within a tolerance bound of this central region. Outlier values beyond this range are shown with a red dot. Assuming a normal distribution, the range between the upper and lower whisker covers more than 99.3% of the data.

Figure 5.19 shows the box-and-whisker plots for Radon projections, Figure 5.20 for real component of the integrated bispectrum and Figure 5.21 shows the imaginary component of the integrated bispectrum. The spread of intra-class Euclidean distances for different image transformations (robustness measures) are shown beside the inter-class Euclidean distances (fragility) and the inter-key Euclidean distances (unpredictability).

![Box Plot](image)

**Figure 5.19:** Distribution of intra-class Euclidean distances for Radon projections ($d_E(\tilde{x}_k, \tilde{x}_l)$) given different image transformations. The fragility and unpredictability is also shown beside for reference.

From the presented results it can be seen that the Radon projections and their integrated bispectrum display reasonable separation between intra-class distances and inter-class distances for most image transformations. The exception being the combination of transforms which combines JPEG compression, mean filtering, Gaussian noise, cropping, rotation and histogram equalisation. This is the hardest case and even so the overlap in distributions is primarily in the whiskers a single threshold could be set to accurately separate around 75% of the data. This lends support to the use of Radon projections and their integrated bispectrum as suitable features for robust image hashing. Furthermore, the average distance for
5.5 Properties of Radon Projections and their Bispectral Invariants

Figure 5.20: Distribution of intra-class Euclidean distances for the real component of the integrated bispectrum ($d_E \left( V^R_k, V^R_l \right)$) given different image transformations. The fragility and unpredictability is also shown beside for reference.

Figure 5.21: Distribution of intra-class Euclidean distances for the imaginary component of the integrated bispectrum ($d_E \left( V^3_k, V^3_l \right)$) given different image transformations. The fragility and unpredictability is also shown beside for reference.
CHAPTER 5. ROBUST HASHING USING BISPECTRA

a fixed image using different secret-keys (the unpredictability measure) is similar to the inter-class distance obtained from separate images (the fragility measure). This indicates the extracted feature are sufficiently sensitive to changes in the secret-key and is another desirable property for robust hashing.

5.6 Proposed Scheme

The proposed randomization scheme generates a projection basis by extracting bispectral invariants from the permuted Radon projections of the input image. These serve as a set of complex, non-linear projection basis that is dependent on both the image and secret-key. The real-valued image feature vector is projected with this basis to produce a complex-valued output of which the angle component is retained to form the randomized features. The benefits of using a complex projection basis was discussed in Chapter 4.

The steps of proposed robust hashing algorithm are:

1. Extract features $\mathbf{F}$ from the input image of size $X \times Y$ pixels. $\mathbf{F}$ is a length $N$ vector or real numbers.

2. Produce the image dependent Projection basis by:

   (a) Applying the Radon transform to the input image at $M$ equally spaced angles ($\theta$) in the range $(0, \pi]$, i.e. $\theta_r = \frac{(r-1)\pi}{M}$ where $r = 1, 2, \ldots, M$ to obtain the set of Radon projection vectors $\mathbf{x}_r$. The length of each $\mathbf{x}_r$ is denoted as $P$, and varies as a function of $\theta_r$ from $\min(X, Y)$ to $\sqrt{X^2 + Y^2}$. The value $M$ is equal to the length of the desired final randomized feature vector, in order to satisfy compression $M$ is usually less than $N$. In our experiments $M = 64$.

   (b) Obtain the normalized vectors $\hat{\mathbf{x}}_r$ for each of the $M$ projection vectors using the method described in section [5.4]

   (c) Randomly permute each of the $M$ normalized vectors ($\hat{\mathbf{x}}_r$) using $K$ as the seed to a PRNG to obtain $\tilde{\mathbf{x}}_r$.

   (d) Take the $4N$ point discrete-time Fourier transform (DTFT) of each $\tilde{\mathbf{x}}_r$ and compute the bispectrum ($B_r(f_1, f_2)$) using (5.5). The input $\tilde{\mathbf{x}}_r$ is zero-padded to length $4N$ in order to improve spectral interpolation.
(e) Obtain the complex-valued integrated Bispectrum $\mathbf{V}_r$ from $B_r(f_1, f_2)$ using (5.6). The gradients $a_n$ is evenly spaced between 0 and 1. The number of gradients ($n$) is chosen to be equal to the length of the real-valued feature vector $\mathbf{F}$ that is to be projected ($n = N$). This is important because it ensures the integrated bispectrum (projection basis) will be same length as the real-valued feature vector $\mathbf{F}$. The length of $\mathbf{F}$ and $\mathbf{V}$ must be the same so projection (using the inner product) can be performed.

3. Compute the length $M$ randomized feature vector ($\mathbf{R} = [\phi_1, \phi_2, \ldots, \phi_M]$) where each hash element ($\phi_r$) is the argument (angle) of the inner product between the real-valued feature vector $\mathbf{F}$ and the $r$-th integrated Bispectrum $\mathbf{V}_r$ where $r = 1, 2, \ldots, M$:

$$
\mathbf{R}(r) = \phi_r = \arctan\left( \frac{\Re(\langle \mathbf{F}, \mathbf{V}_r \rangle)}{\Im(\langle \mathbf{F}, \mathbf{V}_r \rangle)} \right)
$$

(5.25)

Because $\mathbf{V}_r$ is complex (see (5.6)) the resulting inner product is also complex and can be converted to polar form to extract the argument $\phi_r$.

The method described above allows the bispectral invariant randomization scheme to be applied to any type of image features. The original input feature vector is of length $N$ and the randomized feature vector $\mathbf{R}$ is length $M$. Therefore, $M$ must be smaller than $N$ in order to compress the feature vector. In the remainder of the dissertation this scheme is referred to as Generic-HOS where Generic can be replaced with the name of a feature extraction process. For example, the Generic-HOS randomization applied to NMF image features is called NMF-HOS.

The robustness analysis carried out in the previous section has shown Radon projections to be good image features in their own right and it is possible to apply Bispectral invariant randomization to Radon projections directly. There are a number of advantages to doing so:

1. Radon projections are used to produce bispectral invariants so there is no need to compute additional image features.

2. A set of $M$ Radon projections preserve more image information than other image features (see Figure 5.22).
3. When using image features such as NMF or Fourier-Mellin, the same feature vector \( \mathbf{F} \) is projected with each of the \( M \) different basis. With Radon projections each \( M \) different vector can be projected using its corresponding basis and thus the extra information contained in Radon projections can be preserved in the randomized feature vector \( \mathbf{R} \).

![Figure 5.22: Reconstructions of the original Lena image using different types of image features.](image)

Using bispectral invariants to project the Radon projections from which they are derived is called Radon-HOS and can be achieved by:

1. Apply the Radon transform to the input image at \( M \) equally spaced angles \( \theta \) in the range \( (0, \pi] \), i.e. \( \theta_r = \frac{(r-1)\pi}{M} \) where \( r = 1, 2, \ldots, M \) to obtain the set of Radon projection vectors \( \mathbf{x}_r \). Normalize and permute to obtain \( \tilde{\mathbf{x}}_r \).
2. Produce the image dependent Projection basis (the steps are the same as above and are omitted here). Although in step (d) a $4P$ point DTFT is taken instead of a $4N$ point DTFT. This is because $P$ is the length of the $r$-th Radon projection and in the Radon-HOS method it is the Radon projections themselves that are projected with the bispectral invariants.

3. Compute the length $M$ randomized feature vector ($R = [\phi_1, \phi_2, \ldots, \phi_M]$) where each hash element ($\phi_r$) is the argument (angle) of the inner product between the $r$-th Radon projection vector $\tilde{x}_r$ with the $r$-th integrated Bispectrum $V_r$ where $r = 1, 2, \ldots, M$:

$$R(r) = \phi_r = \arctan \left( \frac{\Re(\langle \tilde{x}_r, V_r \rangle)}{\Im(\langle \tilde{x}_r, V_r \rangle)} \right)$$  

(5.26)

The method described here is slightly modified from that published in [61], the main difference being the use of more Radon projection angles which retain more image information.

5.6.1 Security Analysis

Having defined the steps of the Radon-HOS algorithm it is now important to consider the security advantages of such a technique. To do this a comparison of Randomized features generated by the proposed method are compared with those obtained from Random projections. From equation (2.3) it is evident that each hash element $R(r)$ of a random projection based hash is:

$$R_{RP}(r) = \sum_{i=1}^{n} F_{i,j} B_{i,r}$$  

(5.27)

Each hash element is a linear combination of an image feature component and a secret-key component. From the summation in equation (5.27) it follows that owing to the central limit theorem, the hash values will tend to being normal distributed if $F$ is i.i.d. The identical distribution condition can be relaxed. Recall from Chapter 2 for random projection based methods $F$ and $B$ are independent and $B$ is i.i.d by design. The normal distribution and statistical independence properties are good from the perspective of increased entropy and resilience to attacks. However, a linear combination using the summation of independent products allows either $F$ or $B$ to be reconstructed by solving a system of equations.
CHAPTER 5. ROBUST HASHING USING BISPECTRA

created using known image/hash pairs when the other is known.

For Generic-HOS, the image feature vector \( \mathbf{F} = [F(1), F(2), \ldots, F(N)] \) and
\( \mathbf{V}_r = \{V_r^{\Re}(1) + V_r^{\Im}(1)i\}, \ldots, \{V_r^{\Re}(n) + V_r^{\Im}(n)i\} \) then each hash element is the
inner product of \( \mathbf{F} \) and \( \mathbf{V}_r \)

\[
\mathbf{R}(r) = \langle \mathbf{F}, \mathbf{V}_r \rangle = \sum_{N} \left[ F(N). \{V_r^{\Re}(N) + V_r^{\Im}(N)i\} \right]
\]
\[
= \sum_{N} \left[ F(N).V_r^{\Re}(N) \right] + \left( \sum_{N} \left[ F(N).V_r^{\Im}(N) \right] \right) i \quad (5.28)
\]

Substituting (5.28) into (5.25):

\[
\mathbf{R}(r) = \phi_r = \arctan \left( \frac{\sum_{N} \left[ F(N).V_r^{\Re}(N) \right]}{\sum_{N} \left[ F(N).V_r^{\Im}(N) \right]} \right) \quad (5.29)
\]

While in the Radon-HOS method, if \( \tilde{\mathbf{x}}_r = [\tilde{x}_r(1), \tilde{x}_r(2), \ldots, \tilde{x}_r(n)] \) and
\( \mathbf{V}_r = \{V_r^{\Re}(1) + V_r^{\Im}(1)i\}, \ldots, \{V_r^{\Re}(n) + V_r^{\Im}(n)i\} \) then each hash element is the inner
product of \( \tilde{\mathbf{x}}_r \) and \( \mathbf{V}_r \)

\[
\mathbf{R}(r) = \langle \tilde{\mathbf{x}}_r, \mathbf{V}_r \rangle = \sum_{n} \left[ \tilde{x}_r(n). \{V_r^{\Re}(n) + V_r^{\Im}(n)i\} \right]
\]
\[
= \sum_{n} \left[ \tilde{x}_r(n).V_r^{\Re}(n) \right] + \left( \sum_{n} \left[ \tilde{x}_r(n).V_r^{\Im}(n) \right] \right) i \quad (5.30)
\]

Substituting (5.30) into (5.25):

\[
\mathbf{R}(r) = \phi_r = \arctan \left( \frac{\sum_{n} \left[ \tilde{x}_r(n).V_r^{\Re}(n) \right]}{\sum_{n} \left[ \tilde{x}_r(n).V_r^{\Im}(n) \right]} \right) \quad (5.31)
\]

From (5.29) and (5.31) the image features (or the Radon projection values)
are mixed in two non-linear processes. Firstly, \( \mathbf{V}_r(n) \) is the complex integrated
bispectrum of the permuted Radon projection \( \tilde{\mathbf{x}}_\theta(n) \). The bispectrum is a non-
linear function of the input signal and is therefore a non-linear combination of
both the Radon projections (\( \mathbf{x}_r(n) \)) and the secret-key which determines the per-
mutation. The variables in (5.29) and (5.31) will be different for each image/hash
5.7 Experimental Results

pair even if the secret-key remains unchanged. This prevents the use of Unicity
distance attacks since no set of linear equations can be created.

Secondly, the non-linear arctan function is applied to the dot product of $F(N)$ and $V_r(N)$ in the Generic-HOS method or between $\tilde{x}_r(n)$ and $V_r(n)$ in the Radon-HOS approach. This produces angle hash feature $\phi_r$. This has the
added benefit of normalising and limiting the range of output hash values re-
gardless of the secret-key, input feature size, scale or shape. The final hash will
be between $-\pi$ and $\pi$ and the benefits of bounded angle features were previously
presented in Chapter 4.

5.7 Experimental Results

The analysis in the previous section is validated by comparing hashing fragility,
entropy and accuracy for the different image transformations described in table
5.1 using 1000 images from the MIRFLICKR database. The two baseline sys-
tems described in Chapter 2 are used to benchmark performance. The first uses
Fourier-Mellin features and random projection (scheme 2 in [4]) and the second
uses NMF features and random projection [3]. These techniques are referred to as
FM-RP and NMF-RP. In order to highlight the advantages of Bispectral invari-
ant projection over random projection, these baselines are modified by replacing
random projection with Generic-HOS projection. These are referred as FM-HOS
and NMF-HOS. The Radon-HOS method described in the previous section is also
implemented.

The benefits of HOS projections in quantization are shown by performing tests
for two different quantizer training scenarios. In Scenario 1, the quantizer is
trained using a subset of 1000 images from the MIRFLICKR database and a the
secret-key $K_1$. This is identical to Scenario 1 described in Chapter 3. In the
second scenario the quantizer is trained using a different subset of 1000 images
from the MIRFLICKR database and secret-key $K_2$. This is slightly different to
Scenario 5 in Chapter 3 where multiple secret-keys are used. This scenario is
referred to in the remainder of the dissertation as Scenario 6. For both scenarios
the system is tested using the first 1000 images and secret-key $K_1$. In summary,
Scenario 1 trains the quantizer and tests performance using the same data, while
the second scenario tests performance using unseen data. For both scenarios
Adaptive deterministic and Uniform quantization schemes are used.
CHAPTER 5. ROBUST HASHING USING BISPECTRA

The fragility of hashes measured across the test data is presented in table 5.4. For Quantizer Training Scenario 1, we see that adaptive deterministic quantization achieves the theoretical maximum fragility of 0.5 [15] regardless of hashing technique. This result is not surprising considering that quantizer training scenario 1 uses the same data to test and train the hashing process.

Table 5.4: Comparison of hashing Fragility for the different hashing techniques under different Quantizer Training and Quantization Schemes

<table>
<thead>
<tr>
<th>Hashing Technique</th>
<th>Quantizer Training Scenario 1</th>
<th>Quantizer Training Scenario 6</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Adaptive</td>
<td>Uniform</td>
</tr>
<tr>
<td>NMF-RP</td>
<td>2-bit</td>
<td>0.500</td>
</tr>
<tr>
<td></td>
<td>4-bit</td>
<td>0.500</td>
</tr>
<tr>
<td>NMF-HOS</td>
<td>2-bit</td>
<td>0.500</td>
</tr>
<tr>
<td></td>
<td>4-bit</td>
<td>0.500</td>
</tr>
<tr>
<td>FM-RP</td>
<td>2-bit</td>
<td>0.500</td>
</tr>
<tr>
<td></td>
<td>4-bit</td>
<td>0.500</td>
</tr>
<tr>
<td>FM-HOS</td>
<td>2-bit</td>
<td>0.500</td>
</tr>
<tr>
<td></td>
<td>4-bit</td>
<td>0.500</td>
</tr>
<tr>
<td>Radon-HOS</td>
<td>2-bit</td>
<td>0.500</td>
</tr>
<tr>
<td></td>
<td>4-bit</td>
<td>0.500</td>
</tr>
</tbody>
</table>

The results show uniform quantization to be inferior to adaptive deterministic quantization for quantizer training scenario 1. This is a result previously obtained by Zhu [15]. However, the results for quantizer training scenario 6 clearly illustrates the sensitivity of random projection based methods to quantizer training and this reinforces the results from Chapter 3. Due to this sensitivity it is not guaranteed that adaptive quantization will outperform uniform quantization.

The HOS based methods are less sensitive to quantizer training with fragility remaining consistently high for all quantizer schemes and training scenarios. According to Chapter 4, uniform quantization for bounded features is the equivalent of no training. This is because the minimum and maximum feature values are known a priori. In the case of angle features they are $-\pi$ and $\pi$ respectively and thus quantization thresholds can be obtained without using training samples. Quantization thresholds are therefore independent of image data and secret-keys and this overcomes the need for quantizer training data management discussed in Chapter 3.

Hashing accuracy is measured using ROC curves shown in figures 5.24-5.27. These curves are obtained by measuring the accuracy of matching hashes of modified images to their unmodified originals. Modified images are obtained by ap-
Applying a number of common image transformations including JPEG compression, Gaussian noise, mean filtering, cropping, rotation and histogram equalisation. Applying these transformations in combination produces a significant degradation of the original image (see Figure 5.23).

![Original](image1.png) ![Modified](image2.png)

Figure 5.23: Lena image after being transformed by JPEG compression, Gaussian noise, mean filtering, cropping, rotation and histogram equalisation

Expanded hashing accuracy results are presented in Tables 5.5 and 5.6. These provide equal error rates for a larger range of different image transformations. From these results it can be clearly seen that the HOS based techniques outperform their random projection counterparts for most image transformations. The Radon-HOS is also the most superior, outperforming all other methods for all transformations except rotation. Furthermore, the HOS based methods display more consistent accuracy regardless of quantization scheme (adaptive or uniform) and quantizer training. In most cases, the HOS based schemes using uniform quantization (essentially no quantizer training) can outperform the random projection counterparts using Adaptive Deterministic Quantization. **This is a significant result and shows that HOS based projections can alleviate the quantizer training problems presented in Chapter 3.**
Figure 5.24: ROC curves for different hashing methods using a 2-bit Quantizer and Quantizer Training Scenario 1.
Figure 5.25: ROC curves for different hashing methods using a 4-bit Quantizer and Quantizer Training Scenario 1.
Figure 5.26: ROC curves for different hashing methods using a 2-bit Quantizer and Quantizer Training Scenario 6.
Figure 5.27: ROC curves for different hashing methods using a 4-bit Quantizer and Quantizer Training Scenario 6.
Table 5.5: Comparison of equal error rates for Quantizer Training Scenario 1 using different hashing techniques and different image transformations.

<table>
<thead>
<tr>
<th>Image Transformation</th>
<th>Adaptive Quantization</th>
<th>Uniform Quantization</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Mo-RP</td>
<td>Mo-HOS</td>
<td>Sw-RP</td>
</tr>
<tr>
<td>1. JPEG 10% Quality</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2-bit</td>
<td>0.22%</td>
<td>0.17%</td>
<td>0.02%</td>
</tr>
<tr>
<td>4-bit</td>
<td>0.23%</td>
<td>0.10%</td>
<td>0.08%</td>
</tr>
<tr>
<td>2. Mean Filtering (15x15)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2-bit</td>
<td>1.50%</td>
<td>0.20%</td>
<td>40.43%</td>
</tr>
<tr>
<td>4-bit</td>
<td>1.22%</td>
<td>0.12%</td>
<td>41.28%</td>
</tr>
<tr>
<td>3. Gaussian Noise 0.05</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2-bit</td>
<td>1.49%</td>
<td>1.16%</td>
<td>35.17%</td>
</tr>
<tr>
<td>4-bit</td>
<td>1.66%</td>
<td>1.22%</td>
<td>37.41%</td>
</tr>
<tr>
<td>4. Salt &amp; Pepper Noise 15%</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2-bit</td>
<td>1.28%</td>
<td>0.98%</td>
<td>38.45%</td>
</tr>
<tr>
<td>4-bit</td>
<td>1.44%</td>
<td>1.21%</td>
<td>40.17%</td>
</tr>
<tr>
<td>5. Cropping 5 pixels</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2-bit</td>
<td>0.71%</td>
<td>0.56%</td>
<td>0.76%</td>
</tr>
<tr>
<td>4-bit</td>
<td>0.66%</td>
<td>0.34%</td>
<td>0.51%</td>
</tr>
<tr>
<td>6. Rotation 4 degrees</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2-bit</td>
<td>3.49%</td>
<td>1.37%</td>
<td>2.22%</td>
</tr>
<tr>
<td>4-bit</td>
<td>3.18%</td>
<td>0.98%</td>
<td>1.98%</td>
</tr>
<tr>
<td>7. Contrast Stretching</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2-bit</td>
<td>1.46%</td>
<td>0.78%</td>
<td>1.50%</td>
</tr>
<tr>
<td>4-bit</td>
<td>1.45%</td>
<td>0.68%</td>
<td>1.73%</td>
</tr>
<tr>
<td>8. Histogram Equalisation</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2-bit</td>
<td>24.91%</td>
<td>12.55%</td>
<td>27.30%</td>
</tr>
<tr>
<td>4-bit</td>
<td>26.10%</td>
<td>16.69%</td>
<td>27.83%</td>
</tr>
<tr>
<td>9. Sharpening</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2-bit</td>
<td>0.36%</td>
<td>0.10%</td>
<td>2.82%</td>
</tr>
<tr>
<td>4-bit</td>
<td>0.32%</td>
<td>0.03%</td>
<td>3.10%</td>
</tr>
<tr>
<td>Combination of 1, 2, 3, 5, 6 and 8</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2-bit</td>
<td>26.63%</td>
<td>15.05%</td>
<td>34.78%</td>
</tr>
<tr>
<td>4-bit</td>
<td>27.30%</td>
<td>15.72%</td>
<td>35.88%</td>
</tr>
</tbody>
</table>
### Experimental Results

Table 5.6: Comparison of equal error rates for Quantizer Training Scenario 6 using different hashing techniques and different image transformations.

<table>
<thead>
<tr>
<th>Image Transformation</th>
<th>2-bit</th>
<th>4-bit</th>
<th>2-bit</th>
<th>4-bit</th>
<th>2-bit</th>
<th>4-bit</th>
<th>2-bit</th>
<th>4-bit</th>
<th>2-bit</th>
<th>4-bit</th>
<th>2-bit</th>
<th>4-bit</th>
<th>2-bit</th>
<th>4-bit</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Mo-RP</td>
<td>Mo-HOS</td>
<td>Sw-RP</td>
<td>Sw-HOS</td>
<td>Ra-HOS</td>
<td>Mo-RP</td>
<td>Mo-HOS</td>
<td>Sw-RP</td>
<td>Sw-HOS</td>
<td>Ra-HOS</td>
<td>Mo-RP</td>
<td>Mo-HOS</td>
<td>Sw-RP</td>
<td>Sw-HOS</td>
</tr>
<tr>
<td>1. JPEG 10% Quality</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2-bit</td>
<td>3.69%</td>
<td>0.20%</td>
<td>0.44%</td>
<td>0.08%</td>
<td>0.002%</td>
<td>49.40%</td>
<td>0.19%</td>
<td>1.85%</td>
<td>0.02%</td>
<td>0.001%</td>
<td>1.85%</td>
<td>0.02%</td>
<td>0.02%</td>
<td>0.001%</td>
</tr>
<tr>
<td>4-bit</td>
<td>2.04%</td>
<td>0.20%</td>
<td>0.10%</td>
<td>0.03%</td>
<td>0.002%</td>
<td>4.31%</td>
<td>0.20%</td>
<td>0.08%</td>
<td>0.08%</td>
<td>0.001%</td>
<td>4.31%</td>
<td>0.20%</td>
<td>0.08%</td>
<td>0.001%</td>
</tr>
<tr>
<td>2. Mean Filtering (15x15)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2-bit</td>
<td>8.40%</td>
<td>0.26%</td>
<td>37.66%</td>
<td>7.15%</td>
<td>0.80%</td>
<td>49.75%</td>
<td>0.26%</td>
<td>41.87%</td>
<td>9.09%</td>
<td>0.63%</td>
<td>49.75%</td>
<td>0.26%</td>
<td>41.87%</td>
<td>9.09%</td>
</tr>
<tr>
<td>4-bit</td>
<td>7.26%</td>
<td>0.10%</td>
<td>40.35%</td>
<td>6.68%</td>
<td>0.93%</td>
<td>10.74%</td>
<td>0.69%</td>
<td>40.11%</td>
<td>9.01%</td>
<td>0.71%</td>
<td>10.74%</td>
<td>0.69%</td>
<td>40.11%</td>
<td>9.01%</td>
</tr>
<tr>
<td>3. Gaussian Noise 0.05</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2-bit</td>
<td>11.30%</td>
<td>1.50%</td>
<td>40.63%</td>
<td>17.94%</td>
<td>0.76%</td>
<td>50.00%</td>
<td>1.44%</td>
<td>37.08%</td>
<td>18.42%</td>
<td>0.56%</td>
<td>50.00%</td>
<td>1.44%</td>
<td>37.08%</td>
<td>18.42%</td>
</tr>
<tr>
<td>4-bit</td>
<td>9.61%</td>
<td>1.59%</td>
<td>40.98%</td>
<td>17.56%</td>
<td>0.68%</td>
<td>13.90%</td>
<td>2.16%</td>
<td>32.40%</td>
<td>18.87%</td>
<td>0.68%</td>
<td>13.90%</td>
<td>2.16%</td>
<td>32.40%</td>
<td>18.87%</td>
</tr>
<tr>
<td>4. Salt &amp; Pepper Noise 15%</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2-bit</td>
<td>11.83%</td>
<td>1.32%</td>
<td>43.90%</td>
<td>22.66%</td>
<td>0.83%</td>
<td>50.00%</td>
<td>1.45%</td>
<td>37.09%</td>
<td>23.74%</td>
<td>0.81%</td>
<td>50.00%</td>
<td>1.45%</td>
<td>37.09%</td>
<td>23.74%</td>
</tr>
<tr>
<td>4-bit</td>
<td>9.92%</td>
<td>1.39%</td>
<td>44.94%</td>
<td>22.84%</td>
<td>0.87%</td>
<td>14.64%</td>
<td>2.09%</td>
<td>35.23%</td>
<td>24.53%</td>
<td>0.67%</td>
<td>14.64%</td>
<td>2.09%</td>
<td>35.23%</td>
<td>24.53%</td>
</tr>
<tr>
<td>5. Cropping 5 pixels</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2-bit</td>
<td>6.99%</td>
<td>0.63%</td>
<td>2.90%</td>
<td>1.48%</td>
<td>3.53%</td>
<td>49.60%</td>
<td>0.68%</td>
<td>4.85%</td>
<td>1.72%</td>
<td>3.33%</td>
<td>49.60%</td>
<td>0.68%</td>
<td>4.85%</td>
<td>1.72%</td>
</tr>
<tr>
<td>4-bit</td>
<td>5.36%</td>
<td>0.58%</td>
<td>1.43%</td>
<td>1.28%</td>
<td>4.01%</td>
<td>8.03%</td>
<td>0.84%</td>
<td>0.96%</td>
<td>1.63%</td>
<td>3.63%</td>
<td>8.03%</td>
<td>0.84%</td>
<td>0.96%</td>
<td>1.63%</td>
</tr>
<tr>
<td>6. Rotation 4 degrees</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2-bit</td>
<td>15.26%</td>
<td>2.06%</td>
<td>5.49%</td>
<td>3.59%</td>
<td>8.71%</td>
<td>49.70%</td>
<td>1.71%</td>
<td>9.52%</td>
<td>3.91%</td>
<td>7.74%</td>
<td>49.70%</td>
<td>1.71%</td>
<td>9.52%</td>
<td>3.91%</td>
</tr>
<tr>
<td>4-bit</td>
<td>13.66%</td>
<td>1.53%</td>
<td>4.19%</td>
<td>3.37%</td>
<td>8.18%</td>
<td>15.82%</td>
<td>2.34%</td>
<td>2.67%</td>
<td>3.98%</td>
<td>7.84%</td>
<td>15.82%</td>
<td>2.34%</td>
<td>2.67%</td>
<td>3.98%</td>
</tr>
<tr>
<td>7. Contrast Stretching</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2-bit</td>
<td>3.50%</td>
<td>0.69%</td>
<td>2.17%</td>
<td>0.09%</td>
<td>0%</td>
<td>49.40%</td>
<td>1.04%</td>
<td>3.02%</td>
<td>0.09%</td>
<td>0%</td>
<td>49.40%</td>
<td>1.04%</td>
<td>3.02%</td>
<td>0.09%</td>
</tr>
<tr>
<td>4-bit</td>
<td>2.81%</td>
<td>0.68%</td>
<td>2.78%</td>
<td>0.10%</td>
<td>0%</td>
<td>3.99%</td>
<td>0.87%</td>
<td>2.20%</td>
<td>0.08%</td>
<td>0%</td>
<td>3.99%</td>
<td>0.87%</td>
<td>2.20%</td>
<td>0.08%</td>
</tr>
<tr>
<td>8. Histogram Equalisation</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2-bit</td>
<td>33.47%</td>
<td>13.81%</td>
<td>26.04%</td>
<td>9.91%</td>
<td>2.74%</td>
<td>50.00%</td>
<td>14.45%</td>
<td>27.97%</td>
<td>8.81%</td>
<td>2.47%</td>
<td>50.00%</td>
<td>14.45%</td>
<td>27.97%</td>
<td>8.81%</td>
</tr>
<tr>
<td>4-bit</td>
<td>31.77%</td>
<td>13.42%</td>
<td>27.87%</td>
<td>10.58%</td>
<td>3.20%</td>
<td>34.00%</td>
<td>15.29%</td>
<td>28.91%</td>
<td>8.69%</td>
<td>2.82%</td>
<td>34.00%</td>
<td>15.29%</td>
<td>28.91%</td>
<td>8.69%</td>
</tr>
<tr>
<td>9. Sharpening</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2-bit</td>
<td>5.16%</td>
<td>0.10%</td>
<td>6.02%</td>
<td>1.14%</td>
<td>0.002%</td>
<td>49.75%</td>
<td>0.17%</td>
<td>12.08%</td>
<td>1.54%</td>
<td>0.0003%</td>
<td>49.75%</td>
<td>0.17%</td>
<td>12.08%</td>
<td>1.54%</td>
</tr>
<tr>
<td>4-bit</td>
<td>2.78%</td>
<td>0.10%</td>
<td>4.80%</td>
<td>1.15%</td>
<td>0.003%</td>
<td>6.11%</td>
<td>0.20%</td>
<td>3.10%</td>
<td>1.60%</td>
<td>0.003%</td>
<td>6.11%</td>
<td>0.20%</td>
<td>3.10%</td>
<td>1.60%</td>
</tr>
<tr>
<td>Combination of 1, 2, 3, 5, 6 and 8</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2-bit</td>
<td>35.64%</td>
<td>16.68%</td>
<td>33.26%</td>
<td>15.49%</td>
<td>14.07%</td>
<td>50%</td>
<td>17.02%</td>
<td>36.46%</td>
<td>15.70%</td>
<td>12.76%</td>
<td>50%</td>
<td>17.02%</td>
<td>36.46%</td>
<td>15.70%</td>
</tr>
<tr>
<td>4-bit</td>
<td>34.26%</td>
<td>17.09%</td>
<td>34.04%</td>
<td>17.07%</td>
<td>14.93%</td>
<td>36.88%</td>
<td>17.79%</td>
<td>36.87%</td>
<td>15.53%</td>
<td>13.94%</td>
<td>36.88%</td>
<td>17.79%</td>
<td>36.87%</td>
<td>15.53%</td>
</tr>
</tbody>
</table>
CHAPTER 5. ROBUST HASHING USING BISPECTRA

Hashing unpredictability measures the sensitivity of hashes for a fixed image but changing secret-key. This is experimentally measured using the first 10 images from the MIRFLICKR database. For each image 100 hashes are generated using 100 different secret-keys and the binary entropy of the set of 100 hashes is calculated. A measure of entropy is obtained for each image and in order to present the results in a single graph, the average entropy across the 10 images is shown in Figure 5.28. The maximum entropy is a function of the hash length, which is the number of quantization bits multiplied by the number of randomized features \((b \times M)\), entropies are normalized as a percentage of the maximum entropy in order to compare them.

Figure 5.28: Unpredictability of hashes for fixed image but varying secret-key measured using binary entropy.

A higher entropy indicates higher unpredictability and therefore better security. The unpredictability of the Radon-HOS is the highest and most stable,
nearing the maximum for all quantization schemes and quantizer training scenarios.

5.8 Computational Complexity

In this section the computational complexity of the proposed bispectral invariant based hashing methods are compared to the baseline random projection based methods. The need to compute the integrated bispectrum adds considerable overheads to the processing requirements. The processing time of the HOS based hashing methods is directly proportional to \( M \) (the number of Radon projections from which bispectral invariants are extracted) and \( P \) the length of each Radon projection.

Table 5.7 shows the average time (in seconds) to hash a single image. These experiments were carried out on a 2.0GHz Intel Core i7 quad core CPU. Each hashing method is implemented in Matlab and modified to take advantage of parallelisation. The parallel for-loop (parfor function) in Matlab allows the parallel processing of different iterations in a for-loop. In the HOS based methods, calculation of the integrated bispectrum of each of the \( M \) Radon projections is executed in a parallel for-loop. In the FM-RP method, parallel for-loop is used to sum the Fourier coefficients in the \( M \) different radii. For the NMF-RP, a parallel for-loop is used to process the 10 sub-images.

The average times are shown for different number of CPU cores used. The average time per image is calculated from the total time taken to hash the first 50 images from the MIRFLICKR database.

Table 5.7: Average time (in seconds) required to hash a single image using a 2.0 GHz CPU with 1, 2, 3 and 4 CPU cores.

<table>
<thead>
<tr>
<th></th>
<th>Number of CPU cores</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1</td>
</tr>
<tr>
<td>FM-RP</td>
<td>0.13</td>
</tr>
<tr>
<td>NMF-RP</td>
<td>0.89</td>
</tr>
<tr>
<td>FM-HOS</td>
<td>5.23</td>
</tr>
<tr>
<td>NMF-HOS</td>
<td>15.17</td>
</tr>
<tr>
<td>Radon-HOS</td>
<td>42.83</td>
</tr>
</tbody>
</table>

It is clear from Table 5.7 that computing the integrated bispectrum requires considerable amount of processing time. All three HOS based methods use \( M = \)
64 but $P$ varies for each method. For FM-HOS, $P = 64$ the length of the FM feature vector. For NMF-HOS, $P = 215$ the length of the NMF feature vector and for Radon-HOS $P$ varies from 501 to 709, the length of each Radon projection as it varies depending on $\theta$. Processing time for the HOS-based methods increases linearly in proportion with $P$.

Although computationally intensive, computing the integrated bispectrum benefits greatly from parallel processing. Figure 5.29 shows the reduction in processing time (as a percentage) caused by the addition of each processing core. We denote the percentage reduction in processing time as:

$$\kappa_c = \frac{t_{c-1} - t_c}{t_{c-1}}$$

(5.32)

Where $c$ is the number of processing cores used and $t_c$ is the processing time (in seconds) for a single image using $c$ cores. If we assume that processing is evenly divided between each core and each iteration of the parallel for-loop takes the same time than the theoretical maximum value of $\kappa_c$ is:

$$\kappa_c^{\text{max}} = \frac{c - (c - 1)}{c} = \frac{1}{c}$$

(5.33)

Both $\kappa_c$ and $\kappa_c^{\text{max}}$ is shown in Figure 5.29 for each of the hashing methods.

Parallelisation is not always beneficial due to the overheads associated with the parallel for-loop. If each iteration of the parallel for-loop can be quickly evaluated then the overheads of parallelisation can outstrip any performance gains. This is evident for the FM-RP method where performance decreases as the number of cores is increased. Because computing the integrated bispectrum takes considerable time the three HOS based methods benefit the most from parallelisation.

Recent advancements in modern computer processors has favoured increasing the number of cores over increasing processing speed. Standard desktop CPUs can contain 4 cores and a single server CPU such as the Intel Xeon can contain up to 10 cores. With such a system the FM-HOS and NMF-HOS methods could hash an image in around 1 second and be comparable in terms of processing time to the random projection based methods which gain little or nothing from parallelisation.
5.9 Chapter Summary

This chapter has presented an analysis on a new method of robust hashing randomization based on Higher Order Spectra. Bispectral invariants extracted from image Radon projections are used as a set of complex projection basis that is a non-linear function of both the image and secret-key. This non-linear co-dependency prevents the use of linear inversion techniques that existing methods are susceptible to. While improving hashing performance since the projection basis are themselves features of the input image. Furthermore, the randomized features produced by the proposed method are bounded angles with a near uniform distribution and therefore display the encoding and quantization benefits discussed in Chapter 4.

The chapter has covered all steps of the proposed method, beginning with a background on Radon projections and Higher Order Spectra. The experimentation in section 5.5.1 and 5.5.2 examined the robustness, fragility and unpredictability properties of both Radon projections and their Bispectral invariants. It is shown that both Radon projections and Bispectral invariants display robustness to many common image transformations, fragility across a large image database and unpredictability to changes in the secret-key. This experimentation
serves as a proof of concept highlighting the advantages of HOS based randomization.

The Generic-HOS and Radon-HOS randomization methods are described and benchmarked using 1000 images from the MIRFLICKR database for adaptive and uniform quantization using two different quantizer training scenarios from Chapter 3. These results show the HOS methods compare favorably against baseline methods reliant on random projection. Hashing accuracy and fragility are significantly improved for all quantization schemes and training scenarios. Good hashing performance can be achieved even without the need for quantizer training.
Chapter 6

Conclusions and Further Work

6.1 Conclusions

The research presented in this dissertation explores the issues with the randomization, quantization and encoding stages of robust hashing. Most randomization methods are linear in order to preserve Euclidean distance relationship of input features in the randomized domain. This linearity is detrimental to non-invertibility and methods have been proposed that can invert randomization allowing the reconstruction of the original input or the secret-key used. The desirability of binary hash outputs for search and matching efficiency requires the use of quantization and encoding to discretize real-valued features. Although seemingly simple, the process of discretization introduces further problems.

The quantization process requires the training of a background model of feature distributions in order to identify appropriate quantization thresholds. This is similar to the initialisation stage in many pattern recognition algorithms and such a step introduces the potential for overtraining and poor generalisation. In this work, a thorough analysis of quantizer training is conducted and reveals image feature distributions are highly sensitive to choice of training data. This sensitivity increases the chances of overtraining which in turn greatly impacts on hashing accuracy and security. Furthermore, the storage of quantization thresholds is itself problematic because it creates an opportunity for adversaries to tamper with thresholds as a means of manipulating hashes. The encoding stage assigns binary values to quantized outputs and distance distortion is introduced because distances between real numbers are not preserved in their binarised counterparts.

These issues can be addressed by improving on existing randomization tech-
niques. This thesis has proposed a number of such improvements and outlines a new technique of randomization that produces complex projection basis that are derived from the input features and secret-key in a non-linear manner. The resulting randomized features are bounded, with near uniform distribution and are cyclic in nature. These advantages removes the need for quantization training and reduce the distance distortion introduced at the encoding stage. As a result hashing accuracy and security are improved significantly as demonstrated using benchmarks against existing methods on real world data.

6.2 Summary of Contributions

The main contributions of this dissertation are now summarised:

- **A theoretic analysis of the effects of training data selection on randomized feature distributions.** Adaptive deterministic quantization is a method which is currently utilised by many robust hashing algorithms. Surprisingly, little attention has been given to the task of quantizer training. A theoretical analysis of the feature distributions produced by random projection based randomization is carried out. It is theoretically shown that feature distributions are highly sensitive to the choice of secret-key and training examples used in quantizer training. This sensitivity leads to potential overtraining and reduced hashing performance.

- **Analysis of the effects of quantizer overtraining on robust image hashing performance.** Many performance evaluations carried out in the literature do not take into consideration quantizer overtraining. In this dissertation, performance evaluation of three different robust image hashing algorithms is conducted using different quantization training scenarios. The results indicate overtraining is a substantial problem and quantization thresholds do not generalize to unseen data especially in methods based on random projection. This significantly impairs both hashing accuracy and hashing fragility. The results shown in literature are in fact unobtainable in practice.

- **Development of a complex random projection.** A random projection using a complex basis produces a complex-valued output which can
be represented in polar form as a magnitude and angle. The angle is obtained using the non-linear $\arctan$ function and serves as a bounded feature between 0 and $2\pi$. It is shown in this dissertation that angle features suffer less distance distortion than real-valued features when encoded using Gray code. In addition, bounded features simplify the quantization stage because feature limits are known beforehand. This allows uniform quantization to be performed without training and this eliminates the need to store quantization thresholds.

- **Development of a HOS based randomization scheme for image hashing.** Random projection produces randomized features by projecting image features using a secret-key. The secret-key is independent of the input image and thus contributes no discriminating information to the output randomized features. This dissertation presents a new approach where the projection basis are dependent on both the image and secret-key. This co-dependence has two major benefits, firstly the randomization process becomes a form of feature fusion and this can improve hashing performance. Secondly, the projection basis generated with a single secret-key will be different from image to image, this makes the projection difficult to invert and produces randomized features that are more uniformly distributed. The proposed method uses complex-valued HOS features as projection basis that produce angle features. Experimentation shows hashing accuracy and fragility are significantly improved and performance is less dependent on quantizer training.

## 6.3 Future Work

The novel method of randomization proposed in this thesis has primarily been focused for the task of image hashing. A proof of concept for its application in face biometrics is presented in [58] but the method remains largely untested. Face biometrics are not considered to be the most accurate of biometric modalities, intra-class variations between 2D face images can be large and these variations are beyond the tolerance of the proposed HOS method. More invariant biometrics such as iris may be better suited for randomization using the proposed technique.

Biometric template security and biometric cryptography are two areas largely ignored in this dissertation. Both fields are closely related to robust hashing and
CHAPTER 6. CONCLUSIONS AND FURTHER WORK

The proposed methods in this dissertation could be applied. There are a number of challenges that need to be addressed. Biometric features are usually short 1D vectors, and unlike 2D images it is not possible to apply the Radon projection to extract a set of input dependent projection basis. In order to obtain hashes of useful length a new methodology must be developed. One possibility is to iterate the HOS process by feeding back the Bispectral invariants as the input for the next iteration. Each iteration could serve as a feature component so that a length $M$ randomized hash feature can be produced by iterating $M$ times. Such a method would also be useful in Biometric Cryptography where the binary hash vectors $Y$ extracted for a biometric feature can be adapted and used as a cryptographic key. Existing biometric cryptographic methods employ error correction to remove any remaining errors in $Y$ in order to produce an error free deterministic binary sequence. Such a method is desirable because it eliminates the need to store cryptographic keys, they are instead generated as necessary from an input biometric feature. Some preliminary work in these areas have already being presented in [59] and [58].

Another possible future direction is audio hashing. Audio signals are inherently 1D vectors and bispectral invariants can be extracted directly without dimensionality reduction. Common transformations that effect audio signals, such as amplitude changes (volume), time-scaling and additive noise would manifest in a similar manner as image transformations do in Radon projections. This suggests that the proposed method could be adapted easily for audio hashing applications. This could lead to a multi-modal scheme useful in video hashing applications where both audio and image information are present and linked.


Bibliography


