Table of Contents

CHAPTER 1 INTRODUCTION 1

1.1 THE FINGERPRINT AS A BIOMETRIC 2
1.2 PITFALLS OF TRADITIONAL FINGERPRINT STORAGE MECHANISMS 6
1.3 PROPOSED SOLUTIONS FOR SECURING FINGERPRINT TEMPLATES DURING STORAGE 7
1.4 THESIS OBJECTIVES 8
1.5 CONTRIBUTIONS 8
1.6 PUBLICATIONS 10
1.7 THESIS STRUCTURE 11

CHAPTER 2 BACKGROUND AND MOTIVATION 17

2.1 FINGERPRINT RECOGNITION SYSTEM 17
2.2 VULNERABILITIES OF A FINGERPRINT RECOGNITION SYSTEM 19
2.2.1 Type 1: Attack at the Sensor 20
2.2.2 Type 2: Attack on the Communication Channel between the Sensor and Feature Extractor 21
2.2.3 Type 3: Attack on the Feature Extractor 21
2.2.4 Type 4: Attack on the Communication Channel between the Feature Extractor and Template Database 22
2.2.5 Type 5: Attack on the Communication Channel between the Feature Extractor and Matcher 22
2.2.6 Type 6: Attack on the Template Database 23
2.2.7 Type 7: Attack on the Communication Channel between the Template Database and Matcher 23
2.2.8 Type 8: Attack on the Matcher 23
2.2.9 Type 9: Attack on the Communication Channel between the Matcher and Decision Module 24
2.2.10 Type 10: Attack on the Decision Module 24
2.2.11 Type 11: Attack on the Decision Output from the Decision Module 24
2.3 WHY AN ATTACK ON THE TEMPLATE DATABASE IS THE MOST SERIOUS TYPE OF ATTACK 24
2.4 SECURING FINGERPRINT TEMPLATES DURING STORAGE IN A DATABASE 26
2.4.1 Cryptographic Hashing 27
2.4.2 Encryption 28
2.5 SUMMARY 29

CHAPTER 3 LITERATURE REVIEW ON FINGERPRINT TEMPLATE PROTECTION SCHEMES 31

3.1 INTRODUCTION 31
3.2 FEATURE TRANSFORMATIONS 33
3.2.1 Salting 34
3.2.2 Non-invertible Transforms 37
3.3 BIOMETRIC CRYPTOSYSTEMS 44
3.3.1 Key Binding 45
3.3.2 Key Generation 50
3.4 HYBRID PROTECTION SCHEMES 55
3.5 SUMMARY 55

CHAPTER 4 NON-INVERTIBLE FINGERPRINT TRANSFORMS 57

4.1 INTRODUCTION 57
4.2 FINGERPRINT INFORMATION USED 58
4.3 DEALING WITH INTRA-CLASS VARIANCE 59
4.4 TYPES OF EXISTING NON-INVERTIBLE FINGERPRINT TRANSFORMS 60
4.4.1 Perturbation-based Non-invertible Transforms 61

4.4.2 Histogram-based Non-invertible Transforms 64
4.4.3 Non-invertible Transforms based on Local Minutiae Structures 69
4.4.4 Conclusions on the Types of Existing Non-invertible Fingerprint Template Protection Schemes 70
4.5 TECHNIQUES USED TO EVALUATE NON-INVERTIBLE FINGERPRINT TEMPLATE PROTECTION SCHEMES 71
4.5.1 How is Non-invertibility Measured? 71
4.5.2 How is Cancellability Measured? 75
4.5.3 How is Diversity Measured? 76
4.5.4 How is Performance Measured? 77
4.6 SUMMARY 79

CHAPTER 5 A NON-INVERTIBLE CANCELLABLE FINGERPRINT CONSTRUCT BASED ON COMPACT MINUTIAE PATTERNS 81

5.1 INTRODUCTION 81
5.2 THE PROPOSED FINGERPRINT CONSTRUCT 82
5.2.1 Rules for Pattern Formation 83
5.2.2 Local Features 84
5.2.3 Global Features 88
5.2.4 Constructing the Pattern's Feature Vector 91
5.2.5 Comparing Pattern Feature Vectors during Matching 94
5.3 SUITABILITY OF PROPOSED FINGERPRINT CONSTRUCT AS A FINGERPRINT TEMPLATE PROTECTION SCHEME 96
5.3.1 Experiment 1: Pattern Uniqueness 98
5.3.2 Experiment 2: Recognition Accuracy Attainable by Full Minutiae Templates 106
5.3.3 Experiment 3: Verification Speed 108
5.4 SUMMARY 109

CHAPTER 6 CONSISTENCY OF COOPERATIVE USERS IN SCANNING THEIR FINGERPRINTS 111

6.1 INTRODUCTION 111
6.2 FINGERPRINT DATABASE CONSTRUCTION 113
6.2.1 Scanner Specifications 114
6.2.2 Participant Selection 115
6.2.3 Methodology 115
6.3 ANALYSIS OF USER CONSISTENCY IN PLACING FINGER ON SCANNER 117
6.3.1 Translation 117
6.3.2 Rotation 121
6.3.3 Captured Fingerprint Minutiae 123
6.4 IMOPRTANCE OF INVESTIGATION FOR THE PROPOSED FINGERPRINT CONSTRUCT 130
6.5 SUMMARY 130

CHAPTER 7 TRUE FRR OF NEW FINGERPRINT CONSTRUCT 133

7.1 INTRODUCTION 133
7.2 TRUE FRR AS THE NUMBER OF REFERENCE FINGERPRINTS INCREASES 134
7.3 EVALULATION OF THE TRUE FRR WHEN THE USER IS ALLOWED MULTIPLE AUTHENTICATION ATTEMPTS 137
7.4 SUMMARY 140

CHAPTER 8 PERFORMANCE OF PROPOSED FINGERPRINT CONSTRUCT ON COOPERATIVE-USER FINGERPRINT DATABASE 143

8.1 INTRODUCTION 143
8.2 GENERAL EXPERIMENTAL SET-UP 145
8.2.1 Extraction of the Minutiae and Core Points from the Fingerprints 145

8.2.2 Construction of the Reference N-node Patterns 146
8.3 EXPERIMENT 1: EFFECT OF MULTIPLE REFERENCE FINGERPRINTS ON FAR 153
8.3.1 Selection of the Matching Thresholds for Experiment 1 154
8.3.2 Matching Procedure for Experiment 1 156
8.3.3 FAR and FRR for Experiment 1 157
8.4 EXPERIMENT 2: FAR AND FRR IN MFGUAS 158
8.4.1 Selection of the Matching Thresholds for Experiment 2 159
8.4.2 Experiment 2a: Single-Factor Authentication 161
8.4.3 Experiment 2b: Two-Factor Authentication 165
8.5 FAR AND FRR WHEN PROPOSED FINGERPRINT CONSTRUCT IS MODIFIED TO IMPROVE PATTERN UNIQUENESS 168
8.6 SUMMARY 173

CHAPTER 9 PERFORMANCE COMPARISON 175

9.1 INTRODUCTION 175
9.2 EXPERIMENTAL SET-UP 176
9.2.1 Selected Fingerprint Database 176
9.2.2 Minutiae and Core Extraction Procedure 179
9.2.3 Methodology to Calculate EER of $FC_{360°}$ 181
9.3 EXPERIMENTAL RESULTS AND COMPARISON TO OTHER TECHNIQUES 186
9.4 SUMMARY 190

CHAPTER 10 NON-INVERTIBILITY OF $FC_{360°}$ 191

10.1 INTRODUCTION 191
10.2 INFORMATION LEAKED BY AN $FC_{360°}$ *N*-NODE PATTERN 192
10.3 NON-INVERTIBILITY OF AN $FC_{360°}$ *N*-NODE PATTERN 199
10.3.1 Non-invertibility Analysis 1: Proportion of Unrevealed Minutiae Template 199
10.3.2 Non-invertibility Analysis 2: Complexity of Reconstructing Full Minutiae Template 202
10.4 SUMMARY 210

CHAPTER 11 SUSCEPTIBILITY OF $FC_{360°}$ TO A RECORD MULTIPLICITY ATTACK 213

11.1 INTRODUCTION 213
11.2 SUSCEPTIBILITY OF $FC_{360°}$ TO A RECORD MULTIPLICITY ATTACK 214
11.3 MODIFICATION TO $FC_{360°}$ TO STRENGTHEN ITS RESISTANCE TO A RECORD MULTIPLICITY ATTACK 220
11.3.1 The Proposed Modification to $FC_{360°}$ 220
11.3.2 Recognition Accuracy of Floating $FC_{360°}$ versus Fixed $FC_{360°}$ 224
11.4 SUMMARY 227

CHAPTER 12 CANCELLABILITY AND DIVERSITY OF $FC_{360°}$ 229

12.1 INTRODUCTION 229
12.2 CANCELLABILLITY OF *FIXED* $FC_{360°}$ AND *FLOATING* $FC_{360°}$ *N*-NODE PATTERNS 231
12.2.1 Cancellability Analysis 1: Probability of Two N-node Patterns from the Same Fingerprint Matching 231
12.2.2 Cancellability Analysis 2: Number of Different N-node Patterns Available in a Fingerprint 235
12.3 DIVERSITY OF *FIXED* $FC_{360°}$ AND *FLOATING* $FC_{360°}$ *N*-NODE PATTERNS 246
12.4 SUMMARY 249

CHAPTER 13 CONCLUSIONS AND FUTURE WORK 251

13.1 CONCLUSIONS 251
13.2 FUTURE WORK 258

APPENDIX A FAST FINGERPRINT ALIGNMENT METHOD BASED ON MINUTIAE ORIENTATION HISTOGRAMS 263

A.1 INTRODUCTION 263
A.2 LITERATURE REVIEW 265
A.3 ROTATIONAL ALIGNMENT USING MINUTIAE ORIENTATION HISTOGRAMS 267
A.3.1 Creating Minutiae Orientation Histograms 267
A.3.2 Correlating the Histograms 268
A.3.3 Aligning the Query and Reference Minutiae 269
A.4 EXPERIMENTAL RESULTS 270
A.4.1 Experiment 1: Accuracy – Ideal Scenario 270
A.4.2 Experiment 2: Accuracy – Realistic Scenario 272
A.4.3 Experiment 3: Speed 274
A.5 CONCLUSIONS AND FUTURE WORK 276

APPENDIX B A DISSECTION OF FINGERPRINT FUZZY VAULT SCHEMES 279

B.1 INTRODUCTION 279
B.2 THE FUZZY VAULT SCHEME 280
B.2.1 Locking the Vault 281
B.2.2 Unlocking the Vault 281
B.3 FINGERPRINT FUZZY VAULTS 282
B.3.1 Fingerprint Features Used 283
B.3.2 Locking the Vault 283
B.3.3 Unlocking the Vault 287
B.4 CONCLUSION 291

REFERENCES 293

List of Tables

Table 5.1: An estimation of the number of bytes used to store an *N*-node Pattern. 93

Table 5.2: Total number of 3-node, 4-node, and 5-node Patterns obtained using permutations, and the estimated time to sequentially calculate the FRR and FAR for each set of Patterns. 101

Table 5.3: Total number of 3-node, 4-node, and 5-node Patterns obtained using combinations, and the estimated time to sequentially calculate the FRR and FAR for each set of Patterns. 101

Table 5.4: Average maximum differences between corresponding attributes of corresponding genuine Patterns. 102

Table 5.5: Matching thresholds for 3-node, 4-node, and 5-node Patterns, based on the maximum values in Table 5.4. 102

Table 5.6: The FRRs and FARs obtained for 3-node, 4-node, and 5-node Patterns, using the matching thresholds in Table 5.5. 103

Table 5.7: The average time taken to find at least one match for a single 3-node, 4-node, and 5-node Pattern, obtained using MATLAB's "tic toc" function. 108

Table 6.1: Median, interquartile range and range of horizontal and vertical translation distributions in pixels and millimetres. 120

Table 6.2: Comparison of box and whisker plot quantities for minutiae persistence when the minutiae are extracted and matched manually versus automatically. 126

Table 7.1: True FRR for 3-node Patterns when different numbers of reference fingerprints (*n*) are used and when the user is allowed different numbers of authentication attempts (*m*). 139

Table 7.2: True FRR for 4-node Patterns when different numbers of reference fingerprints (*n*) are used and when the user is allowed different numbers of authentication attempts (*m*). 139

Table 7.3: True FRR for 5-node Patterns when different numbers of reference fingerprints (*n*) are used and when the user is allowed different numbers of authentication attempts (*m*). 139

Table 8.1: The 99^{th} percentiles of Pattern attribute differences, and the corresponding matching thresholds. 155

Table 8.2: FAR and FRR across 3 runs of Experiment 1 as the number of reference fingerprints (*n*) increases from 1 to 7. 157

Table 8.3: Average FAR and FRR across the 3 trials from Table 8.2. 158

Table 8.4: The maximum, 99^{th} percentile, and 99.5^{th} percentile of Pattern attribute differences. ... 160

Table 8.5: Attribute-specific thresholds based on the 99^{th} and 99.5^{th} percentiles from Table 8.4. ... 160

Table 8.6: FAR and FRR for Single-Factor Authentication, when $\tau_l = 16$, $\tau_{\alpha\beta} = 14°$, $\tau_{loc} = 22$, $\tau_\omega = 16°$. 162

Table 8.7: FAR and FRR for Single-Factor Authentication, when $\tau_l = 20$, $\tau_{\alpha\beta} = 18°$, $\tau_{loc} = 26$, $\tau_\omega = 23°$. 163

Table 8.8: Comparison of the FARs for Single-Factor Authentication and Two-Factor Authentication, when $\tau_l = 16$, $\tau_{\alpha\beta} = 14°$, $\tau_{loc} = 22$, $\tau_\omega = 16°$. 167

Table 8.9: Comparison of the FARs for Single-Factor Authentication and Two-Factor Authentication, when $\tau_l = 20$, $\tau_{\alpha\beta} = 18°$, $\tau_{loc} = 26$, $\tau_\omega = 23°$. 167

Table 8.10: Comparison of FAR and FRR for $FC_{360°}$ versus $FC_{180°}$ under Single-Factor Authentication, when $\tau_l = 16$, $\tau_{\alpha\beta} = 14°$, $\tau_{loc} = 22$, $\tau_\omega = 16°$. 170

Table 8.11: Comparison of FAR and FRR for $FC_{360°}$ versus $FC_{180°}$ under Single-Factor Authentication, when $\tau_l = 20$, $\tau_{\alpha\beta} = 18°$, $\tau_{loc} = 26$, $\tau_\omega = 23°$. 171

Table 8.12: Comparison of FAR and FRR for $FC_{360°}$ versus $FC_{180°}$ under Two-Factor Authentication, when $\tau_l = 16$, $\tau_{\alpha\beta} = 14°$, $\tau_{loc} = 22$, $\tau_\omega = 16°$. 171

Table 8.13: Comparison of FAR and FRR for $FC_{360°}$ versus $FC_{180°}$ under Two-Factor Authentication, when $\tau_l = 20$, $\tau_{\alpha\beta} = 18°$, $\tau_{loc} = 26$, $\tau_\omega = 23°$. 171

Table 8.14: Comparison of the estimated amount of storage space required for an $FC_{180°}$ *N*-node Pattern versus an $FC_{360°}$ *N*-node Pattern. 173

Table 9.1: The maximum and 99th percentile of Pattern attribute differences. 182

Table 9.2: Attribute-specific thresholds based on the 99th percentiles from Table 9.1. 183

Table 9.3: EER obtained for $FC_{360°}$ for each $N = \{3, 4, 5\}$ in the Normal and Stolen-token Scenarios. 186

Table 9.4: A comparison of the EERs obtained for $FC_{360°}$ at different Pattern sizes to the reported EERs of other non-invertible fingerprint template protection schemes in the literature, both in the Normal and the Stolen-token Scenarios. 187

Table 10.1: Information entropy of minutiae x', y', and θ' attributes. 206

Table 10.2: Maximum number of guesses required to recover the entire minutiae template as the Pattern size, *N*, increases. 206

Table 10.3: The equivalent "bit-strength" of the maximum number of guesses from Table 10.2. 207

Table 11.1: Median number of $FC_{360°}$ *N*-node Patterns needed to reconstruct the entire minutiae template in the worst-case and best-case scenarios. 216

Table 11.2: Median number of $FC_{360°}$ *N*-node Patterns needed to reconstruct the entire minutiae template. 217

Table 11.3: Median number of *Fixed* $FC_{360°}$ versus *Floating* $FC_{360°}$ *N*-node Patterns needed to reconstruct the entire minutiae template. 224

Table 11.4: Recognition accuracy of *Fixed* $FC_{360°}$ VS *Floating* $FC_{360°}$ under Single-Factor Authentication, when $\tau_l = 16$, $\tau_{\alpha\beta} = 14°$, $\tau_{loc} = 22$, $\tau_\omega = 16°$. 225

Table 11.5: Recognition accuracy of *Fixed* $FC_{360°}$ VS *Floating* $FC_{360°}$ under Single-Factor Authentication, when $\tau_l = 20$, $\tau_{\alpha\beta} = 18°$, $\tau_{loc} = 26$, $\tau_\omega = 23°$. 225

Table 11.6: Recognition accuracy of *Fixed* $FC_{360°}$ VS *Floating* $FC_{360°}$ under Two-Factor Authentication, when $\tau_l = 16$, $\tau_{\alpha\beta} = 14°$, $\tau_{loc} = 22$, $\tau_\omega = 16°$. 225

Table 11.7: Recognition accuracy of *Fixed* $FC_{360°}$ VS *Floating* $FC_{360°}$ under Two-Factor Authentication, when $\tau_l = 20$, $\tau_{\alpha\beta} = 18°$, $\tau_{loc} = 26$, $\tau_\omega = 23°$. 226

Table 12.1: Probability of a replacement *N*-node Pattern matching a compromised *N*-node Pattern from the same fingerprint, when $\tau_l = 16$, $\tau_{\alpha\beta} = 14°$, $\tau_{loc} = 22$, $\tau_\omega = 16°$. 233

Table 12.2: Probability of a replacement *N*-node Pattern matching a compromised *N*-node Pattern from the same fingerprint, when $\tau_l = 20$, $\tau_{\alpha\beta} = 18°$, $\tau_{loc} = 26$, $\tau_\omega = 23°$. 233

Table 12.3: Estimation of the median true number of *N*-node Patterns available in a single fingerprint. 236

Table 12.4: Median *practical* number of different *N*-node Patterns available in a fingerprint, when $\tau_l = 16$, $\tau_{\alpha\beta} = 14°$, $\tau_{loc} = 22$, $\tau_\omega = 16°$. 237

Table 12.5: Median *practical* number of different *N*-node Patterns available in a fingerprint, when $\tau_l = 20$, $\tau_{\alpha\beta} = 18°$, $\tau_{loc} = 26$, $\tau_\omega = 23°$. 237

Table 12.6: The median of the maximum number of *diverse* *N*-node Patterns possible from a reference fingerprint for our cooperative-user fingerprint database. 248

Table A.1: Accuracy of the proposed alignment method in a simulated ideal scenario, for various bin sizes. 271

Table A.2: Accuracy of the proposed alignment method on a real fingerprint database, for various bin sizes. 273

Table A.3: A comparison of the accuracy of the proposed alignment method in the ideal and realistic scenarios. 273

Table A.4: Average amount of time taken by the proposed and typical alignment methods (in MATLAB) to determine the rotation parameters for 100 (*Q*, *R*) pairs. 275

List of Figures

Figure 1.1: The 5 Henry fingerprint classes [1]: (a) Plain Arch, (b) Tented Arch, (c) Left Loop, (d) Right Loop, (e) Whorl (Plain), (f) Whorl (Twin Loop). 3

Figure 1.2: The most common fingerprint minutia types. 4

Figure 1.3: The (x, y) location coordinates and orientation, θ, of (a) a termination, and (b) a bifurcation. (Image from [1]) 4

Figure 1.4: Core (indicated by circle) and delta (indicated by triangle) located in all appropriate fingerprints from Figure 1.1. 5

Figure 1.5: Illustration of the core angle in a Right Loop fingerprint. 6

Figure 1.6: Graphical representation of the thesis structure. 15

Figure 2.1: Typical fingerprint recognition system during enrolment (marked by dotted blue arrows) and authentication (marked by solid orange arrows). 18

Figure 2.2: Attack points in a typical fingerprint recognition system. The Type 6 Attack, corresponding to the template database, is circled to indicate that this is the focus of the research presented in this thesis. 20

Figure 3.1: The enrolment and authentication stages in a fingerprint recognition system employing feature transformation to secure its fingerprint templates. (Image from [2]) 33

Figure 3.2: The creation of a user's BioHash (enrolment). Note: In this figure, letters written in bold represent vectors. 36

Figure 3.3: (*Left*) Grid morphing applied to a face image; (*Right*) Block permutations applied to a fingerprint image. (Images from [40]) 39

Figure 3.4: Cartesian Transformation. (*Left*) A fingerprint's minutiae points are overlaid onto a rectangular grid; (*Right*) the grid cells are shuffled to produce the transformed fingerprint template. (Images adapted from [2]) 40

Figure 3.5: Polar Transformation. (*Left*) A fingerprint's minutiae points are overlaid onto a radial grid; (*Right*) the grid shells and sectors are shuffled to produce the transformed fingerprint template. (Images adapted from [2]) 41

Figure 3.6: Functional Transform. (*Left*) A fingerprint's minutiae points are embedded into a 'sheet'; (*Right*) the 'sheet' is then 'crumpled' to produce the transformed fingerprint template. (Images adapted from [3]) 42

Figure 3.7: The enrolment process in a fingerprint fuzzy vault scheme. (Image adapted from [1]) ... 46

Figure 3.8: The verification stage in a fingerprint fuzzy vault scheme. (Image adapted from [1]) 47

Figure 3.9: The enrolment and authentication stages in a key generation biometric cryptosystem. Note that this process would be similar in a key binding biometric cryptosystem, except that the helper data would be constructed using both the biometric template and an external key, *K*, i.e., $H = F(T, K)$. (Image from [2]) 51

Figure 3.10: The enrolment (indicated by dashed blue arrows) and authentication (indicated by solid orange arrows) stages in an example fuzzy extractor framework. 53

Figure 5.1: Valid and invalid 4-node Patterns, where the nodes (minutiae) are labelled by red dots: (a) VALID; (b) INVALID: The two dashed orange lines are exiting from the same node; (c) INVALID: The two dashed orange lines are entering the same node; (d) INVALID: Does not form a closed shape. .. 83

Figure 5.2: Local features of the Pattern from Figure 5.1 (a). The dashed red lines represent the corresponding minutiae orientations, θ. To avoid cluttering, α and β are labelled only for the first connection line. 84

Figure 5.3: Global features of the Pattern from Figure 5.2: (a) Cartesian, (*x*, *y*), and polar, (*d*, *γ*), coordinates depicting the location of the Pattern centroid relative to the core; (b) Pattern orientation, *ω*, is the difference between the orientation of the first connection line (coloured in black and dotted in (a) and (b)), $\boldsymbol{\varphi}_{12}$, and the core angle, θ_c. 88

Figure 5.4: Enrolment of a user's (4-node) reference Pattern into the recognition system's database. 95

Figure 5.5: Two-Factor Authentication. 95

Figure 5.6: Single-Factor Authentication. 95

Figure 6.1: Three samples of the same fingerprint from FVC2002 DB1_A. 114

Figure 6.2: Guide on the proper placement of a finger on the Futronic FS88 scanner: the horizontal lines inside the red rectangles should approximately align. 116

Figure 6.3: Box and whisker plots comparing the horizontal and vertical translation distributions.. 119

Figure 6.4: Box and whisker plot of the rotation distribution. 122

Figure 6.5: Box and whisker plot of the distribution corresponding to the percentage of reference minutiae persisting in a query sample of the reference fingerprint, when the minutiae are extracted and matched manually. 125

Figure 6.6: Box and whisker plot of the distribution corresponding to the percentage of reference minutiae persisting in a query sample of the reference fingerprint, when the minutiae are extracted and matched automatically. 126

Figure 6.7: Box and whisker plots comparing the percentage of reference minutiae persisting in a query sample of the same fingerprint as the number of reference fingerprints increases. 128

Figure 6.8: Plot showing the trend in the median number of reference minutiae remaining for fingerprint recognition purposes as the number of reference fingerprints increases. 129

Figure 7.1: Plot showing the trend in the true FRR for *N*-node Patterns as the number of reference fingerprints increases from 1 to 7. 136

Figure 8.1: Illustration of the method used to obtain (a) the first 4 lists of mated minutiae indices (L1 to L4), and (b) the next 4 lists of mated minutiae indices (L5 to L8), where S*i* refers to fingerprint sample *i*. This process continued until we had 20 lists of mated minutiae indices (i.e., until we reached L20). 148

Figure 9.1: The 8 fingerprint samples from person 3 in FVC2002 DB2_A: Illustrating user inconsistency in placing their finger on the scanner. 177

Figure 9.2: The 8 fingerprint samples from person 93 in FVC2002 DB2_A: Illustrating varying fingerprint quality. 178

Figure 9.3: Minutiae detected (marked by red dots) and missed (circled in green) for image 14_4 in FVC2002 DB2_A, when VeriFinger 6.7 is used for feature extraction. 180

Figure 10.1: Attributes of an example 4-node Pattern. 194

Figure 10.2: Illustration of the fact that the centroid of a shape is located at the point of intersection of the N lines extending from each of its N vertices to the midpoint of the opposite line (NB: In this figure, N = 3). 198

Figure 10.3: Box and whisker plots comparing the non-invertibility of an *N*-node Pattern (i.e., the proportion of the underlying minutiae template that is not revealed by an *N*-node Pattern feature vector) as the Pattern size, *N*, increases from 3 to 5. 201

Figure 10.4: Probability distribution corresponding to minutiae x' attributes. 204

Figure 10.5: Probability distribution corresponding to minutiae y' attributes. 204

Figure 10.6: Probability distribution corresponding to minutiae θ' attributes. 205

Figure 11.1: A 4-node Pattern represented via the (a) *Fixed* $FC_{360°}$ and (b) *Floating* $FC_{360°}$ versions of our proposed fingerprint construct. 222

Figure 12.1: Average number of guesses required to guess the replacement 3-node Pattern as the number of recovered minutiae increases. 240

Figure 12.2: Average number of guesses required to guess the replacement 4-node Pattern as the number of recovered minutiae increases. 241

Figure 12.3: Average number of guesses required to guess the replacement 5-node Pattern as the number of recovered minutiae increases. 241

Figure 12.4: Plot showing the minimum number of minutiae that must be recovered in order for the number of guesses required to guess the replacement 3-node Pattern to exceed 18,250. 243

Figure 12.5: Plot showing the minimum number of minutiae that must be recovered in order for the number of guesses required to guess the replacement 4-node Pattern to exceed 18,250. 244

Figure 12.6: Plot showing the minimum number of minutiae that must be recovered in order for the number of guesses required to guess the replacement 5-node Pattern to exceed 18,250. 245

Figure A.1: Correlation results at different shifts of H^Q. 269

Glossary

FAR	False Accept Rate
FRR	False Reject Rate
EER	Equal Error Rate
N-node Pattern	A Pattern consisting of *N* minutiae
l_{ij}	Length of line connecting minutia *i* to minutia *j* in an *N*-node Pattern
α_{ij}	Angle between connection line connecting minutia *i* to minutia *j*, and orientation of minutia *i*, in an *N*-node Pattern
β_{ij}	Angle between connection line connecting minutia *i* to minutia *j*, and orientation of minutia *j*, in an *N*-node Pattern
x	x-coordinate of an *N*-node Pattern's location relative to the core
y	y-coordinate of an *N*-node Pattern's location relative to the core
ω	Orientation of an *N*-node Pattern relative to the core
τ_l	Matching threshold for Pattern l attributes
$\tau_{\alpha\beta}$	Matching threshold for Pattern α and β attributes
τ_{loc}	Matching threshold for Pattern (x, y) coordinates (Euclidean distance)
τ_ω	Matching threshold for Pattern ω attribute
$\mathbf{FC}_{180°}$	Original Fingerprint Construct with α, β, and ω in [0°, 180°) range
$\mathbf{FC}_{360°}$	Modified Fingerprint Construct with α, β, and ω in [0°, 360°) range
Fixed $\mathbf{FC}_{360°}$	$\mathbf{FC}_{360°}$ with local attributes (l, α, β) *and* global attributes (x, y, ω)
Floating $\mathbf{FC}_{360°}$	$\mathbf{FC}_{360°}$ with local attributes (l, α, β) *only*
MFGUAS	Most Favourable Genuine User Authentication Scenario
Single-Factor Authentication	Authentication scenario in which a person is required to present only their fingerprint for authentication, then the system searches for that person's reference *N*-node Pattern in the query fingerprint
Two-Factor Authentication	Authentication scenario in which a person is required to present both their fingerprint *and* their reference *N*-node Pattern for authentication
FVC2002 DBs	Public fingerprint databases, which were used in the 2002 Fingerprint Verification Competition

THE UNIVERSITY OF AUCKLAND
SCHOOL OF GRADUATE STUDIES

Co-Authorship Form

Graduate Centre
ClockTower – East Wing
22 Princes Street, Auckland
Phone: +64 9 373 7599 ext 81321
Fax: +64 9 373 7610
Email: postgraduate@auckland.ac.nz
www.postgrad.auckland.ac.nz

This form is to accompany the submission of any PhD that contains research reported in published or unpublished co-authored work. **Please include one copy of this form for each co-authored work.** Completed forms should be included in all copies of your thesis submitted for examination and library deposit (including digital deposit), following your thesis Acknowledgements.

Please indicate the chapter/section/pages of this thesis that are extracted from a co-authored work and give the title and publication details or details of submission of the co-authored work.

Chapter 5 is a more detailed version (contains extra analysis and more detailed explanations of various aspects of the proposed method) of the following paper:
V. Krivokuća, et al., "A non-invertible cancellable fingerprint construct based on compact minutiae patterns," International Journal of Biometrics, vol. 6, pp. 125-142, 2014.

Nature of contribution by PhD candidate	Conception, implementation, and analysis of the proposed fingerprint construct; writing of the paper
Extent of contribution by PhD candidate (%)	70

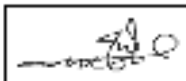
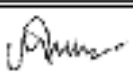
CO-AUTHORS

Name	Nature of Contribution
Waleed Abdulla	Discussion on the idea, advice on experimental procedures, reading and providing feedback on the paper
Akshya Swain	Discussion on the idea and suggestions on points to consider during the analysis, reading and providing feedback on the paper

Certification by Co-Authors

The undersigned hereby certify that:

- the above statement correctly reflects the nature and extent of the PhD candidate's contribution to this work, and the nature of the contribution of each of the co-authors; and
- in cases where the PhD candidate was the lead author of the work that the candidate wrote the text.

Name	Signature	Date
Waleed Abdulla		28/08/2014
Akshya Swain		28/08/2014

THE UNIVERSITY OF AUCKLAND
SCHOOL OF GRADUATE STUDIES

Co-Authorship Form

Graduate Centre
ClockTower – East Wing
22 Princes Street, Auckland
Phone: +64 9 373 7599 ext 81321
Fax: +64 9 373 7610
Email: postgraduate@auckland.ac.nz
www.postgrad.auckland.ac.nz

This form is to accompany the submission of any PhD that contains research reported in published or unpublished co-authored work. **Please include one copy of this form for each co-authored work.** Completed forms should be included in all copies of your thesis submitted for examination and library deposit (including digital deposit), following your thesis Acknowledgements.

Please indicate the chapter/section/pages of this thesis that are extracted from a co-authored work and give the title and publication details or details of submission of the co-authored work.

The material in Chapter 6 was based on the following paper:

V. Krivokuća, et al., "Minutiae Persistence among Multiple Samples of the Same Person's Fingerprint in a Cooperative User Scenario," in Proceedings of the 3rd International Conference on Pattern Recognition Applications and Methods, ESEO, Angers, Loire Valley, France, 2014, pp. 76-86.

There are some differences, however, in the way in which the analysis was done.

Nature of contribution by PhD candidate	Design of the fingerprint database collection procedure and subsequent construction of the database, implementation of experiments on the resulting database, analysis of the experimental findings, and writing of the paper
Extent of contribution by PhD candidate (%)	70

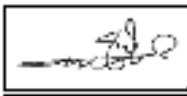
CO-AUTHORS

Name	Nature of Contribution
Waleed Abdulla	Providing guidance and feedback for the ethics application form, discussion on the database collection procedure and the analysis, reading and providing feedback on the paper
Akshya Swain	Involvement in discussions on the database construction procedure and the analysis

Certification by Co-Authors

The undersigned hereby certify that:

- the above statement correctly reflects the nature and extent of the PhD candidate's contribution to this work, and the nature of the contribution of each of the co-authors; and
- in cases where the PhD candidate was the lead author of the work that the candidate wrote the text.

Name	Signature	Date
Waleed Abdulla		28/08/2014
Akshya Swain		28/08/2014

THE UNIVERSITY OF AUCKLAND
SCHOOL OF GRADUATE STUDIES

Co-Authorship Form

Graduate Centre
ClockTower – East Wing
22 Princes Street, Auckland
Phone: +64 9 373 7599 ext 81321
Fax: +64 9 373 7610
Email: postgraduate@auckland.ac.nz
www.postgrad.auckland.ac.nz

This form is to accompany the submission of any PhD that contains research reported in published or unpublished co-authored work. **Please include one copy of this form for each co-authored work.** Completed forms should be included in all copies of your thesis submitted for examination and library deposit (including digital deposit), following your thesis Acknowledgements.

Please indicate the chapter/section/pages of this thesis that are extracted from a co-authored work and give the title and publication details or details of submission of the co-authored work.

Most of the material in Chapter 6 was extracted from the following paper:

V. Krivokuća and W. Abdulla, "Intra-Class Variance among Multiple Samples of the Same Person's Fingerprint in a Cooperative User Scenario," in ICPRAM 2014 - Best Papers, M. De Marsico, et al., Eds., ed: Springer 2014.

Some additional explanations and insights were added into the chapter, however.

Nature of contribution by PhD candidate	Design of the fingerprint database collection procedure and subsequent construction of the database, implementation of experiments on the resulting database, analysis of the experimental findings, and writing of the paper
Extent of contribution by PhD candidate (%)	70

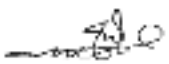
CO-AUTHORS

Name	Nature of Contribution
Waleed Abdulla	Providing guidance and feedback for the ethics application form, discussion on the database collection procedure and the analysis, reading and providing feedback on the paper

Certification by Co-Authors

The undersigned hereby certify that:

- the above statement correctly reflects the nature and extent of the PhD candidate's contribution to this work, and the nature of the contribution of each of the co-authors; and
- in cases where the PhD candidate was the lead author of the work that the candidate wrote the text.

Name	Signature	Date
Waleed Abdulla		28/08/2014

Last updated: 25 March 2013

THE UNIVERSITY OF AUCKLAND
SCHOOL OF GRADUATE STUDIES

Co-Authorship Form

Graduate Centre
ClockTower – East Wing
22 Princes Street, Auckland
Phone: +64 9 373 7599 ext 81321
Fax: +64 9 373 7610
Email: postgraduate@auckland.ac.nz
www.postgrad.auckland.ac.nz

This form is to accompany the submission of any PhD that contains research reported in published or unpublished co-authored work. **Please include one copy of this form for each co-authored work.** Completed forms should be included in all copies of your thesis submitted for examination and library deposit (including digital deposit), following your thesis Acknowledgements.

Please indicate the chapter/section/pages of this thesis that are extracted from a co-authored work and give the title and publication details or details of submission of the co-authored work. Appendix A - most of this appendix was extracted from the following paper: V. Krivokuća and W. Abdulla, "Fast fingerprint alignment method based on minutiae orientation histograms," in Proceedings of the 27th Conference on Image and Vision Computing New Zealand, Dunedin, New Zealand, 2012, pp. 486-491.

Nature of contribution by PhD candidate	Conception of the idea, implementation and testing of the method, writing of the paper
Extent of contribution by PhD candidate (%)	70

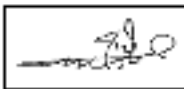
CO-AUTHORS

Name	Nature of Contribution
Waleed Abdulla	Involvement in idea conception and disucssion, reading the paper and providing feedback on its content

Certification by Co-Authors

The undersigned hereby certify that:

- the above statement correctly reflects the nature and extent of the PhD candidate's contribution to this work, and the nature of the contribution of each of the co-authors; and
- in cases where the PhD candidate was the lead author of the work that the candidate wrote the text.

Name	Signature	Date
Waleed Abdulla		28/08/2014

Last updated: 25 March 2013

THE UNIVERSITY OF AUCKLAND
SCHOOL OF GRADUATE STUDIES

Co-Authorship Form

Graduate Centre
ClockTower – East Wing
22 Princes Street, Auckland
Phone: +64 9 373 7599 ext 81321
Fax: +64 9 373 7610
Email: postgraduate@auckland.ac.nz
www.postgrad.auckland.ac.nz

This form is to accompany the submission of any PhD that contains research reported in published or unpublished co-authored work. **Please include one copy of this form for each co-authored work.** Completed forms should be included in all copies of your thesis submitted for examination and library deposit (including digital deposit), following your thesis Acknowledgements.

Please indicate the chapter/section/pages of this thesis that are extracted from a co-authored work and give the title and publication details or details of submission of the co-authored work.

Appendix B - most of this appendix was extracted from the following paper:
V. Krivokuca, et al., "A dissection of fingerprint fuzzy vault schemes," in Proceedings of the 27th Conference on Image and Vision Computing New Zealand, Dunedin, New Zealand, 2012, pp. 256-261.

Nature of contribution by PhD candidate	Literature review on the fuzzy vault scheme, implementation of various relevant concepts, writing of the paper
Extent of contribution by PhD candidate (%)	70

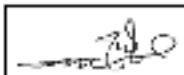
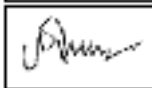
CO-AUTHORS

Name	Nature of Contribution
Waleed Abdulla	Discussion on fuzzy vault concepts, reading and providing feedback on the paper
Akshya Swain	Discussion on fuzzy vault concepts, reading and providing feedback on the paper

Certification by Co-Authors

The undersigned hereby certify that:

- the above statement correctly reflects the nature and extent of the PhD candidate's contribution to this work, and the nature of the contribution of each of the co-authors; and
- in cases where the PhD candidate was the lead author of the work that the candidate wrote the text.

Name	Signature	Date
Waleed Abdulla		28/08/2014
Akshya Swain		28/08/2014

Chapter 1

Introduction

We live in an information-age, where easy access to proliferating amounts of data makes identity theft a lucid reality and an escalating issue to contend with. Traditional security tokens, such as passwords and access cards, are becoming inadequate for the enhanced identity management and user authentication methods we urgently require. This point is particularly important when viewed in the context of the increasing reliance on automated authentication in an ever-growing computerised global network. As our need for more robust security practices increases, we are beginning to witness a surge in biometric recognition technologies, whose global market is expected to reach an estimated $16.7 billion in 2019 – more than a three-fold increase from its $5.2 billion market in 2012 [4].

Biometrics are distinctive physiological and behavioural characteristics of a person, which may be used to recognise them in an automated manner [5, 6]. Physiological biometrics are anatomical human traits, and examples include fingerprints; palm prints; hand and finger geometry; iris patterns; retina patterns; vascular patterns in fingers, palms and wrists; face contour; ear contour; DNA; finger knuckle prints; skin pores; fingernail ridgelines [7]; the lunula (white area at the base of a fingernail) [8]; dentures [9]; buttocks [10]; and even odour. Examples of behavioural biometrics include gait, handwritten signature, keystroke dynamics, speaker recognition, and personality traits.

The motivation behind the escalating use of biometrics for recognition purposes stems from their numerous benefits over traditional security tokens, including persistence over time, the ability to provide non-repudiation due to the fact that biometrics are inherently linked to their owner, uniqueness, universality, and the convenience they offer to the public by providing highly secure and reliable means of authentication without the need to memorise complex passwords or carry around access tokens [5].

Despite the potential security benefits of biometrics, their widespread deployment ultimately depends on the public's acceptance of biometric technologies in their everyday lives. Several security issues prevalent in biometric systems require urgent attention before biometric technologies can be fully integrated into authentication systems in practice, the

most serious of which concerns the safe storage of biometrics in databases. Since the pervasiveness of biometric technologies is reliant upon the collection of the users' biometric data, growing biometrics databases bring with them increasing concerns about the possibility of theft or misuse of people's biometrics and an ensuing lack of privacy in an impending *Big Brother* scenario. The fact that biometrics are an inherent part of the human body means that theft or misuse of biometric data would lead to a lifelong compromise of the victim's identity and privacy. Such an unfortunate occurrence would defeat the purpose of using biometrics to eliminate identity theft that is accomplished via traditional means, such as stealing a credit card, for example.

To prevent the theft or misuse of biometric data, and consequently maintain the privacy of biometric technology users, biometric data must be effectively secured during storage in a database. The importance of securely storing digital biometric data has only been realised relatively recently (i.e., within the past decade or so) as a result of the rapidly escalating focus on biometrics as the answer to growing security needs. The relative infancy of this field of work and the challenging nature of this task means that there does not yet exist an agreed-upon solution to the problem of secure biometric data storage. The difficulty of this problem, combined with the urgency of developing an effective solution, has motivated the research presented in this thesis.

The remainder of this chapter begins by justifying the choice of fingerprints as the focus biometric modality for the work presented in this thesis, and the identifying features of a fingerprint are discussed. We then briefly consider the limitations of traditional fingerprint database storage mechanisms and the unsuitability of the solutions proposed thus far. This is succeeded by the objectives of this thesis, followed by a summary of the thesis' contributions and a list of publications obtained from this research thus far. The chapter concludes by laying out the structure of the remainder of this thesis.

1.1 THE FINGERPRINT AS A BIOMETRIC

The work presented in this thesis is based on the fingerprint modality. The choice of fingerprints as the focus biometric for our research is founded on their popularity. Fingerprints are the most popular biometric in use today, which may be largely attributed to their maturity as a biometric identifier: fingerprints have been used for recognition purposes for over a century (e.g., see [11]), and the validity of this means of authentication has been well established. The maturity of fingerprint recognition, in conjunction with the decreasing size and cost of modern fingerprint scanners, means that fingerprints are highly likely to

continue being widely deployed in biometric recognition systems in the future. Indeed, a recent biometrics market report, the summary of which is available in the Wall Street Journal [12], forecasts that fingerprint recognition will continue to dominate the biometrics market in the foreseeable future.

The current popularity of fingerprint recognition technologies, as well as the foreseen pervasiveness of this means of authentication, urgently calls for an effective mechanism of securely storing the growing fingerprint databases. Note that, while fingerprints have traditionally been employed in forensics, the focus of this thesis is only on *civilian* fingerprint-based recognition systems. This is because civilian fingerprint authentication applications may be expected to become widespread in the near future, which will impact a significantly larger sector of the population than that in the forensics arena.

A fingerprint consists of a pattern of interleaved *ridges* (the raised parts) and *valleys* (the dips). The first step in fingerprint recognition commonly involves categorising the fingerprint into one of five fundamental classes, called the Henry classes, which classify a fingerprint according to its *global* pattern. Figure 1.1 illustrates the five Henry classes, which consist of *Plain Arch, Tented Arch, Left Loop*, *Right Loop*, and *Whorl* [13] (note that two types of Whorl patterns are depicted in Figure 1.1).

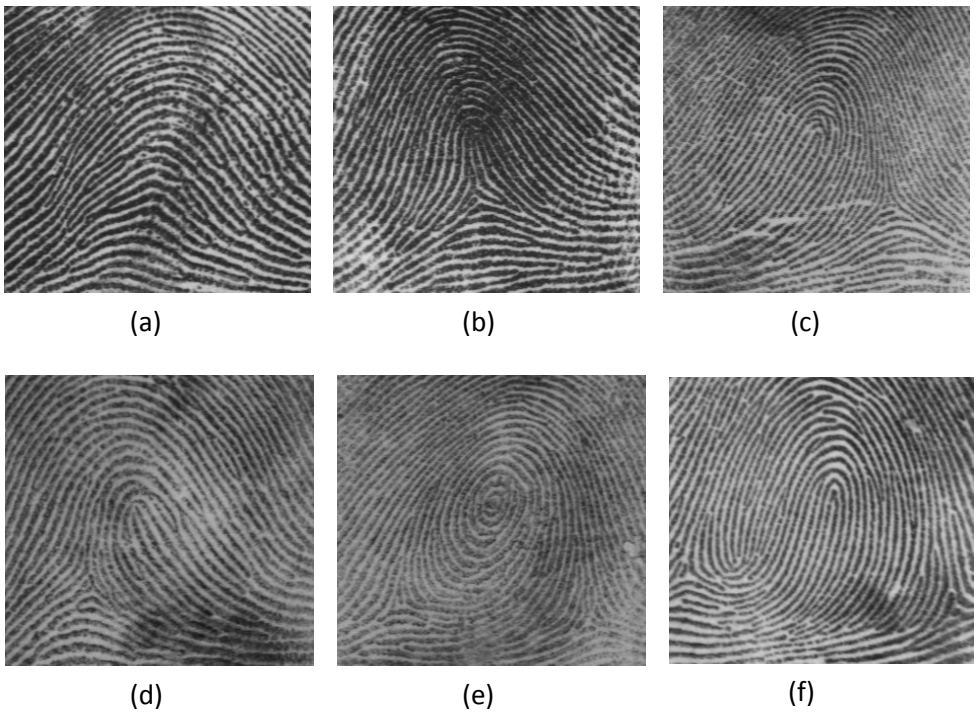


Figure 1.1: The 5 Henry fingerprint classes [1]: (a) Plain Arch, (b) Tented Arch, (c) Left Loop, (d) Right Loop, (e) Whorl (Plain), (f) Whorl (Twin Loop).

The second step in fingerprint recognition is to analyse the fingerprint at the *local* level. Local level analysis involves the examination of small ridge discontinuities called *minutiae*. The two most common minutia types are the *bifurcation*, which occurs at the point where a

ridge line forks out into two separate ridges, and the *termination*, which represents a prematurely ending ridge. Figure 1.2 illustrates the bifurcation and termination minutia types.

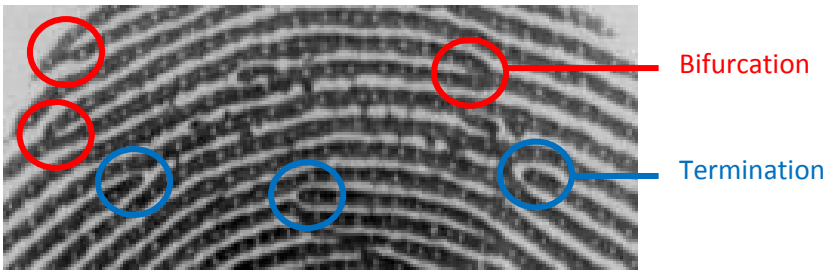


Figure 1.2: The most common fingerprint minutia types.

Note that bifurcations and terminations are generally the only types of minutiae considered in fingerprint recognition due to their frequent occurrence and ease of detection by automated minutiae extraction algorithms [1]; therefore, these are the only minutiae types considered in this thesis.

A minutia is generally represented in terms of four attributes [5]: the x- and y-coordinates pertaining to the minutia's location in the fingerprint; the orientation of the ridge line to which the minutia is attached, θ; and the minutia type (i.e., bifurcation or termination). Since varying quality of the acquired fingerprint image may result in a bifurcation being mistaken for a termination and vice-versa [1], the minutia type is commonly emitted from the minutia representation. This thesis shall, therefore, consider the representation of a minutia in terms of its x, y, and θ attributes only. Figure 1.3 illustrates the (x, y) coordinates and the orientation, θ, of a termination and a b.

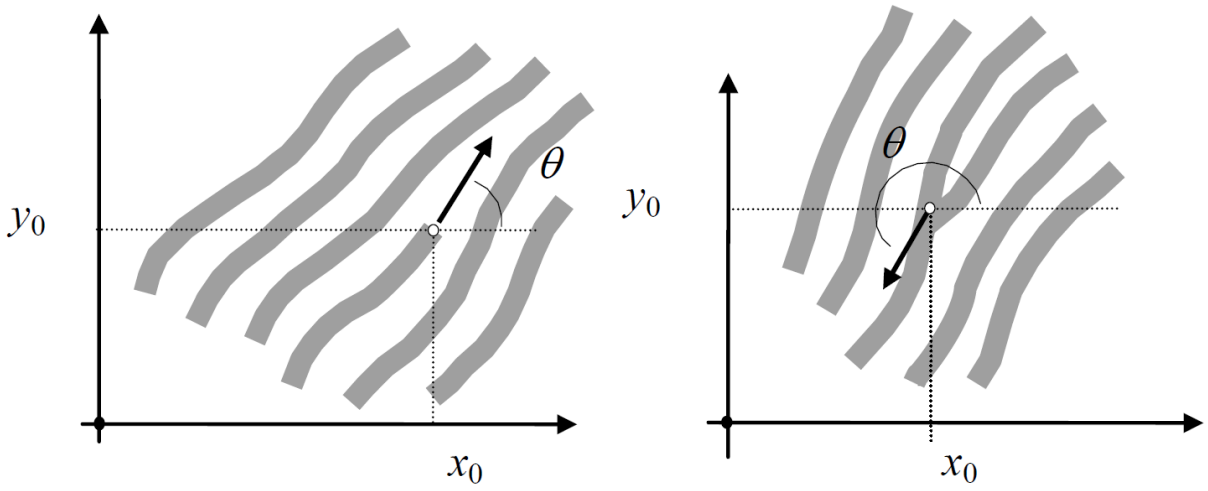


Figure 1.3: The (x, y) location coordinates and orientation, θ, of (a) a termination, and (b) a bifurcation. (Image from [1])

Automated fingerprint matching is most commonly based on minutiae [1]. The more minutiae that two fingerprints have in common, the greater the probability that they originated from the same finger. Traditionally, a minimum of 12 matching minutiae has been considered

sufficient evidence for confirming a fingerprint match [1, 14]. Minutiae are considered to match if the difference between their respective locations and orientations is smaller than a pre-defined threshold. Although a variety of minutiae matching algorithms exist in the literature and in practice, the idea is essentially the same. The two minutiae sets must first be aligned so that the minutiae locations and orientations are expressed relative to a common reference frame. Note that a common reference point is the fingerprint's *core* point, which is defined as the centre of the north-most loop type pattern in a fingerprint image [5]. For fingerprints that do not contain loops, the core usually corresponds to the point of maximum ridge line curvature. The orientation of the line between the core point and a secondary reference point is commonly used to define the orientation of the fingerprint as a whole and thus the minutiae angles are expressed relative to this reference orientation [5]. The secondary reference point is often taken to be the *delta*, which is marked along with the core point in Figure 1.4 for the appropriate fingerprints from Figure 1.1. Note that the delta is not marked in Figure 1.4(a) because this fingerprint is of the Arch type, and Arch types do not contain a delta.

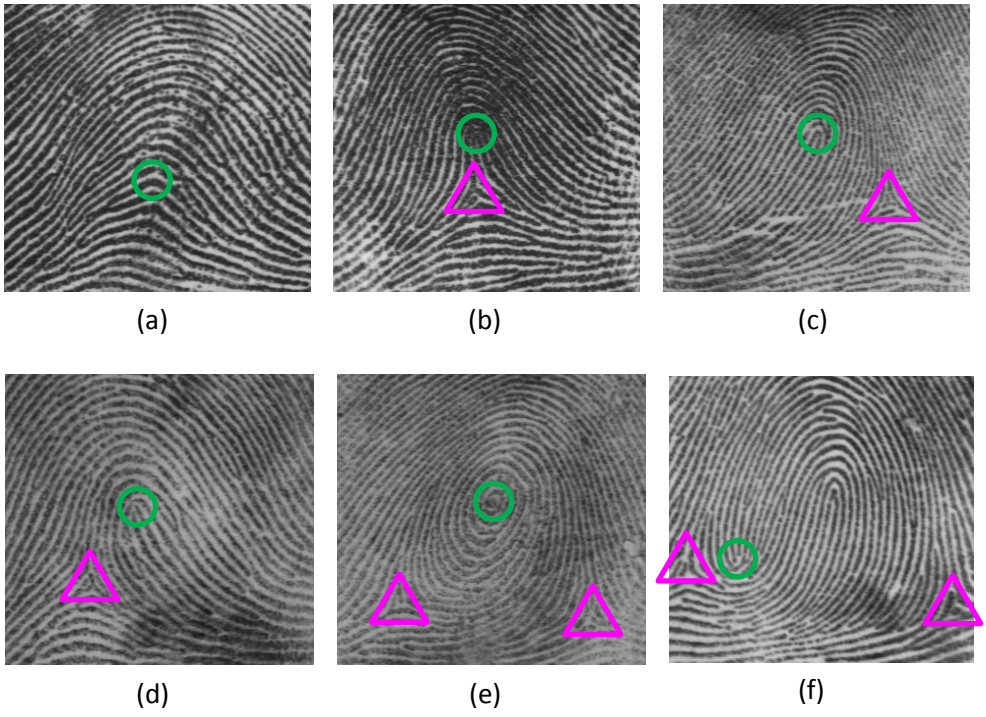


Figure 1.4: Core (indicated by circle) and delta (indicated by triangle) located in all appropriate fingerprints from Figure 1.1.

Since the delta may not always be detected in a fingerprint image, another common way of defining the reference orientation of a fingerprint is via the core angle. Figure 1.5 illustrates an example of a core angle:

Figure 1.5: Illustration of the core angle in a Right Loop fingerprint.

After alignment, minutia matching takes the form of some sort of point-pattern matching problem, where the objective is to pair up as many minutiae from the two fingerprints as possible.

1.2 PITFALLS OF TRADITIONAL FINGERPRINT STORAGE MECHANISMS

Traditionally, the storage of a fingerprint in a database has involved the storage of the raw fingerprint image. While this storage mechanism is still in use today, in more recent years there has been a shift towards representing a fingerprint in terms of a compact feature set consisting of the fingerprint's minutiae set, which is referred to as a fingerprint template. The main benefit of the latter approach is a reduction in the amount of required storage space, particularly for large fingerprint databases. Furthermore, the compactness of a fingerprint template has understandably led to the common belief that templates do not reveal significant information about the original fingerprint image. In fact, traditionally, template generation algorithms have been assumed to be a one-way process [15], similar to a password hash [16]. Consequently, minutiae templates were not believed to contain sufficient information to enable the reconstruction of the underlying fingerprint image [17], and it was thus assumed that the storage of a fingerprint in the form of a minutiae template would provide ample security in terms of preventing the revelation of the underlying fingerprint.

Several researchers have gone on to prove, however, that minutiae templates are actually *reversible*, in that the information they provide is indeed sufficient to reconstruct the original

fingerprint image; for example [7, 15, 17, 18]. As a result, the protection of fingerprint templates during storage in a database has gained considerable focus in the past few years.

1.3 PROPOSED SOLUTIONS FOR SECURING FINGERPRINT TEMPLATES DURING STORAGE

To prevent recovery of the underlying fingerprint from its stored minutiae template, it has become the norm to *encrypt* fingerprint templates during storage in a database [19]. The main problem with this approach is that the encrypted fingerprint templates are only secure insofar as the decryption key is kept secret. Revelation of the decryption key to an adversary is sufficient to enable them to obtain the unsecured minutiae template. Encryption is thus an unsuitable protection mechanism for securing fingerprint data. Further details on the unsuitability of traditional data protection mechanisms for securing fingerprint templates are provided in Chapter 2.

The unsuitability of traditional data protection mechanisms for securing fingerprint templates has effectively led to the creation of a new research field in the design of special protection schemes specifically suited to the nature of fingerprint data. Perhaps the most influential pioneering work in this area has been the solution proposed in [20], where the authors introduced the concept of *cancellable biometrics* to contend with the irreplaceable nature of fingerprints and other biometric modalities. This concept essentially involves storing a distorted version of a fingerprint in the database, such that the original fingerprint remains hidden. Changing the distortion function enables a person to effectively create multiple templates from the same fingerprint, thereby allowing them to *cancel* and *replace* a stored template in the event of compromise.

While the *cancellable biometrics* concept is sound in theory, it is difficult to implement in practice. This is largely due to the fact that it is challenging to design a distortion function that is *non-invertible*, whilst at the same time maintaining acceptable recognition accuracy; in fact, there is generally a trade-off between these two somewhat contradicting goals. The highly non-trivial nature of this task has seen the emergence of several creative solutions for securing fingerprint templates during the past decade. As of yet, however, there does not exist an agreed-upon solution, which means that the design of an ideal fingerprint template protection scheme remains an open problem.

1.4 THESIS OBJECTIVES

The fundamental aim of this thesis is to develop a new mechanism for securing fingerprint templates during storage in a database, for cooperative-user civilian fingerprint recognition applications, with the particular intention of presenting a fresh point of view on the problem. In order to realise this objective, a number of contributions have been made to the field of fingerprint template protection. These contributions are summarised in Section 1.5.

1.5 CONTRIBUTIONS

The following are the contributions made to the field of fingerprint template protection during this PhD:

- A new fingerprint construct, which is a non-invertible fingerprint template protection scheme by its very nature. The crux of our scheme entails the representation of a fingerprint by a single *N*-node Pattern constructed using a small subset of *N* minutiae from the corresponding minutiae template. The sparsity of the resulting Pattern makes it impossible to reconstruct the original fingerprint template and it ensures that the Pattern is cancellable in the event of compromise. Furthermore, despite its sparsity, an *N*-node Pattern is found to have acceptable recognition accuracy in the cooperative-user scenario for which it is intended. The most important aspect of this fingerprint template protection scheme, which sets it apart from other methods in the associated literature, is that it incorporates only a small fraction of the entire fingerprint template in the generation of the protected template. Consequently, the protected template is intuitively more secure. A rigorous analysis of the proposed fingerprint construct shows that our new method complies satisfactorily with the four properties of an ideal fingerprint template protection scheme: non-invertibility, cancellability, diversity, and performance. It thus seems worthwhile to further mine the potential of this promising new fingerprint template protection scheme. Note that the conception of this scheme and its associated rigorous analysis, which are covered in Chapters 5 to 12, are the main contributions of this thesis. The conceptualisation and preliminary analysis on the potential of our new fingerprint template protection scheme have also been published in [21].
- A cooperative-user fingerprint database was constructed and subsequently mined to provide insight into the manner in which cooperative users may be expected to interact with a fingerprint scanner in civilian fingerprint recognition systems in practice. The results of this investigation are useful for gauging the practicality of our proposed

fingerprint construct in its intended application scenario, but their applicability also extends to the development of automated fingerprint recognition algorithms in general. This contribution is discussed in Chapter 6, and it has also been published in [22, 23]. Note that the collection of fingerprints for the cooperative-user database employed in this investigation was approved by the University of Auckland Ethics Committee under the condition that the resulting database cannot be shared with the wider research community. The database *can*, however, be used within the university for similar investigations in the near future.

- A convenient classification of the methods that fall under the *non-invertible transforms* category of fingerprint template protection schemes in the literature (see Chapter 4). This is important for putting our new fingerprint construct into the context of existing solutions within the same category of fingerprint template protection schemes. The classification will also be useful for other researchers when surveying the associated literature to understand the solutions that have been proposed thus far. A related contribution is a discussion of the most common techniques used to evaluate the methods that fall under the *non-invertible transforms* category of fingerprint template protection schemes in the literature. This will be a useful guide for other developers of fingerprint template protection schemes.
- A new fingerprint alignment algorithm, which corrects rotational differences between two fingerprints by correlating histograms based on the orientations of the fingerprints' minutiae (see Appendix A). Since fingerprint alignment is a difficult and ever-present issue in automated fingerprint recognition systems, the development of a new alignment algorithm is an important contribution. Although our proposed fingerprint construct does not rely on this alignment method, fingerprint template protection schemes of a different nature may find this alignment technique suitable for their purposes. Note that this contribution has been published in [24].
- A dissection of the methods used to implement a popular fingerprint template protection scheme in the literature, the Fuzzy Vault (see Appendix B). The purpose of this work is to assist interested researchers in their own implementations of this method. We believe that this is an important contribution, since researchers are often faced with the intimidating task of attempting to implement an accurate rendition of someone else's work. The most difficult part in this endeavour is knowing where to start. It is our hope that our contribution will help lay the groundwork for this undertaking. Note that this contribution has been published in [25].

1.6 PUBLICATIONS

The publications obtained thus far from the material presented in this thesis are the following:

New fingerprint construct:

- V. Krivokuća*, et al.*, "A non-invertible cancellable fingerprint construct based on compact minutiae patterns," *International Journal of Biometrics,* vol. 6, pp. 125-142, 2014.

Note: *Much of the rigorous analysis conducted on our new fingerprint template protection scheme was conducted after this journal publication. It is our aim to publish this work following the submission of this thesis.*

Investigations on our cooperative-user fingerprint database:

- V. Krivokuća*, et al.*, "Minutiae Persistence among Multiple Samples of the Same Person's Fingerprint in a Cooperative User Scenario," in *Proceedings of the 3rd International Conference on Pattern Recognition Applications and Methods*, ESEO, Angers, Loire Valley, France, 2014, pp. 76-86.
- V. Krivokuća and W. Abdulla, "Intra-Class Variance among Multiple Samples of the Same Person's Fingerprint in a Cooperative User Scenario," in *ICPRAM 2014 - Best Papers*, M. De Marsico*, et al.*, Eds., ed: Springer 2014.

New alignment method:

- V. Krivokuća and W. Abdulla, "Fast fingerprint alignment method based on minutiae orientation histograms," in *Proceedings of the 27th Conference on Image and Vision Computing New Zealand*, Dunedin, New Zealand, 2012, pp. 486-491.

Dissection of the methods used to implement a popular fingerprint template protection scheme (the Fuzzy Vault) in the literature:

- V. Krivokuca*, et al.*, "A dissection of fingerprint fuzzy vault schemes," in *Proceedings of the 27th Conference on Image and Vision Computing New Zealand*, Dunedin, New Zealand, 2012, pp. 256-261.

1.7 THESIS STRUCTURE

The current chapter introduced the motivation behind the work presented in this thesis. The focus on fingerprints as the biometric of choice for this thesis was justified, and local ridge discontinuities called *minutiae* were identified as the most commonly used features in automated fingerprint recognition systems. A brief foray into the pitfalls of traditional fingerprint storage mechanisms and the issues with existing solutions showed that the design of an ideal fingerprint template protection scheme remains an open problem. This formed the motivation for this thesis, the main objective of which is to develop a new fingerprint template protection scheme. A list of contributions and associated publications resulting from this objective were discussed. The remainder of the thesis is structured as follows. Figure 1.6 provides a graphical representation of the thesis structure.

CHAPTER 2 covers the background necessary to understand the motivation for this thesis on a deeper level. A number of attack points on a typical fingerprint recognition system are described, and an attack on the fingerprint template database is identified as the most serious type of attack. The need for an effective fingerprint template protection scheme is thus justified, and this is followed by a discussion on the unsuitability of two traditional, well-established cryptographic mechanisms for this purpose. The chapter concludes by motivating the urgent need for a fingerprint template protection scheme that is specifically suited to the nature of fingerprint templates.

CHAPTER 3 presents a detailed review of the four categories of existing fingerprint template protection schemes: *salting*, *non-invertible transforms*, *key binding* and *key generation*. A discussion on the merits and limitations of each category is provided, along with examples of the most popular technique in each category.

CHAPTER 4 focuses more specifically on the *non-invertible transforms* category of fingerprint template protection schemes. The nature of the existing approaches in this category is explored, and the techniques used to evaluate the robustness of these methods are discussed.

CHAPTER 5 proposes a new non-invertible fingerprint template protection scheme, which entails the representation of a fingerprint by a single N-node Pattern constructed using a small subset of N minutiae from the underlying minutiae template. The sparsity of the resulting

Pattern makes it impossible to reconstruct the original fingerprint template and it ensures that the Pattern is cancellable in the event of compromise. A preliminary investigation suggests that the recognition accuracy attainable by our method would make it suitable for deployment in cooperative-user civilian fingerprint recognition applications. The remainder of this thesis is thus dedicated to a rigorous analysis of our new fingerprint template protection scheme. The proposal of the new fingerprint template protection scheme and the subsequent mining of its potential are the main contributions of this thesis.

CHAPTER 6 investigates the intra-class variability between multiple samples of the same fingerprint acquired from cooperative users of a civilian fingerprint recognition application. The results of this investigation provide encouraging evidence to support the assumption that our proposed fingerprint construct should infrequently suffer from the problem of missing minutiae in practice, provided that it is deployed in the cooperative-user scenario for which it is intended. The results of this investigation will also be useful in the development of fingerprint recognition algorithms intended for deployment in cooperative-user scenarios.

CHAPTER 7 investigates the *true* False Reject Rate (FRR) of our proposed fingerprint construct. A *true* False Reject occurs as a result of a person's reference *N*-node Pattern *physically missing* in a query sample of their reference fingerprint. It is demonstrated that using multiple reference fingerprints to determine the most reliable reference minutiae and allowing a user to have multiple authentication attempts can significantly reduce the FRR. The most favourable balance between the number of reference fingerprints and the number of authentication attempts is established.

CHAPTER 8 evaluates the False Reject Rate (FRR) and False Accept Rate (FAR) of our proposed fingerprint construct on our cooperative-user fingerprint database. This database provides a suitable platform for testing the performance of our proposed fingerprint construct in its intended application scenario. The performance is found to be satisfactory for practical purposes, and the recognition accuracy is further improved by proposing a modification to our fingerprint construct.

CHAPTER 9 compares the performance of our improved fingerprint construct from Chapter 8 with the reported recognition accuracy of other non-invertible fingerprint template protection schemes in the literature. The comparison is found to be favourable.

CHAPTER 10 evaluates the non-invertibility of our proposed fingerprint construct. It is demonstrated that the sparsity of the new construct ensures that the majority of the original fingerprint template remains unrecovered in the event that a person's reference Pattern is stolen from the database. Furthermore, it is shown that recovering the entire minutiae template from the information leaked by a single *N*-node Pattern is computationally infeasible.

CHAPTER 11 analyses the vulnerability of our proposed fingerprint construct to a Record Multiplicity Attack. We demonstrate that it is possible to reconstruct the original fingerprint template by piecing together multiple *N*-node Patterns acquired from the same fingerprint, but that the number of Patterns expected to be required for this endeavour is larger than should be possible to acquire in practice. Nevertheless, in light of these findings, an improved version of our fingerprint construct, which is more resistant to a Record Multiplicity Attack, is proposed.

CHAPTER 12 evaluates the cancellability and diversity of our proposed fingerprint construct. The probability that two different *N*-node Patterns generated from the same fingerprint will match in practice is found to be low, and the number of different *N*-node Patterns existing in a fingerprint is estimated to be more than sufficient for cancellability purposes in practice. Furthermore, it is demonstrated that our proposed fingerprint construct would enable a person to enrol into multiple applications using a different *N*-node Pattern from the same fingerprint in each application, without the possibility of being cross-matched across these applications' databases.

CHAPTER 13 concludes this thesis and provides avenues for future work directed at further mining the potential of our new fingerprint template protection scheme.

APPENDIX A proposes a new fingerprint alignment technique, which can be used to offset rotational differences between two fingerprint samples prior to attempting to match them. Since fingerprint misalignment is one of the most problematic aspects of designing a fingerprint template protection scheme, the proposed alignment method may be considered a beneficial contribution to this field, as well as to the development of automated fingerprint matching algorithms in general. Note that this contribution is located in the appendices, rather than in the main body of this thesis, because the focus of this thesis is on our new

fingerprint template protection scheme, which does not require alignment during authentication.

APPENDIX B presents a dissection of fingerprint-based fuzzy vault implementations in the literature. The purpose of this contribution is to assist interested researchers in their own implementations of the fuzzy vault framework in the context of the fingerprint biometric. Note that this contribution forms a part of the appendices, rather than the main body of the thesis, because the focus of this thesis is not on the fuzzy vault scheme.

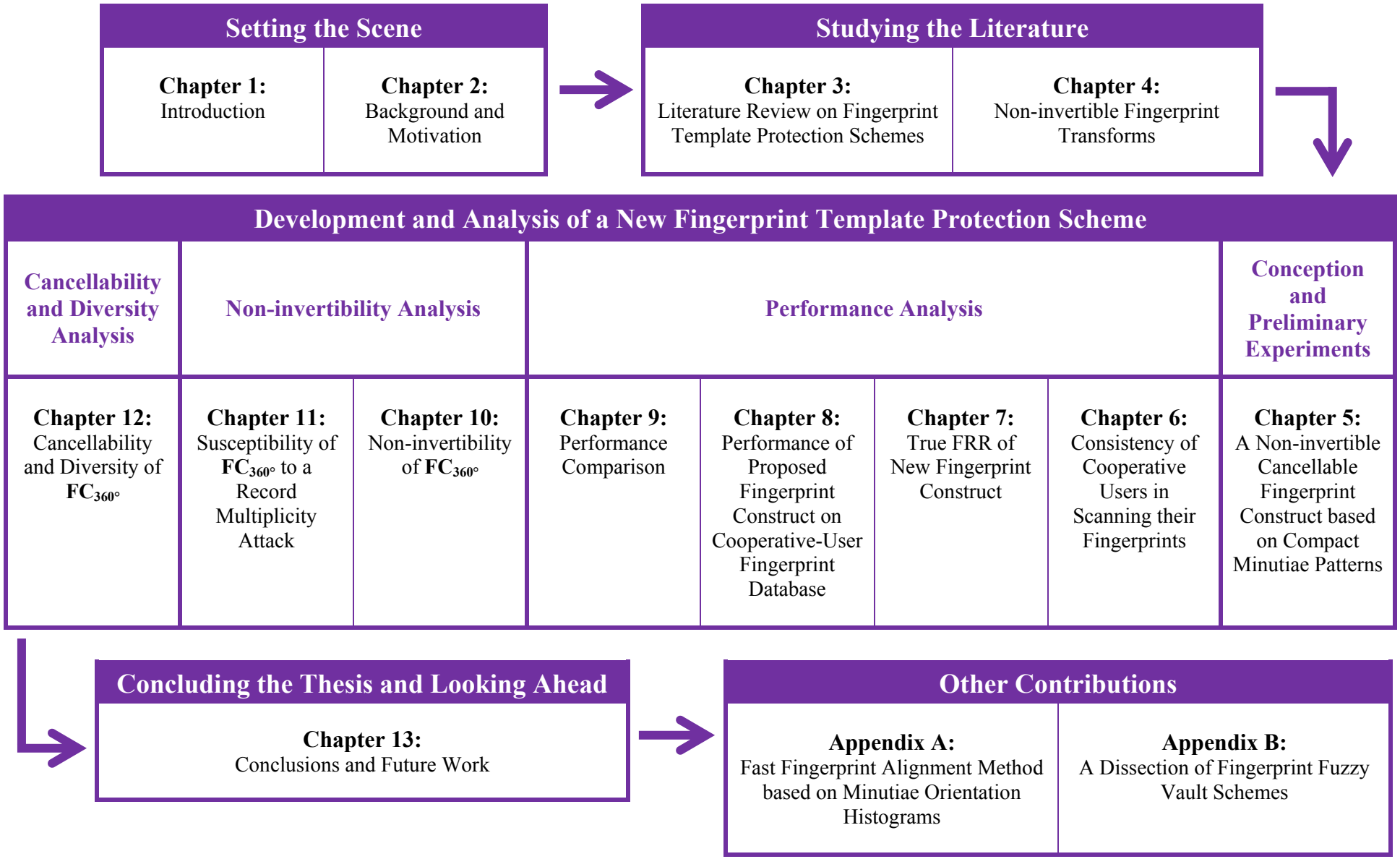


Figure 1.6: Graphical representation of the thesis structure.

Chapter 2

Background and Motivation

This chapter covers the background necessary to understand the motivation for this thesis. The structure and operation of a typical fingerprint recognition system are first described, and a number of typical attack points in a fingerprint recognition system are then identified. This is followed by a discussion on why an attack on the fingerprint template database can be considered the most serious type of attack, thereby providing a natural segue to the focus of this thesis, which concerns the secure storage of fingerprint templates in a database. The notion that the compactness of a fingerprint template is sufficient for protecting the underlying fingerprint is then dispelled, followed by an explanation of why decentralisation of the template database, on its own, is not the solution. The need for an effective fingerprint template protection scheme is thus justified. This is succeeded by a discussion on the unsuitability of two traditional, well-established cryptographic mechanisms for this purpose. The chapter concludes by motivating the urgent need for a fingerprint template protection scheme that is specifically suited to the nature of fingerprint templates.

2.1 FINGERPRINT RECOGNITION SYSTEM

A fingerprint recognition system generally operates in one of two modes: *verification* mode or *identification* mode. The role of a *verification* system is to confirm an individual's identity by conducting a one-to-one comparison between the fingerprint captured at the time of authentication (referred to as the *query* fingerprint) and the person's claimed fingerprint stored in the system's database (referred to as the *reference* fingerprint). A fingerprint system would normally operate in verification mode when it is used to grant access to restricted locations and/or resources. An *identification* system attempts to establish a person's identity by searching the system's entire database for a potential match, thereby conducting a one-to-many search. This sort of system would be applied in situations where one needs to determine whether a person is "known" or not; for example, in searching through a database of people with an established criminal record. When talking about a fingerprint system in

general, one normally refers neither to verification nor identification; rather, the generic term *recognition* is used [6].

A fingerprint recognition system typically consists of five modules, namely: Sensor, Feature Extractor, Template Database, Matcher, and Decision Module [2]. These are illustrated in Figure 2.1.

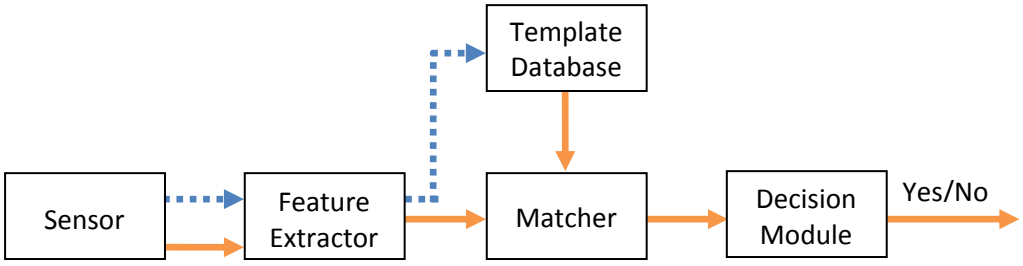


Figure 2.1: Typical fingerprint recognition system during enrolment (marked by dotted blue arrows) and authentication (marked by solid orange arrows).

The sensor scans the user's fingerprint and provides a raw digital image to the feature extractor. The feature extractor then extracts salient features from the fingerprint image to form a compact feature set called a *template*. The template will typically consist of the location coordinates and orientations of all the minutiae in the fingerprint image. During the enrolment of an individual into the biometric system (indicated by the dotted blue arrows in Figure 2.1), their minutiae template is stored in the system database and indexed by the user's corresponding identity information. At the time of authentication (indicated by the solid orange arrows in Figure 2.1), a user provides a fresh sample of their fingerprint to the sensor and, if the system is operating in verification mode, enters some information claiming a particular identity. The matcher then compares the minutiae template extracted from the query fingerprint to the minutiae template stored in the database associated with the person's claimed identity and outputs a numerical match score. If the system is operating in identification mode instead, then a match score will be generated for *every* minutiae template stored in the database. The match score is next fed to the decision module. In a verification system, the decision module will either accept or reject the person's claimed identity based on whether or not the match score is, respectively, above or below the required threshold for successful authentication. In an identification system, the decision module will make a decision on whether or not the query person exists in the template database based on whether *any* of the match scores are above the pre-determined threshold.

Unlike passwords, which remain the same at every presentation, fingerprint measurements are inherently noisy due to changes in physical and environmental conditions. For example, fingerprint images of the same finger will be slightly different at each presentation due to factors such as dirt on the finger or scanner, accumulation of sweat or

moisture on the finger, excessive finger dryness, injuries such as cuts or bruises on the finger, inconsistency in the way in which the finger is placed on the scanner (e.g., translation or rotation of the finger on the scanner surface, variation in the pressure of the finger on the scanner), noise introduced by the sensor (e.g., from residues left over on the glass platen as a result of previous fingerprint captures), and so on [5, 26]. Consequently, the minutiae resulting from multiple acquisitions of the same fingerprint will often exhibit some differences, which means that minutiae templates will never match 100 percent. In fact, a perfect match is almost a guarantee that the system in question has been compromised, since this would generally result from a "replay" of a stolen minutiae template [27].

Due to this discrepancy between different samples of the same fingerprint, the similarity between two minutiae templates is indicated in terms of a *match score*, which essentially indicates the *probability* that the two fingerprints originate from the same finger. The match/no match decision depends upon a threshold, where pairs of fingerprints scoring higher than or equal to the threshold would be considered to *match*, whereas pairs of fingerprints generating scores lower than the threshold would *not* be considered to match. As a consequence of this 'fuzzy' scoring method, fingerprint recognition systems commit two types of errors: a *false accept*, where fingerprints from two different fingers are mistakenly assumed to originate from the same finger; and a *false reject*, where two samples of the same fingerprint are mistakenly assumed to originate from two different fingers. For this reason, the performance of a fingerprint recognition system is normally evaluated in terms of its False Accept Rate (FAR) and False Reject Rate (FRR). There is a strict trade-off between the FAR and FRR: minimizing one error rate will cause an increase in the other, and vice-versa. The point at which the FAR is equal to the FRR is called the Equal Error Rate (EER).

2.2 VULNERABILITIES OF A FINGERPRINT RECOGNITION SYSTEM

A fingerprint recognition system is vulnerable to several types of attacks, each of which has the intention of either circumventing the security afforded by the system or else deterring the normal functioning of the system [28]. Figure 2.2 illustrates a number of attack points in the typical fingerprint recognition system depicted in Figure 2.1, as identified in [29]. Sections 2.2.1 to 2.2.11 describe each type of attack illustrated in Figure 2.2.

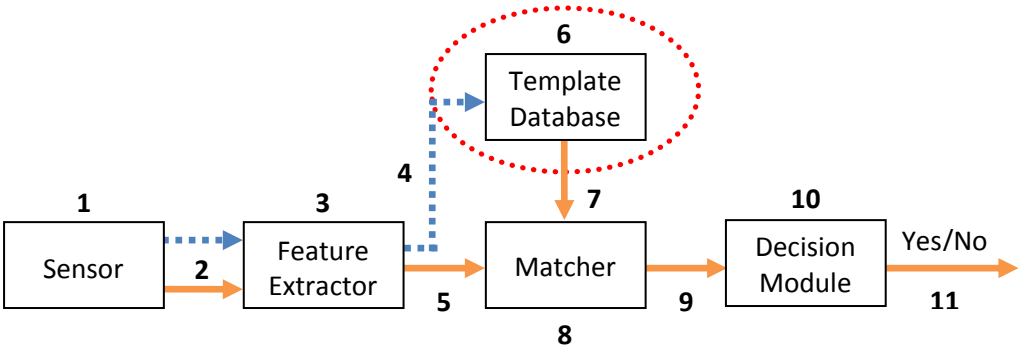


Figure 2.2: Attack points in a typical fingerprint recognition system. The Type 6 Attack, corresponding to the template database, is circled to indicate that this is the focus of the research presented in this thesis.

2.2.1 Type 1: Attack at the Sensor

The predominant type of attack that can be launched against the sensor in a fingerprint recognition system is the presentation of a fake fingerprint (a *spoof*) – for example, a 3D fingerprint created out of gelatine mould or a printed copy of a fingerprint image – in order to gain illegitimate access to the location or resources that the fingerprint system aims to protect. This type of attack is facilitated by the fact that fingerprints are not secrets, since latent fingerprints are left behind on every surface we touch [30]. Consequently, it is often possible to obtain a person's fingerprint without their knowledge or consent, which permits *covert recognition* of people that were previously enrolled in a fingerprint recognition system [5]. Schneier [30] considers this lack of secrecy to be one of the main problems of fingerprint recognition systems (and biometric systems in general).

Several researchers have described effective spoofing techniques that are capable of fooling commercial fingerprint recognition systems, e.g., [31-36]. The most common method of dealing with spoofing attacks at the sensor is via the use of liveness detection, which aims to establish whether the fingerprint data being captured by the sensor comes from a legitimate, live user, who is physically present at the point of authentication [37]. Examples of liveness detection techniques can be found in [5, 20, 36, 38, 39]. Another solution towards thwarting spoofing attacks involves the use of multiple biometric traits to confirm the identity of a single person, which is referred to as a multibiometric system [40]. Examples of multibiometric systems employing fingerprints may be found in [40-45].

Note that, out of all the attacks depicted in Figure 2.2, a Type 1 attack is perhaps the most feasible [46]. This is because, firstly, no knowledge of the underlying operation of the system's modules is necessary, and secondly, this sort of attack occurs in the analogue domain, so digital protection techniques, such as encryption, are not applicable.

2.2.2 Type 2: Attack on the Communication Channel between the Sensor and Feature Extractor

The channel between the sensor and feature extractor could be intercepted by an intruder in order to obtain the digital image of an authorized user's fingerprint originating from the sensor. The adversary could then use this image to create an artificial spoof, which could later be presented at the sensor in order to gain unauthorized access to the restricted location or resources that the fingerprint system aims to protect. Alternatively, the digital image could be resubmitted to the feature extractor at a later time, thereby bypassing the sensor altogether. The latter type of attack is referred to as a replay attack [5].

Encryption, digital signatures, timestamps, and challenge-response mechanisms [5, 20] are the most common methods of dealing with replay attacks. Another way to address a replay attack is to use data-hiding techniques (for example, digital watermarking) to secretly embed a tell-tale mark in the fingerprint image [20, 47-53]. Any tampering with the image would be detectable, and the absence of the mark would likewise suggest an illegitimate image submission. Note that, out of all these prevention techniques, encryption and data hiding are the only ones that actually aim to prevent theft of the fingerprint data; hence, aside from dealing with replay attacks, these two techniques would also be effective in preventing spoof creation resulting from stealing a user's raw fingerprint image from this channel. Having said that, encryption on its own is not very effective at preventing replay attacks, since it cannot check for 'liveness' of the presented data; that is, the fingerprint data could be stolen in encrypted format and replayed later on in the communication channel. For this reason, encryption must usually be combined with one of the other aforementioned techniques, which is capable of checking when the transmitted data was created.

2.2.3 Type 3: Attack on the Feature Extractor

In this type of attack, the feature extractor can be replaced by a Trojan horse program, which would produce its own feature sets and thus bypass the feature extractor [5]. A common way for a Trojan horse program to launch an attack against the feature extractor is by using a strategy referred to as Hill climbing [5]. In a Hill-climbing attack, the Trojan horse program iteratively modifies an initial template, observing the match score at each stage, until the match score exceeds the threshold for successful authentication and the false fingerprint representation is accepted by the recognition system. Hill-climbing attacks on fingerprint recognition systems have been demonstrated in [46, 54].

Suggested ways of mitigating the Hill-climbing attack include avoiding the publication of match scores and just outputting the accept/reject decision [46], using quantised match scores (instead of absolute scores) in order to increase the amount of time needed to launch a hill climbing attack and thereby render the process impractical [55] (however, it was shown that it is still possible to regenerate biometric images using a Hill-climbing attack, even if the match score data is quantised [56]), limiting the number of sequential authentication attempts in order to reduce the number of Hill-climbing attacks against a single user [55], and using a masking operator on output match scores of unsuccessful authentication attempts, such that the matcher outputs a random score that is smaller than the pre-set decision threshold for authentication [46].

An alternative method of overcoming the feature extractor is to generate synthetic fingerprint features, which the Trojan horse program could enrol into the system instead of the actual fingerprint images coming through to the feature extractor. For example, [57, 58] have shown that it is possible to artificially create highly realistic fingerprint images.

2.2.4 Type 4: Attack on the Communication Channel between the Feature Extractor and Template Database

This is similar to a Type 2 attack, except this time the feature set (e.g., minutiae template) of a legitimate user's fingerprint, rather than the fingerprint image, is being snooped for. This feature set can be saved and replayed later on in the channel [5]. Alternatively, the features can be employed in the reconstruction of the original fingerprint image, thereby facilitating spoof creation [7]. Prevention techniques for a Type 2 attack can also be used to mitigate a Type 4 attack.

2.2.5 Type 5: Attack on the Communication Channel between the Feature Extractor and Matcher

This type of attack is essentially the same as a Type 4 attack, in that the feature sets of legitimate users can be snooped for and used to launch replay or spoofing attacks. Additionally, since this communication channel connects to the matcher, it may be targeted in order to influence the resulting match score. The templates could either be corrupted to result in a low score at the matcher, or their contents could be changed to allow an intruder to obtain a high match score. Since the nature of this attack is similar to a Type 4 attack, the same prevention techniques can be employed.

2.2.6 Type 6: Attack on the Template Database

In this type of attack, database records could be modified to match the fingerprint information of the intruder and thereby fraudulently enrol them into the system as an authorised user. Alternatively, one or more of the existing records may be corrupted to result in a denial-of-service attack on authorized users. Fingerprint templates could also be stolen from the database and used to create spoofs or launch replay attacks in order to gain unauthorized access to the system (and other systems that employ the same fingerprint) [2].

The template database is the most likely target for any ambitious attacker due to two main reasons. Firstly, and most importantly, the template database contains the fingerprint information of every single user enrolled in the system. Getting their hands on such an enormous amount of personal data would put the attacker in a very powerful position to exploit the information in whatever way they desire. Secondly, the template database is static, which means that the attacker has plenty of time to access the information stored within. The difference between an attack on the template database and attacks preying on the processing modules or transmission channels lies in the amount of information that can be compromised during any single attack. While most attacks can only target one user's fingerprint features at a time, a single attack on the template database is likely to compromise *every single user* of the fingerprint recognition system. The extent of the consequences that would result from an attack on the templates stored in the system database provides strong reason to believe that this is the most serious type of attack that can be launched against a biometric system. On that account, robust protection mechanisms are needed to secure fingerprint data during storage, and this will be the focus of the research presented in this thesis. Section 2.3 will further discuss the motivation for this thesis.

2.2.7 Type 7: Attack on the Communication Channel between the Template Database and Matcher

This channel could be snooped for templates coming through from the database to the matcher; therefore, a Type 7 attack is fundamentally identical to a Type 5 attack.

2.2.8 Type 8: Attack on the Matcher

The matcher could be replaced by a Trojan horse program, which may be designed to always output either a high or a low match score, thereby bypassing the matcher. Constantly outputting a high score would result in a circumvention attack, whereby authentication would always be successful – this would be beneficial for impostors attempting illegitimate access to

the recognition system. Persistent output of a low match score would result in a denial-of-service attack, where authentication would permanently fail [5] – this would prevent authorised users from accessing the system, which would be beneficial for attackers attempting to defer the normal functioning of the system. Prevention of this type of attack has received little attention in the literature.

Note that an attack on the matcher could also involve theft of the fingerprint features that are being compared during authentication. For this reason, it is important to secure fingerprint templates during matching.

2.2.9 Type 9: Attack on the Communication Channel between the Matcher and Decision Module

The resulting match score being transmitted through this channel may be altered to suit the attacker's intentions. Encrypting the score may be a useful prevention mechanism.

2.2.10 Type 10: Attack on the Decision Module

A Trojan horse program could replace the decision module, such that the threshold for a successful match is controlled according to the hacker's intents. Prevention of this type of attack has received little attention in the literature.

2.2.11 Type 11: Attack on the Decision Output from the Decision Module

The decision made by the decision module may be overridden to suit the hacker's intents. Outputting a simple binary decision (e.g., a "1" corresponding to a "Yes" decision and a "0" indicating a "No") makes the system particularly vulnerable to this type of attack.

A prevention technique could involve the release of a complex cryptographic key instead of a binary response, whereby the correct key would indicate a successful authentication attempt and an incorrect key would imply a failed authentication attempt. Such a response is much more difficult to alter, since the adversary would not know what the correct key is.

2.3 WHY AN ATTACK ON THE TEMPLATE DATABASE IS THE MOST SERIOUS TYPE OF ATTACK

While each type of attack discussed in Section 2.2 has the potential to significantly undermine the security offered by the fingerprint recognition system, some attacks are more serious than others. The most serious attacks are those involving theft of fingerprint data, the consequences of which are most pronounced in an attack on the template database. The sheer

amount of information contained in a typical fingerprint template database makes evident the fact that an attack on the database is capable of incurring the most severe repercussions, both in terms of the security afforded by the recognition system, as well as the safety and privacy of the users of the system.

Although an attack on any database (not necessarily a fingerprint database) is generally a serious issue to contend with due to the large amounts of data that may be compromised as a result, an attack on a fingerprint database is especially severe due to the nature of fingerprint data. Since fingerprints are inherently linked to a person, compromise of a fingerprint would violate a person's security and privacy for life. An individual's security may be threatened by theft of their fingerprint data, which may lead to impersonation of that person by an adversary. Such a scenario would provide a literal definition to the expression *identity theft*. Identity theft may be carried out for the purpose of obtaining unauthorized access to restricted locations or resources, which are protected with a fingerprint recognition system, by employing a legitimate user's fingerprint to launch spoofing or replay attacks against the recognition system. More sinister motives may involve incrimination of an innocent person by using their stolen fingerprint to create an artificial spoof (such as a gummy fingerprint), which could then be used to leave fake fingerprints at a crime scene.

Access to fingerprint data stored in a system's database could also invade the privacy of the users of that system. The biological nature of fingerprints suggests the possibility of gleaning some additional personal information from the stored fingerprint measurements [5, 59]. For example, unusual fingerprint patterns have been linked to Down syndrome, Turner's syndrome, Klinefelter's syndrome, chronic intestinal pseudoobstruction (CIP), leukaemia, breast cancer, and Rubella syndrome [60, 61]. Such additional information may be extracted from fingerprint measurements and used for unintended purposes; for example, to compile statistical data or, worse, to discriminate against the perceived 'risky' sections of the population [5, 62]. Furthermore, there is the danger of unintended application scope, where fingerprint templates may be illegitimately extracted from a database and used to track people across various applications which employ the same fingerprint. This problem becomes even more serious if organizations begin sharing fingerprint data or selling it to interested parties [5, 62, 63]. The use of fingerprints (and biometrics in general) beyond the limits for which the fingerprint system was officially adopted has been referred to as *function creep* [27, 59, 63], which borders on the realm of a Big Brother scenario.

All of the aforementioned potential consequences of an attack on the template database arise from, and are intensified by, the nature of fingerprints. Firstly, a person has only a limited number of fingerprints that can be utilized, so the replacement of a fingerprint is not a

feasible option in the event of compromise [64]. Secondly, the fact that fingerprints are permanently associated with a user means that they cannot be revoked or cancelled in the event of compromise [65-67], unlike traditional security tokens (such as a credit card, for example). The third, and perhaps most serious, drawback of fingerprints is, ironically, one of the properties that contributes most highly towards the attractiveness of fingerprints for authentication properties; namely, their invariance over time [20, 67-69]. The reason for this is that, once a fingerprint is compromised, it is compromised forever, and consequently all applications that rely on the use of this specific fingerprint are likewise compromised [65].

All of these reasons combined strongly indicate the need for an effective mechanism to secure the fingerprint templates during storage in a database.

2.4 SECURING FINGERPRINT TEMPLATES DURING STORAGE IN A DATABASE

In Section 1.1, it was noted that a fingerprint is usually represented in terms of its minutiae template, and thus the database in a fingerprint recognition system typically consists of minutiae templates rather than fingerprint images. The most obvious reasons for storing a minutiae template instead of a complete fingerprint image are in order to save space in large fingerprint databases and to increase the speed of matching. Another important reason is for security purposes. The compactness of a minutiae template has understandably led to the common belief that templates do not reveal significant information about the original fingerprint image. In fact, traditionally, template generation algorithms have been assumed to be a one-way process [15], similar to a password hash [16]. Consequently, minutiae templates were not believed to contain sufficient information to enable the reconstruction of the underlying fingerprint image [17], and it was thus assumed that the storage of a fingerprint in the form of a minutiae template would provide ample security. However, several researchers have gone on to prove that minutiae templates are actually *reversible*, in that the information they provide is indeed sufficient to reconstruct the original fingerprint image; for example [7, 15, 17, 18]. As a result, the protection of fingerprint templates during storage in a database has gained considerable focus in the past few years.

The simplest way to secure a fingerprint database would be to place the entire recognition system on tamper-resistant secure hardware [70], such as a smartcard [2]. This means that the system modules, the interfaces between the modules, and the templates themselves all reside on the card [2, 26], such that all the processing takes place on the card itself. One of the advantages of using a smartcard is that template storage is distributed across system users,

thus avoiding a central template database [70]. Since the fingerprint data never leaves the card, users have full control over their templates [70]; so, it may seem that smartcards provide a suitable solution to the protection of fingerprint templates. However, the fact that smartcards are portable poses the issues of potential loss, misplacement, or theft of the card, akin to traditional security tokens (such as an EFTPOS card). These problems are made worse by the fact that there is a possibility of gleaning the stored fingerprint template from a stolen smartcard. This suggests that the protection of fingerprint templates is crucial even in tamper-resistant hardware, like smartcards; so, the simple fact that a fingerprint template resides on a smartcard instead of in a centralized database does not, on its own, guarantee security of the stored template. Therefore, robust fingerprint template protection techniques are necessary to ensure the security of stored templates.

A logical solution towards securing fingerprint templates during storage in a database would be to use traditional, well-established data-protection techniques. One such technique is cryptographic hashing, which is employed for the protection of passwords in modern-day authentication systems. Another technique is encryption, which is used in securing data during transmission across insecure communication channels. Section 2.4.1 and Section 2.4.2 explain why cryptographic hashing and encryption, respectively, are unsuitable for the protection of fingerprint templates during storage in a database.

2.4.1 Cryptographic Hashing

Although a number of different cryptographic hash functions exist, the fundamental operation remains the same. A hashing function is applied to an input of arbitrary length to produce a fixed length hash [7]. This means that a hashing function essentially produces a short, predictable summary of a large chunk of data. The hashing function can be designed to operate either in a one-way or a two-way manner [5, 7]. A two-way transformation is invertible, which means that knowledge of the function and/or its parameters can be used to recover the original data [5]. Conversely, a one-way function is mathematically non-invertible, which means that knowledge of the exact transformation and/or its parameters, as well as the hash, would not permit recovery of the original data [5, 7].

One-way hashing is typically used to secure passwords stored in a database. Password-based authentication consists of two stages: registration and authentication. During registration, the user selects a password and enters it into the system. The system then applies a one-way hash function, such as SHA-1 (Secure Hash Algorithm) [71], to the password, and the hash of the password is stored in the system database instead of the password itself.

During authentication, the *hash* of the user's newly input password is compared to the *hash* of the stored password, and the two either match or do not [67].

Since hashing works for the protection of passwords, it appears to be a reasonable solution for the protection of fingerprint templates. However, there is a fundamental difference between a password and a fingerprint template. While the password of a particular user remains the same at every presentation, there is inherent 'fuzziness' associated with fingerprint measurements [67], as mentioned in Section 2.1. This means that multiple acquisitions of the same fingerprint may generate considerably different feature sets (e.g., minutiae), as a result of intra-user variability in the acquired fingerprint image [72]. Consequently, the noisy nature of fingerprint measurements makes hashing an inappropriate solution [67, 73]. This is because cryptographic hash functions are designed to break the order of sequential data [7], which means that two fingerprint templates that differ only marginally would result in completely different hashes[5, 7, 54]. As a result, most (if not all) authentication attempts would persistently fail, which would be extremely impractical and, frankly, useless.

2.4.2 Encryption

Encryption algorithms rely on a string of bits referred to as a *key*, which essentially specifies a transformation function that is applied to the original data to transform it into a protected (encrypted) domain. The two types of key-based encryption algorithms in use today are called *symmetric* (or *private key*) and *asymmetric* (or *public key*) encryption. *Symmetric key* encryption uses the same secret key, called a *private key*, for both encrypting and decrypting a message. In a*symmetric key* encryption, the key used to encrypt a message is different from the key used to decrypt the message. The key used to encrypt a message is called the *public key*, and this key is known to everybody who wishes to communicate with a particular person. The key used to decrypt a message is referred to as the *private key*, and this key is known only to the person for whom the particular message is intended.

Once again, it appears that encryption would be a suitable answer to the problem of fingerprint template protection. However, similar to cryptographic hash functions, encryption is not a smooth function, which means that small differences in the input may result in large differences in the output [2]. So, two fingerprint templates that vary only slightly before encryption may be completely unrelated in the encrypted domain. Furthermore, since each user would have a different encryption key, we would not know what the encrypted domain for each fingerprint template would look like. These two issues combined would render matching in the encrypted domain practically impossible. This brings to mind an obvious

solution, namely, why not keep the fingerprint templates encrypted during storage in the database, but decrypt them during matching? After all, encrypted data is *meant* to be decrypted. While this solution is commonly used for the protection of fingerprint templates at the present time, it is not an effective solution simply because of the fact that encrypted data *can* be decrypted. This is because encryption is intended for securing data in transit (e.g., when passing through communication channels), not data at rest, so the idea is to decrypt the data when it reaches its destination. Consequently, using encryption for securing fingerprint templates during storage in a database constitutes an improper use of this mechanism, and the security of the protected fingerprint template would then rely entirely upon the safe storage of the decryption key. Furthermore, since encryption does not provide security once the data is decrypted [52], decrypting fingerprint templates during matching would leave them exposed and vulnerable to attacks during that stage [2]. All of these reasons combined prove that standard encryption techniques are not suitable for securing fingerprint templates [2].

2.5 SUMMARY

This chapter provided the necessary background required to justify the motivation for this thesis.

The structure and operation of a typical fingerprint recognition system were described. It was noted that a fingerprint recognition system can operate in either the *verification* mode (i.e., one-to-one matching) or *identification* mode (i.e., one-to-many matching); however, when talking about a fingerprint system in general, the generic term *recognition* is used. It was also explained that, due to the natural variability between multiple samples of the same fingerprint, a fingerprint recognition system can never be 100% accurate. Consequently, the accuracy of a fingerprint recognition system is usually measured in terms of its False Accept Rate (FAR) and False Reject Rate (FRR).

Several points of attack in a typical fingerprint recognition system were then identified, and a brief discussion of proposed countermeasures for the different types of attacks was provided. The focus of this thesis was narrowed to attacks on the fingerprint template database, with a particular emphasis on attacks involving theft of the stored fingerprint templates.

The reasons behind believing an attack on the template database to be the most serious type of attack on a fingerprint recognition system, especially when it involves theft of the enrolled fingerprint templates, were examined. The first reason relates to the fact that an attack on the template database would affect the entire population of enrolled users. The

second reason is that the biological nature of fingerprints may reveal private medical conditions. The third reason is that the stolen fingerprints may be used for purposes other than those intended by the recognition system in which the users are enrolled – this is referred to as *function creep*. The final reason is that the permanence of a person's fingerprints means that a compromised fingerprint is forever compromised, since the cancellability and replacement of a fingerprint is not an option. Bringing these reasons to light emphasized the need for effective mechanisms to protect fingerprint templates during storage in a database.

The proven reversibility of a minutiae template to the original fingerprint image was used to explain why a minutiae template, on its own, does not sufficiently protect the underlying fingerprint. Furthermore, it was stated that decentralisation of the template database, alone, is not the solution, due to the possibility of loss or theft of the smartcard on which a person's fingerprint template may be stored. The use of traditional data protection mechanisms (in particular, cryptographic hashing and encryption) for securing fingerprint templates was then explored. The non-smooth nature of these functions was found to be unsuitable to the variable nature of fingerprint data. This motivated the urgent need for fingerprint template protection schemes specifically suited to the nature of fingerprint data. The inspiration for this thesis was thus born.

Chapter 3

Literature Review on Fingerprint Template Protection Schemes

In Chapter 2, it was established that traditional data protection mechanisms (such as cryptographic hashing and encryption) are unsuitable for securing fingerprint templates due to the variable nature of fingerprint measurements. The need for fingerprint template protection schemes that are more suitable to the nature of fingerprints was thus motivated. This chapter considers the characteristics of an ideal fingerprint template protection scheme, followed by a discussion on the types of fingerprint template protection schemes that have been proposed in the literature to date.

3.1 INTRODUCTION

Researchers in the field of biometric template security in general have agreed upon a set of four characteristics that define an ideal biometric template protection scheme [5]:

1. **Non-invertibility:** It should be impossible (or at least computationally infeasible) to reconstruct the original biometric template from the protected template.
2. **Cancellability (revocability):** It should be possible to cancel (revoke) a compromised biometric template and replace it with a new template originating from the same biometric data.
3. **Diversity:** It should not be possible to cross-match a protected biometric template across different databases.
4. **Performance:** The incorporation of a biometric template protection scheme into a biometric recognition system should not have an adverse effect on that system's recognition accuracy (in terms of its FAR and FRR).

An ideal biometric template protection scheme should satisfy all four requirements. While a strategy of this calibre has thus far remained elusive, a number of creative solutions have been proposed in the literature.

Jain et al. [2] provide an extensive review of biometric template protection schemes presented in the literature up to the year 2008. Although this review was written in 2008, it still encapsulates the main characteristics of the majority of biometric template protection schemes emerging in the literature today. Indeed, judging by the number of citations this paper has received since its publication, it is fair to assume that much of the research on biometric template protection schemes today has been influenced by this review.

The popularity of the aforementioned paper may be mainly attributed to the fact that the authors provide a convenient and effective classification of biometric template protection schemes into two main categories: *feature transformations* and *biometric cryptosystems*.

A feature transformation approach essentially relies on the use of a specific function to transform a biometric template into a protected version of its former self. Depending on the characteristics of the transformation function, feature transformation approaches may be further divided into *salting* and *non-invertible transforms*.

A biometric cryptosystem incorporates ideas from traditional cryptographic protection schemes with biometrics. Its fundamental operation depends on extracting *helper data* from a biometric template in order to reconstruct a particular key, the validity of which is used for authentication decisions. Depending on the method in which the helper data is obtained, biometric cryptosystems can be further classified as *key binding* and *key generation* systems.

While categorising biometric template protection schemes is not a straightforward process, the classification proposed in [2] is very suitable and has thus been widely adopted by researchers in this field. For this reason, we have chosen to adopt this classification in reviewing the associated literature in this chapter. Borrowing Maltoni et al.'s terminology [70], the term *protected template* will henceforth be used to refer to an enrolled biometric template that has been subjected to a certain protection technique, while *unprotected template* will be reserved for the description of a biometric template in its native form (before the application of any specific protection strategies).

The remainder of this chapter discusses the general methodology behind the *feature transformation* and *biometric cryptosystem* approaches, lists several strengths and difficulties associated with the nature of these methods, and provides examples of well-known techniques that fall into these categories.

3.2 FEATURE TRANSFORMATIONS

In a feature transformation approach, the unprotected biometric template, *T*, of a user to be enrolled in the system is transformed into a protected template, *T'*, via a transformation function, *F*. The transformation function is characterized by a set of user-specific parameters, which are normally derived from a random external key or password, *K*. Thereafter, only the protected template, *F*(*T, K*), is stored in the system database. The enrolment process in a feature transformation approach is clearly illustrated on the left side of Figure 3.1. During verification, which is depicted in the right half of Figure 3.1, the same transformation function, *F*, and its governing parameters, *K*, are applied to the unprotected query feature set, *Q*, such that matching between the enrolled and query templates occurs in the transformed space, i.e., *F*(*T, K*) is matched against *F*(*Q, K*).

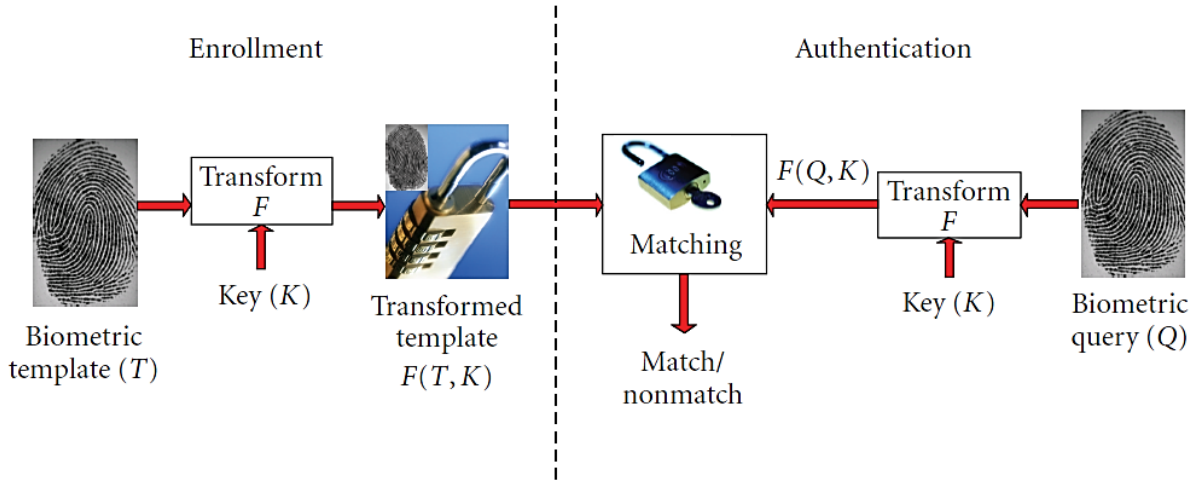


Figure 3.1: The enrolment and authentication stages in a fingerprint recognition system employing feature transformation to secure its fingerprint templates. (Image from [2])

Several advantages of the feature transformation approach readily present themselves. Firstly, the fact that the key is user-specific suggests the incorporation of diversity into the protected biometric templates, since different keys can generate multiple protected templates from the same unprotected template. Furthermore, should a protected template be compromised, it can easily be revoked and replaced with a new one by applying a different user-specific key to the same unprotected biometric data. Finally, since matching is done in the transformed domain, this means that biometric templates can remain secure even during authentication.

A difficulty faced by the feature transformation approach is dealing with the intra-user variations in the unprotected biometric template. There are typically two ways of dealing with this problem: either the transformation function must be tolerant to input variations, or the transform must leave the protected biometric template in its original (feature) space (for

example, fingerprint minutiae can be transformed into a different set of minutiae). The latter method could deal with intra-class variations by employing the same matcher on the transformed features as on the original feature set. An example of a typical form of intra-user variation is misalignment of the enrolled and query biometric feature sets. A common method of ensuring that the two feature sets are aligned is to pre-align the biometric templates prior to applying the transform to them (for example, by using the core point[1] in a fingerprint as the registration point). An alternative approach is to design a transform that produces an alignment-invariant biometric representation.

Feature transformations may be further classified into *salting* and *non-invertible transforms*, depending on the properties of the transformation function.

3.2.1 Salting

Salting is a two-factor authentication scheme, in which an unprotected biometric template is transformed into a protected template via a function specified by a user-specific external key or password. The main advantage of salting is the increase in entropy of the biometric template that is a result of the incorporation of additional information into the biometric template in the form of a key [2]. The entropy of a biometric template may be defined as "a measure of the number of different identities that are distinguishable by a biometric system" [2], hence increasing the entropy of a biometric template makes it more difficult for an adversary to guess the template, which means that there is a decrease in False Accept Rates.

The main drawback of the salting approach is that the security of this scheme relies upon the secrecy of the key or password [2, 70, 72]. This means that the transformation function is irreversible (non-invertible) only as long as the adversary remains ignorant of the key. Simultaneous availability of both the key and the protected template would enable recovery of the original, unprotected biometric template (or a close approximation of it) [2, 70, 72]. As a result, effective key management procedures must be put into place, or else the user is obliged to memorise the secret key; however, relying on users' memory for the protection of complex secret keys re-introduces the weakness of password-based schemes that we are trying to circumvent [70].

Since matching is performed directly in the transformed domain, the salting functions must be designed such that they do not have an adverse effect on the recognition performance. This becomes especially important in the presence of large intra-user variations. Salting

[1] Recall, from Section 1.1, that the core is the centre of the north most loop-type pattern in a fingerprint image, or for fingerprints that do not contain loops the core usually corresponds to the point of maximum ridge line curvature.

methods generally use quantization to deal with intra-user variability during matching in the transformed domain [2].

The most popular example of biometric salting is the *BioHashing* approach, introduced by Jin et al. [74]. BioHashing is a two-factor authentication method, which is based on iterative inner products between biometric feature vectors and token-derived random number sequences that are generated by a unique hash key. The BioHashing procedure was initially proposed for the fingerprint modality, and it consists of two stages. Firstly, the extracted fingerprint feature vector is transformed into a translation, rotation, and scale invariant feature set, employing the Wavelet Fourier-Mellin Transform (WFMT)[2]. Secondly, the resulting data is discretised via an inner product computation between the invariant feature vector and a tokenised pseudorandom number sequence [74]. The second stage of this process produces the protected biometric template vector, which is referred to as a BioHash [74].

Figure 3.2 portrays the creation of a user's BioHash vector, i.e., the enrolment stage (note that it is assumed that the biometric feature vector of the user has already been obtained and made translation, rotation, and scale invariant). During enrolment, each user is presented with a secret seed, K (hash key), which is stored on an external device such as a USB token or a smart-card. The seed is used to generate a set of m pseudorandom vectors, $\boldsymbol{r}$, and these random vectors constitute the "salt" of the BioHashing scheme. The vectors are orthonormalised using the Gram-Schmidt orthonormalisation method, after which the dot products between the invariant biometric feature vector, $\boldsymbol{x}$, and the orthonormal set of vectors, $\hat{\boldsymbol{r}}$, are calculated. The resulting vector is binarised in order to account for intra-user variations, where the binarised vector, $\boldsymbol{b}$, constitutes the protected biometric template (BioHash). The binarisation is computed based on a pre-set threshold, τ, where 0 corresponds to a dot product that is less than or equal to τ, while 1 represents a dot product greater than τ. The threshold, τ, is selected based on the criterion that the expected number of zeros in the resulting BioHash vector, $\boldsymbol{b}$, is equal to the expected number of ones, in order to maximize the entropy of the protected template, $\boldsymbol{b}$ [2]. During verification, the invariant query biometric feature set is transformed in the same fashion, and the resulting bit vectors are compared using Hamming distance.

[2] Take the FFT of an image – the resulting spectral magnitude is *translation invariant*. Since we want our image to be rotation and scale invariant as well, define rotation and scale in terms of translation. Do this by first defining the spectral magnitude in terms of polar coordinates, in order to decouple rotation and scaling. Rotation is now expressed in terms of translation. Reduce scaling to a translation by expressing the radial coordinate in terms of a logarithmic scale. The resulting image is now translation, rotation, and scale invariant.

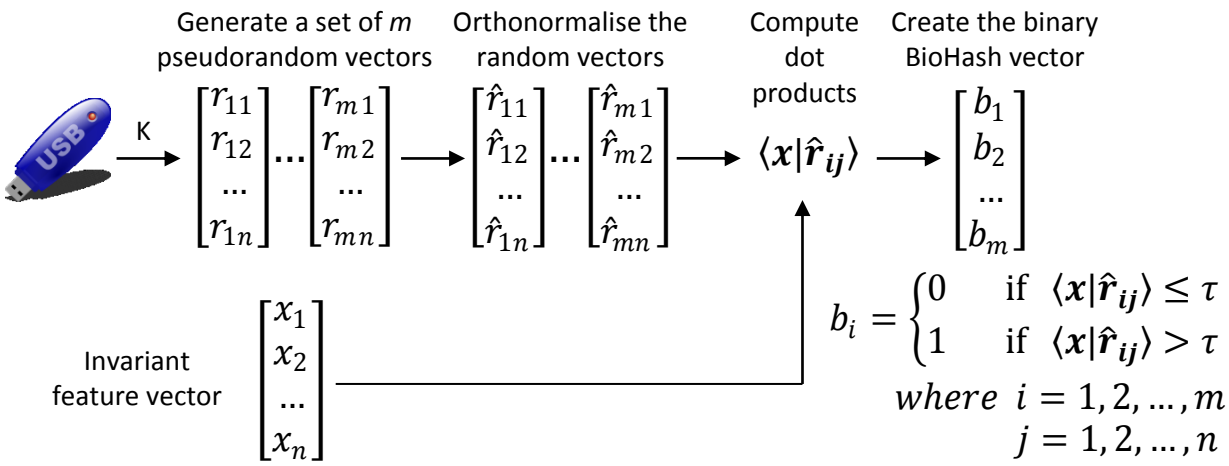


Figure 3.2: The creation of a user's BioHash (enrolment). Note: In this figure, letters written in bold represent vectors.

Variations of the BioHashing approach have been applied to several biometric modalities, including fingerprints (e.g., [74, 75]), face (e.g., [64, 76-78]), palm prints (e.g., [79-81]), and iris (e.g., [82]), where the differences essentially lie in the method used to create the invariant feature vector, which depends on the characteristics of the biometric modality involved.

The BioHashing procedure has been proven to be advantageous in several ways. Firstly, BioHashing simultaneously provides high intra-class variation and extremely low inter-class correlation, which essentially leads to an Equal Error Rate (EER) of zero (when the legitimate token is used). This means that the occurrence of a False Accept is eliminated without a corresponding increase in the FRR [64, 74, 76, 77, 79, 81, 83-88]. It has also been claimed that BioHashing has a high tolerance to data capture offsets, such that the same biometric trait acquired at different times will produce highly correlated bit strings (BioHashes) [74, 76, 81, 83]. This is due to the invariance of the feature vector created during the first stage of the BioHashing process, as well as the subsequent discretization of the invariant feature vector in the second stage. Another advantage of BioHashing is that it addresses the problem of irrevocability of biometric features: a user's compromised BioHash can be easily revoked and replaced with a new one by using a different secret seed for enrolment [74]. Finally, it has been claimed in several literary publications [64, 74, 76, 77, 79, 81, 88] that the BioHashing procedure is a one-way transformation, in that it is impossible to glean information about the original biometric template from the BioHash, without simultaneous access to both the BioHash and the user's secret token.

Unfortunately, some important drawbacks of the BioHashing scheme have subsequently been presented. The most commonly analysed limitation of the BioHashing approach is the degradation in matching performance when an adversary has access to a user's secret key (seed) and uses the legitimate key with their own biometric features in order to fool the

system into authenticating them [84, 87, 89-94]. For example, Kong et al. [90, 91] analysed the claim of a zero Equal Error Rate in the base BioHashing method [74] as well as several of its variants [64, 76, 79-81, 83]. The authors concluded that the outstanding achievement of zero EER in BioHashing and its variants is based on the impractical hidden assumption that the token containing the secret seed would never be revealed to an adversary by way of theft, loss, duplication, or sharing. Kong et al. [90, 91] then went on to show that the resulting performance of the recognition system in the case where an impostor uses a genuine user's token and his/her own biometric data is worse than that obtained using solely biometrics. This is referred to as the stolen-token scenario, which is facilitated by the quantization of biometric features and dimensionality reduction [95]. Another shortcoming of BioHashing is that it is easy to invert when a user's key is known to an adversary, such that the original biometric template (or a close approximation of it) can be recovered from the user's BioHash [89, 95-97] and used to fool an authentication system. Yongjin et al. [96] even proved that a lost BioHash on its own (without the genuine user's private transform) is enough for an impostor to fool the authentication system. It has also been suggested [98] that the combination of different BioHashes of the same user can leak important information about the original biometric feature.

Several researchers have presented methods for overcoming the performance degradation resulting from a stolen-token scenario, for example [84-88, 92-94, 97, 99-102].

Since a salting approach is by nature invertible, hardly any existing literary works focus on improving the non-invertibility property of BioHashing; however, two suggestions are presented in [95, 97]. In fact, BioHashing on its own technically cannot be made to be non-invertible. A hybrid protection scheme, incorporating techniques other than salting, would be required; for example, applying BioHashing to a non-invertible template.

Other salting techniques, which do not adopt BioHashing, are also available in the literature; for example [103-105].

3.2.2 Non-invertible Transforms

As its name implies, this approach secures a biometric template by applying a non-invertible transform to it. A non-invertible transform indicates near impossibility on obtaining the original biometric data from its transformed version. The parameters of the transformation function are specified by a key, yet knowledge of the key and/or the transformed template does not facilitate recovery of the original biometric template [2, 70, 72]. This is the main advantage of the non-invertible transform approach compared to the salting approach, and it means that biometric templates that are protected using non-invertible transforms are

generally more secure than those protected using the salting approach [2]. A related advantage of the non-invertible transform approach is that, unlike salting, it does not require storage or memorising of any secret information. Another positive aspect of non-invertible transforms is that they tend to leave the protected biometric template in the same feature space as its unprotected counterpart. In this case, intra-user variations in the transformed biometric templates can be robustly handled by using existing, sophisticated matchers, thereby reducing the error rates of the biometric system [95]. Furthermore, the obtained match score is comparable to that obtained in the original space, and can thus be employed in the design of a secure multibiometric system via score-level fusion methods [72].

The major limitation of the non-invertible transformation method lies in the difficulty of designing a good one-way function. The transformation function should ensure that biometric features from the same user retain high similarity in the transformed space, while features from different users are completely unrelated after transformation. However, the transformation should also be non-invertible, such that an adversary is unable to glean any information about the original biometric template from its protected counterpart. There is a trade-off between discriminability and non-invertibility, since it is challenging to design transform functions that satisfy both requirements simultaneously. Consequently, often the greater the amount of distortion applied to the original biometric data by the transformation, the worse the recognition performance among the protected biometric templates. This means that the non-invertible transform approach typically suffers from a security versus performance trade-off. Furthermore, the transformation function is dependent to a large extent on the biometric features to be employed in a specific application [2]. This analysis makes evident a clear comparison between the salting and non-invertible transform approaches. While salting schemes (such as BioHashing) generally tend to either preserve or improve the recognition performance of the biometric system into which they are incorporated, non-invertible transforms often have the effect of degrading the recognition accuracy somewhat. On the other hand, non-invertible transforms tend to impart more security to the protected biometric templates compared to salting approaches, which are invertible with the revelation of the user-specific key.

The notion of applying non-invertible transforms to biometric templates for the purpose of converting them into a protected form was pioneered by Ratha et al. [20]. Ratha et al. introduced the concept of *cancellable biometrics* in order to alleviate the issue of permanence of biometric features in the event of compromise. This method essentially involves "an intentional, repeatable distortion of a biometric signal based on a chosen transform" [20], where the transformation applied to the biometric signal at enrolment is used to distort the

biometric signal in the same manner for every subsequent authentication. The transformation can be applied to the biometric signal either in the signal domain (whereby the signal is distorted directly upon acquisition) or the feature domain (where the biometric signal's extracted features are transformed). The authors suggest grid morphing and block permutations as examples of signal level transforms [20]. The left side of Figure 3.3 illustrates grid morphing on a face image. Here, a grid is overlaid on the person's original face image, and it is subsequently distorted to transform the underlying face image into an unrecognizable form of its original self. The right side of Figure 3.3 depicts block permutation on a fingerprint image. The original fingerprint image is tessellated into a grid, and the transformation involves scrambling the grid cells (or *blocks*) to create the protected fingerprint image. Similar transforms have been demonstrated on the iris and speech patterns in [69].

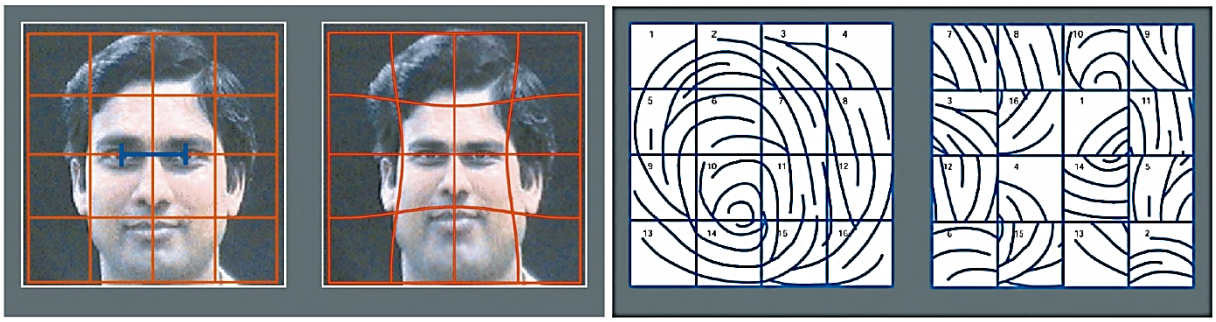


Figure 3.3: (*Left*) Grid morphing applied to a face image; (*Right*) Block permutations applied to a fingerprint image. (Images from [40])

Examples of feature domain transforms include a set of random, repeatable perturbations of feature points (which is essentially the same concept as block permutation, except that extracted features, instead of parts of the whole image, are now being scrambled) [20], and a high order polynomial function [20, 69]. The latter method is illustrated by mapping a set of minutiae x-coordinates to different values via a high order polynomial, where the mapping is many-to-one. This means that, while it is easy to compute the transformation, it is computationally difficult to reverse it to obtain the original minutiae coordinates, since the reverse mapping is one-to-many. Hence, this function is non-invertible.

As a result of its non-invertibility property, the cancellable biometrics concept preserves privacy since the recovery of the original biometric from its transformed version is practically impossible or extremely difficult computationally [65]. Moreover, the cancellable biometrics scheme provides diversity among protected biometric templates, since each instance of enrolment can apply a different transform (or the same transform but using different parameters). This mitigates the issue of cross-matching across databases. Furthermore, a protected template that is compromised in some way can simply be cancelled and a different

transformation can be used to generate a new version of the biometric template, thereby essentially enrolling the user as a new person. This means that the cancellable biometrics technique provides revocability of biometrics data [20, 65]. Finally, and importantly, the transformation (both signal and feature domain) does not change the feature representation; for example, transformed minutiae points are still represented as minutiae points. Therefore, existing feature extraction and matching algorithms can be employed on the transformed biometric features, and the approach is backward compatible with current biometric authentication systems [65].

Ratha et al. [65] expanded upon their concept of cancellable biometrics by proposing and analysing three feature domain non-invertible transforms for generating cancellable fingerprint templates. Their approach initially locates singular points (cores and deltas) in fingerprint images, and these points are used for alignment purposes. The minutiae points are then transformed with respect to the core via three functions: Cartesian transformation, polar transformation and functional transformation. In the Cartesian transformation, depicted in Figure 3.4, the salient minutiae points in a fingerprint image are first identified. Subsequently, the fingerprint image is tessellated into a rectangular grid, where each cell may contain some minutiae. The transformation then involves shuffling the cells in an irreversible manner (based on a user-specific key or password), such that several cells are mapped to the same position.

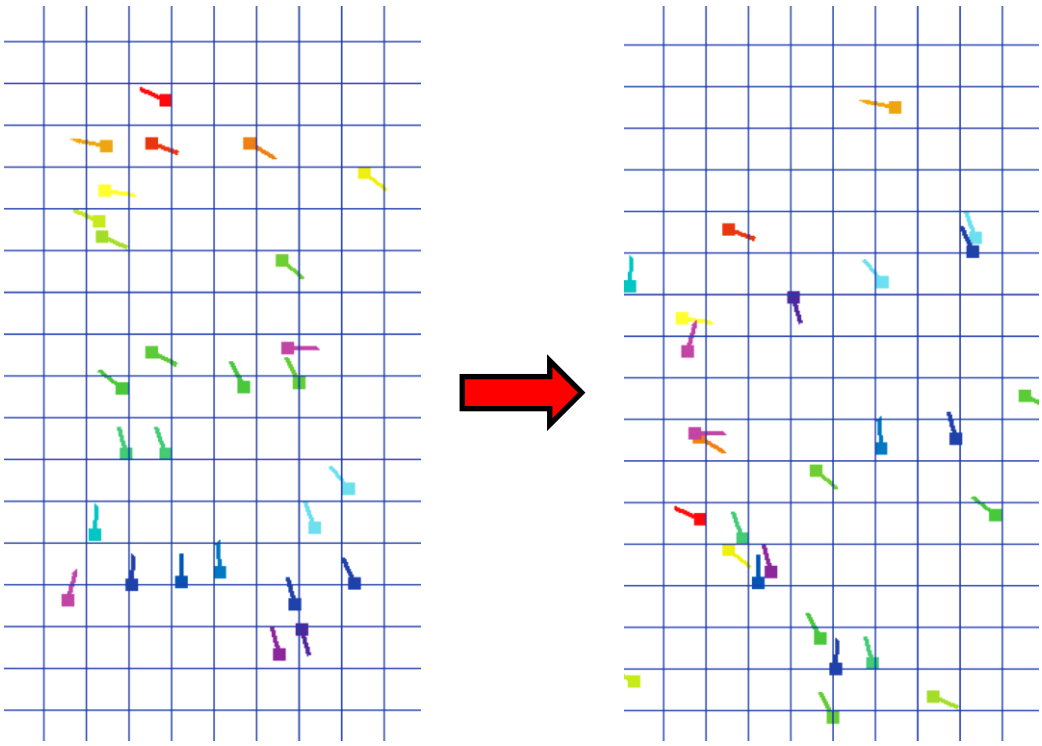


Figure 3.4: Cartesian Transformation. (*Left*) A fingerprint's minutiae points are overlaid onto a rectangular grid; (*Right*) the grid cells are shuffled to produce the transformed fingerprint template. (Images adapted from [2])

The polar transformation, illustrated in Figure 3.5, is similar to the Cartesian transformation, except that the minutiae positions are now expressed in terms of polar coordinates relative to the core, and the minutiae space is tessellated into a number of shells. Each shell is further divided into several polar sectors. Since the sector size expands with increasing distance from the core (different shells have differently-sized sectors), restrictions are placed on the translation parameters such that each transformed sector remains in close proximity, in terms of radial distance, to its original position [2]. This means that, although both transforms are non-invertible in that the reverse mapping is one-to-many, the polar transformation is slightly weaker than the Cartesian transformation in terms of security [65]. However, the polar transformation tends to have higher recognition accuracy due to the fact that it preserves the natural distribution of minutiae points to a higher degree than the Cartesian transformation [65].

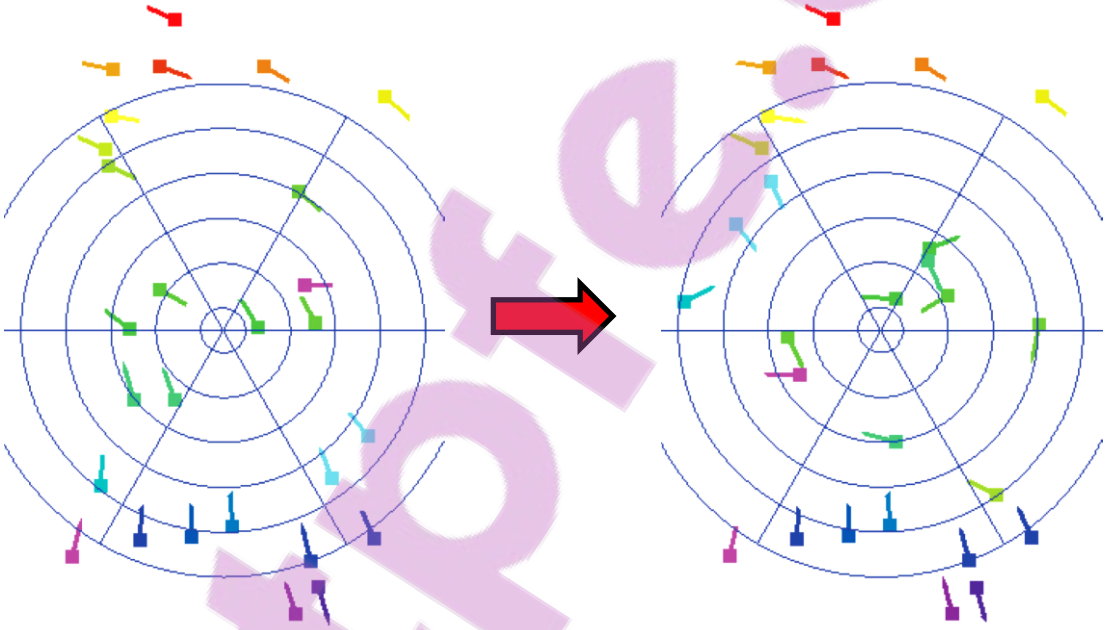


Figure 3.5: Polar Transformation. (*Left*) A fingerprint's minutiae points are overlaid onto a radial grid; (*Right*) the grid shells and sectors are shuffled to produce the transformed fingerprint template. (Images adapted from [2])

The main shortcoming of both the Cartesian and polar transformations is that a small variation in minutiae positions in the original fingerprint may result in a large change in minutiae positions following the transformation, if the minutiae points cross a sharp boundary [65]. This would increase intra-user variation in the transformed biometric template, thereby having an adverse effect on the matching performance. In order to deal with this issue, the authors proposed an alternative: a "locally smooth but globally not smooth" [65] functional transformation. The "locally smooth" aspect helps to preserve local distances between neighbouring minutiae and thereby maintain matching accuracy, while the "globally not smooth" property provides ambiguity in reversing the transform and thereby ensures that the

function is non-invertible [70]. The proposed functional transformation, portrayed by Figure 3.6, is essentially a surface folding transform, which consists of *folding* regions arising from multiple locations in the original minutiae space. This may be visualised as the embedding of minutiae points into a 'sheet', and the subsequent 'crumpling' of the 'sheet' to produce the transformed fingerprint template. This means that several locations in the original minutiae space are mapped to the same location in the transformed space [70]. One suggestion for this transform (i.e., the 'crumpling') is an electric potential field parameterized by a random 2D charge distribution; the other proposition is a mixture of 2D Gaussian kernels. These functions are evaluated at the minutiae locations to obtain the translation corresponding to that minutia [95]: the magnitude of the functions at the point of occurrence of a particular minutia defines the extent of translation of that minutia, while the gradient indicates the direction of the translation [2, 65].

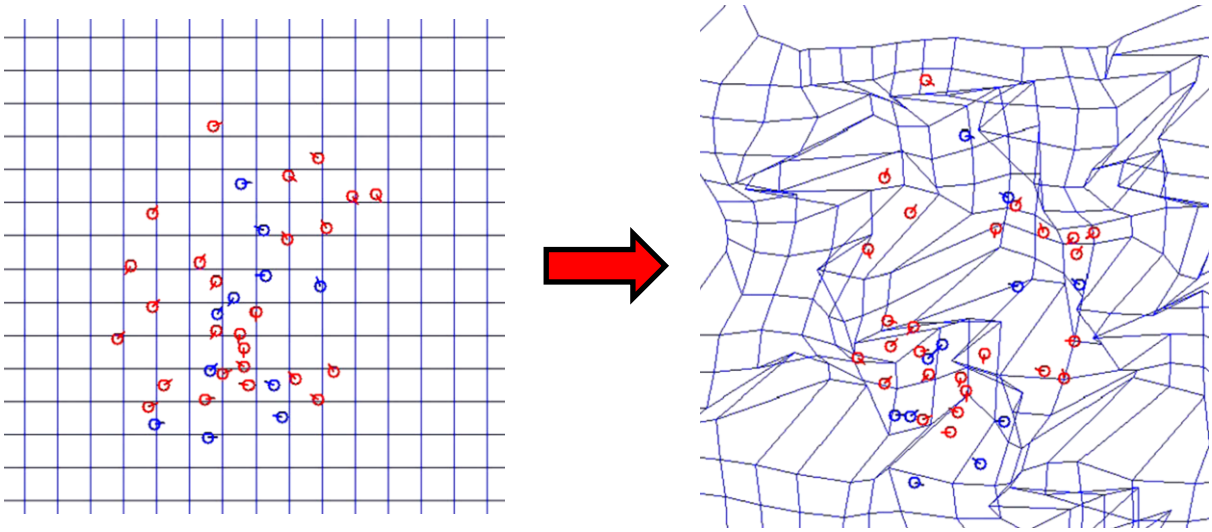


Figure 3.6: Functional Transform. (*Left*) A fingerprint's minutiae points are embedded into a 'sheet'; (*Right*) the 'sheet' is then 'crumpled' to produce the transformed fingerprint template. (Images adapted from [3])

The functional transformation preserves the local structure of the minutiae to a large extent, due to its "locally smooth" property. However, upon conducting experiments to measure the degree of 'folding' offered by the surface folding transform, Ratha et al. [65] found that only a small fraction of minutiae (8%) have their neighbourhood perturbed by this transformation. Therefore, despite its preservation of matching accuracy, the non-invertibility offered by the proposed functional transform approach is not very strong [70]. Having said that, the authors [65] concluded from experimental analysis that the surface folding transformation outperforms the Cartesian and polar transforms both in terms of recognition accuracy and security strength. Unfortunately, transformations such as the ones proposed by Ratha et al. [65] require the biometric templates (in this case fingerprints) to be aligned prior to the transformation; misalignment can result in further increase in intra-user variation [95].

Feng et al. [106] suggested that the transforms proposed by Ratha et al. [65] are weak in terms of their many-to-one mapping properties. They proved this claim by demonstrating that this weakness can be easily exploited to obtain the original fingerprint templates. Having access to two or more transformed versions of the same minutiae pattern, the authors showed that it is possible to identify the original minutiae points, thereby proving that the schemes presented in [65] are vulnerable to a Record Multiplicity Attack. Similarly, Shin et al. [107] showed that Ratha et al.'s surface folding transform [65] is invertible if two transformed templates originating from the same fingerprint are used to launch a dictionary attack to determine the original minutiae points. Nagar et al. [95] developed six metrics that can be used to evaluate the security strength of feature transformation approaches, and they used these to evaluate the security strength of Ratha et al.'s mixture of Gaussians based transformation [65]. Their analysis showed a number of drawbacks of this method. Firstly, there is a significant degradation in matching performance following the transformation compared to that achieved with unprotected minutiae, and the extent of degradation increases with increasing distortion of the minutia points. Secondly, matching performance decreases even further when an adversary knows a genuine user's key (transformation parameters). Thirdly, the feasibility of intruding a different biometric system that employs the same fingerprint using the template inverted from the current system is very high. Finally, there is an extremely high possibility of successfully cross-matching two templates obtained from the same fingerprint but transformed using different transformation parameters. The degradation in matching performance has been attributed to an increase in FRR, which is mainly caused by misalignment of minutiae prior to applying the transformation function. The authors' analysis also demonstrates that, the more secure a transformation is, the less its degree of usability, thus exhibiting the commonly encountered issue of a trade-off between security and usability in biometric template protection. Their evaluation further indicates the importance of designing a transformation function that is not only computationally difficult to invert to recover the original template, but using which it is also computationally hard to obtain the pre-image[3] of a transformed template, in order to prevent intrusion into other biometric systems that employ the same biometric trait and to mitigate cross-matching (linkage) attacks. This conclusion stems from the fact that vulnerabilities in Ratha et al.'s cancellable fingerprint template scheme [65] arise from the ease with which an impostor is able to obtain the pre-image of the transformed template; while it is computationally difficult to obtain the

[3] The pre-image of a transformed template may be defined as "the collection of all the templates in the original domain that can generate the given transformed template" [95] A. Nagar, *et al.*, "Biometric Template Transformation: A Security Analysis," in *Media Forensics and Security II*, San Jose, CA, USA, 2010.

original fingerprint template from the pre-image, the pre-image itself suffices in the execution of linkage and intrusion attacks [95].

Inspired by Ratha et al.'s concept of *cancellable biometrics*, several other researchers have attempted to design non-invertible transformations for the protection of biometric templates, some of which do not require pre-alignment of biometric features. Examples can be found in [3, 20, 65, 69, 105, 108-151]. Note that the fingerprint-based non-invertible transforms proposed in [3, 20, 65, 69, 108, 109, 111, 112, 132-151] are investigated in more detail in Chapter 4.

3.3 BIOMETRIC CRYPTOSYSTEMS

A biometric cryptosystem incorporates ideas from traditional cryptographic protection schemes with biometrics. The initial motivation behind the merging of biometrics with cryptosystems was for the purpose of either using biometric features to secure a cryptographic key or for directly generating a cryptographic key from the biometric features themselves [2, 70, 152]. However, it has since been realised that biometric cryptosystems can also be employed in the protection of biometric templates [2].

The fundamental idea behind biometric cryptosystems is to store a small amount of information, referred to as *helper data,* about the unprotected biometric template [2], which can be used for matching purposes while simultaneously securing the biometric data. For this reason, biometric cryptosystems are also known as helper data based methods [2]. The helper data is ideally not supposed to reveal any information about the original biometric template, but it is employed during matching to extract a cryptographic key from the query biometric features. Verification success is determined by the validity of the extracted key, and error correction codes and quantisation are commonly used as a means of dealing with intra-user variability in the biometric features [2].

The main operational difference between biometric cryptosystems and the feature transformation approach is that, while feature transformations rely on the non-invertibility of the transform function (in the case of non-invertible transforms) or on the secrecy of the transformation parameters (for the salting approach) to impart security to the biometric templates, in the case of biometric cryptosystems security of the protected templates is dependent upon the amount of information revealed by the helper data about the corresponding unprotected biometric templates [2]. Furthermore, despite their applicability in biometric template protection schemes, biometric cryptosystems in general are not designed with the intention of providing diversity and revocability to the protected biometric templates

[2, 70]. This is in contrast to the feature transformation approach, which exhibits diversity and revocability among its chief advantages.

Depending on the method in which the helper data is generated, biometric cryptosystems can be further categorized into *key binding* and *key generation* systems [2].

3.3.1 Key Binding

Key binding approaches have also been referred to as *Biometric Encryption* [153]. In a key binding approach, the helper data consists of the biometric template monolithically bound with an external cryptographic key, where the key is independent of the biometric features. The helper data, from which it is computationally difficult to extract either the original biometric template or the cryptographic key, is stored in the database. Therefore, the helper data is essentially a publicly available protected template, which secures both the biometric template and the key. Since the design of the helper data relies on the biometric features to be employed and the nature of their intra-user variations, a considerable amount of effort must be invested into ensuring that the helper data is suitably constructed [2]. The helper data is typically generated via association of the enrolled biometric template with a codeword, where the codeword is obtained from an error correcting code using the external cryptographic key as the message. An erroneous codeword recovered from a query biometric template that is similar to the enrolled template is corrected using error correction codes. The corrected codeword can subsequently be decoded to recover the embedded cryptographic key. Successful biometric verification relies on the exact recovery of the key from the helper data using the query biometric features [2, 70].

The main advantage of a key binding cryptosystem is its tolerance to intra-user variation in acquired biometric features, which is provided via the error correction capabilities of this approach [2, 70]. The most important limitation of this method is its inability to use existing matchers developed exclusively for comparison of original biometric templates. This is because key binding cryptosystems require matching to be performed in the encrypted domain, using error correction schemes [2, 70]. As a result, matching accuracy may degenerate in situations where intra-user variation increases beyond the error correcting capability of this cryptosystem. Furthermore, despite the fact that a key binding cryptosystem is generally not designed with the intention of providing diversity and revocability to the protected biometric templates [2, 70], this approach may be adapted to take these two properties into account, and indeed some efforts in this direction have already been made (see Section 3.4). Changing the external key and the way in which it is monolithically

bound to the unprotected biometric template may also help to introduce both diversity and revocability (to some extent) into key binding cryptosystems.

The most popular key binding approach is the fuzzy vault scheme, which was proposed by Juels and Sudan [154] and is closely related to the fuzzy commitment scheme, proposed by Juels and Wattenberg [155]. The popularity of the fuzzy vault scheme stems from its ability to overcome the difficulty of order dependence of the elements in a biometric template, which is considered to be the main shortcoming of Juels and Wattenberg's fuzzy commitment scheme [155]. During enrolment, a user's external secret key is bound with their biometric features to construct and evaluate a high-order polynomial. This is denoted as "locking" the vault. For instance, the coefficients of a polynomial, P could be fixed according to the user's secret key, K, whereupon the polynomial is evaluated at each element of the unprotected biometric template, T (the template elements are treated as distinct x-coordinate values). Finally, some noise is added in the form of "chaff points", which are simply random points whose values do not lie on the polynomial, P, to derive the final point set, V, which constitutes the fuzzy vault for this particular user. Note that the chaff points are included for the purpose of hiding the polynomial from an attacker; therefore, the addition of these random points constitutes the security of the fuzzy vault scheme. The resulting fuzzy vault is stored in the database as helper data, which is employed in authenticating this user at a later stage. Figure 3.7 illustrates the enrolment process in a fingerprint fuzzy vault scheme.

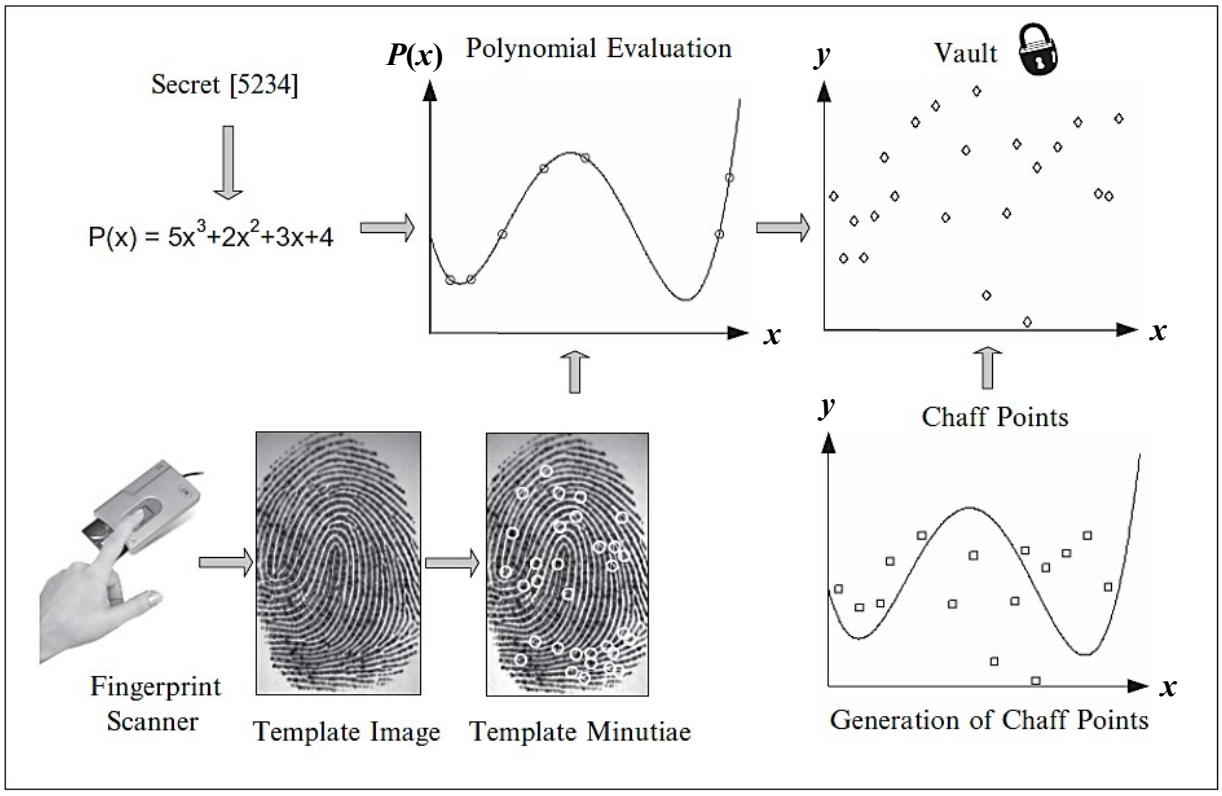


Figure 3.7: The enrolment process in a fingerprint fuzzy vault scheme. (Image adapted from [70])

In the enrolment process illustrated in Figure 3.7, a user's external secret (the number sequence 5234) is combined with the minutiae points extracted from their scanned fingerprint image to form a polynomial, $P(x)$. The numbers in the user's secret sequence form the coefficients of the polynomial, while the minutiae points constitute the polynomial's abscissa values (x-coordinates). The polynomial is evaluated at the minutiae points, and subsequently some random chaff points (which do not lie on the polynomial) are added. The resulting point set constitutes the fuzzy vault of this particular user.

During verification, an unprotected query biometric template, Q, is presented to the system. Note that pre-alignment of the enrolled biometric template, T, and the query biometric template, Q, is assumed. If the elements of Q are sufficiently similar to the elements of T (regardless of their ordering) within some error tolerance, then Q can be corrected using error correcting codes to match the template T, which can then be used to reconstruct the polynomial, P, and thereby obtain the secret key, K. This is referred to as "unlocking" the vault. Release of the key, K, indicates successful polynomial reconstruction, which signifies a successful authentication attempt. Figure 3.8 depicts the verification stage in a fingerprint fuzzy vault scheme.

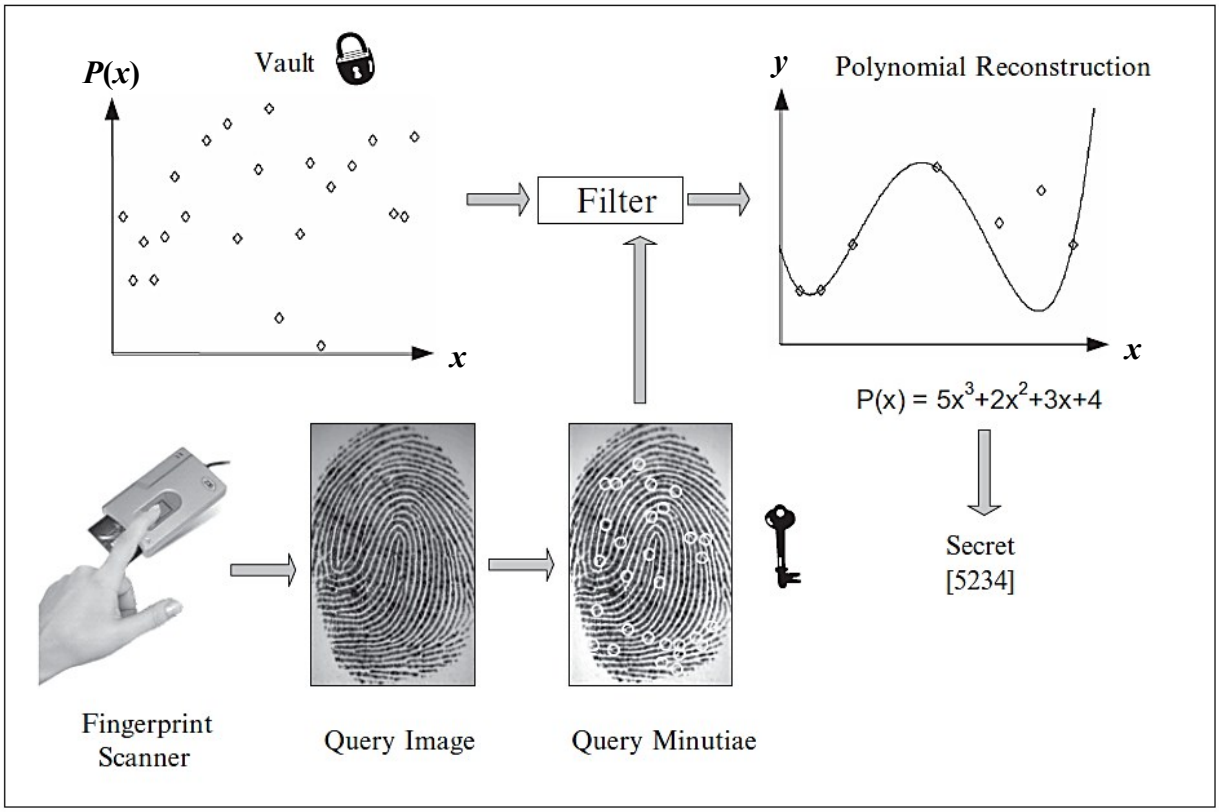


Figure 3.8: The verification stage in a fingerprint fuzzy vault scheme. (Image adapted from [70])

During verification, as illustrated by Figure 3.8, the user scans their fingerprint and claims an identity, upon which the corresponding fuzzy vault is retrieved from the database. The

user's query minutiae set, extracted from their newly scanned fingerprint, is used to identify a set of potentially matching minutiae in the retrieved vault (this process is depicted by a filter in Figure 3.8), and these points are then used to "unlock" the retrieved fuzzy vault (the unlocking mechanism is depicted by the polynomial reconstruction). If the query minutiae set is similar enough to the minutiae set that was used to construct the fuzzy vault that has been retrieved from the database, then the user's query minutiae set will be able to identify enough true points that lie on the secret polynomial. In this case, the polynomial will be reconstructed and the user's secret key (5234) will be extracted, thereby signifying a successful verification attempt. Alternatively, if the query fingerprint is not similar enough to the fingerprint used to construct the fuzzy vault during enrolment, then the query minutiae set would not be able to "unlock" the vault; hence, verification would fail since the user's secret would not be able to be extracted.

The security of a fuzzy vault essentially relies on the difficulty of the polynomial reconstruction problem: if an adversary does not know many points that lie on the polynomial, P, then they will be unable to find the parameters of P that are necessary for the polynomial's reconstruction. The polynomial reconstruction problem is controlled by two main components: the number of true points making up the polynomial, and the number of chaff points added to conceal the true polynomial, where the number of chaff points is the more influential factor in determining the security robustness of the proposed scheme. The security of the fuzzy vault scheme is proportional to the number of chaff points added during the enrolment stage; the more chaff points that are generated, the more spurious polynomials appear, thereby concealing the true polynomial to a greater extent. Additionally, a biometric template with a large number of elements will have more coefficients with which to construct the polynomial, P, which means that the degree of the polynomial will be higher than that for a small biometric template. A higher degree polynomial would increase the security robustness of the resulting fuzzy vault [70, 154]. An advantage of the fuzzy vault scheme is the ability to control the amount of security imparted to protected biometric templates by increasing the number of chaff points and consequently the difficulty of the polynomial reconstruction problem.

The fuzzy vault scheme proposed by Juels and Sudan [154] has gained a place amongst the most widely adopted approaches for the protection of biometric templates, having been implemented on a number of biometric modalities [2]. Examples include fingerprints [156-167], iris [168], face [169], signature [170], and a multibiometric system consisting of fingerprints and iris [171].

Despite its popularity, analysis of the fuzzy vault scheme has indicated that this approach has several serious drawbacks. Chang et al. [172] present a method of distinguishing the genuine points (specifically, fingerprint minutiae) from the chaff points in a fuzzy vault that employs fingerprint minutiae as a locking set. Their method is essentially based on the idea of *free area* of the random chaff points, where chaff points that are added later on tend to have more neighbouring points (and thus a smaller free area) than points added earlier on in the chaff generation process. Consequently, the authors suggest a way in which an adversary can utilize this information to separate the genuine minutiae points from the chaff points, by giving higher priority to points with a larger free area during the search. While Chang et al. [172] were unable to prove their method analytically, they do present empirical proof that suggests that their proposed technique finds the genuine minutiae points in a fuzzy vault faster than a brute force search.

Scheirer and Boult [173] discussed the vulnerability of fuzzy vaults to three potential attacks, namely, attacks via record multiplicity (ARM), surreptitious key-inversion (SKI) attacks, and blended substitution attacks. The authors suggest that a fuzzy vault is particularly vulnerable to ARM attacks, where access to two or more fuzzy vaults generated from the same biometric data, but with different keys and chaff points, would enable an adversary to easily identify the genuine points in the two vaults and thereby decode the vault [2, 70, 173]. Therefore, the fuzzy vault scheme does not provide diversity and revocability [2]. This means that, if a fuzzy vault is compromised, a new vault cannot be created from the same biometric data by simply binding it with a different key [174]. Furthermore, the vulnerability of fuzzy vaults to ARM attacks allows cross-matching of templates across different systems, thereby user privacy is not ensured [174]. In a stolen key-inversion attack, if an attacker is able to recover the secret key embedded in the fuzzy vault (for example, through snooping), the secret polynomial may be directly reconstructed; thereby, the unprotected biometric template can be easily separated from the chaff points [2, 70, 173]. A blended substitution attack is straightforward if an adversary is able to modify an existing fuzzy vault. In this attack, an impostor takes advantage of the myriad chaff points existing in the fuzzy vault to substitute some of these random points with his or her own biometric data, in which case both the legitimate user and the impostor would be able to be identified using the same fuzzy vault [2, 70, 173]. Kholmatov and Yanikoglu [175] extended Scheirer and Boult's work [173] by presenting experimental evidence that confirms a fuzzy vault's vulnerability to record multiplicity (correlation) attacks.

Nandakumar et al. [174] further mention the possibility of exploiting the non-uniform nature of biometric features to launch an attack on a fuzzy vault based on statistical analysis

of points in the vault. The authors also note the vulnerability of a fuzzy vault to attacks during the authentication stage, where a genuine user's original template is temporarily exposed and therefore vulnerable to snooping [5].

Another liability of the fuzzy vault scheme is the considerable increase in biometric template size as a result of the addition of a large number of chaff points [5]. This may be undesirable in recognition systems that require a small template size. Furthermore, recognition accuracy may be adversely affected as a result of the large number of false points or too few true points in the protected template. For example, Clancy et al.'s implementation of the fuzzy vault scheme on fingerprint minutiae [156] showed an unacceptably high FRR of 20-30%.

Examples of key binding techniques that are not based on the fuzzy vault scheme include [155, 176-180].

Note: *The fuzzy vault scheme currently appears to be the most popular fingerprint template protection scheme in the literature, which is probably largely due to the interesting concept on which it is based. Since the aim of this thesis is to present a new point of view on the challenging task of fingerprint template protection, our focus is not on the fuzzy vault scheme. Nevertheless, the initial stages of this research involved a detailed study of this fingerprint template protection strategy, and thus a contribution to this particular area has been made in the form of the publication in [25], which is replicated in Appendix B. This publication presents a dissection of the methods used to implement fingerprint-based fuzzy vaults in the literature. The purpose of the aforementioned publication is to assist interested researchers in their own implementations of the fuzzy vault framework in the context of the fingerprint biometric. We believe that this is an important contribution, since researchers are often faced with the intimidating task of attempting to implement an accurate rendition of someone else's work. The most difficult part in this endeavour is knowing where to start. It is our hope that the material presented in [25] and in Appendix B will help lay the groundwork for this undertaking.*

3.3.2 Key Generation

In a key generation biometric cryptosystem, the idea is to generate a cryptographic key directly from the biometric data, rather than binding an existing key with the biometric template as in key binding biometric cryptosystems [2, 70]. This means that the helper data in this case is derived solely from the biometric template, and cryptographic key generation relies only on the helper data and the query biometric template [2]. Figure 3.9 depicts the

enrolment and authentication stages in a key generation biometric cryptosystem. During enrolment, a user's unprotected biometric template, T, is used to generate a cryptographic key, using some function, F. Subsequently, the generated key is stored in the database as helper data, $H = F(T)$. During authentication, the user's unprotected query biometric template, Q, is processed in the same manner to generate the (hopefully same) cryptographic key. The validity of the newly generated key is assessed by comparing it against the key stored as helper data during enrolment. If the two keys match, then authentication is successful.

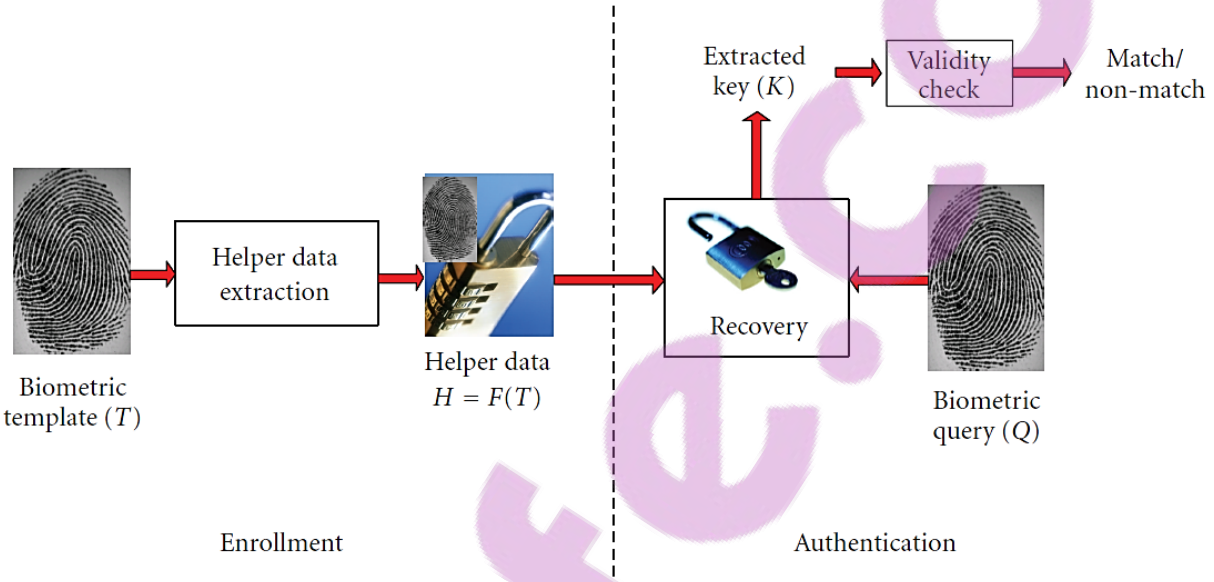


Figure 3.9: The enrolment and authentication stages in a key generation biometric cryptosystem. Note that this process would be similar in a key binding biometric cryptosystem, except that the helper data would be constructed using both the biometric template and an external key, *K*, i.e., *H = F(T, K)*. (Image from [2])

The attractiveness of this approach lies in its potential to be useful in both the protection of biometric templates and in cryptographic applications [2]. Unfortunately, fulfilment of this potential is a challenging task due to the intra-user variability inherent in biometric measurements, which hinders reliable key extraction. This means that key generation biometric cryptosystems often exhibit low discriminability, which stems from the difficulty of generating a key that simultaneously possesses high stability and entropy. A highly stable key with zero entropy would mean that the same key is generated regardless of the input template, which would lead to a high FAR. Alternatively, a key possessing high entropy but no stability would result from a scheme that generates a different key for each variation of the same user's template, which would lead to a high FRR [2]. Furthermore, the only way in which keys generated from the same biometric template can be made more diverse and revocable is by either transforming the unprotected biometric template into a different version of itself prior to key generation, or else by modifying the extracted keys in some way after

they are generated. This sort of approach has been referred to as a *hybrid* protection scheme [2], which will be discussed in Section 3.4.

Linnartz and Tuyls [181] introduced the concept of *shielding functions*, in order to protect biometric data both during storage and during authentication. The role of a shielding function, G, is essentially to reliably generate a binary secret, S, from a quantised biometric template, X, using some *helper data*, W, extracted from the biometric template, such that $G(X, W) = S$. The overall goal of a shielding function is to leak a minimal amount of information about the biometric template, X. During enrolment, a user's biometric template, X, is extracted from their biometric features, then a secret, S, is randomly generated, and some helper data, W, is computed. The secret, S, is next hashed (using a one-way hash function, H), and $H(S)$, W, and some information identifying the user are stored in the database. During verification, a noisy version, X', of the user's biometric template, X, is acquired at the sensor. The user claims an identity and the corresponding helper data, W, is released and used to compute $S' = G(X', W)$ and subsequently $H(S')$. If $H(S')$ is exactly equivalent to $H(S)$ stored in the database, then the verification attempt is successful [182].

Dodis et al. [183], Smith [9], and Dodis et al. [184] generalised the shielding functions approach of Linnartz and Tuyls [181] by introducing two primitives for reliable key extraction directly from noisy biometric data: *secure sketch* and *fuzzy extractor*. The secure sketch is a concept that provides error tolerance to intra-user variation in biometric measurements. A secure sketch is essentially a protected biometric template generated from its unprotected counterpart. The secure sketch does not reveal any significant information about the unprotected biometric template, but it does publish some information about the template in the public domain in order to facilitate exact reconstruction of the original unprotected template from a query feature set that is close enough to the unprotected biometric template. An example of the publicly available data could be quantization boundaries (for example [185, 186]), which, when applied to the query biometric template, should produce a biometric template that is equivalent to the enrolled template, provided that the query and enrolled templates are close enough. The "key", in this case, is the quantised biometric template, and matching is usually done in the hashed domain of the key. While the publicly available data does reduce the entropy of the biometric template somewhat, a well-designed secure sketch will minimise this diminution in entropy such that the biometric template remains useful despite the publicly available information. The sketch can thus be viewed as helper data that can be made public while still protecting the biometric template.

A secure sketch does not, however, take into account the non-uniformity of biometric data. A fuzzy extractor, on the other hand, addresses both error tolerance and non-uniformity,

such that it is able to reliably extract a uniform string (key) from a biometric input in an error-tolerant manner. A minor change in the biometric input will still result in extraction of the same key. In order to facilitate reconstruction of the biometric key, the fuzzy extractor publishes some public information about the unprotected biometric template, much like a secure sketch. However, the extracted key remains uniformly random despite the revelation of the public data. Note that a fuzzy extractor can be generated from a secure sketch using strong randomness extractors. Figure 3.10 illustrates an example of the fuzzy extractor framework.

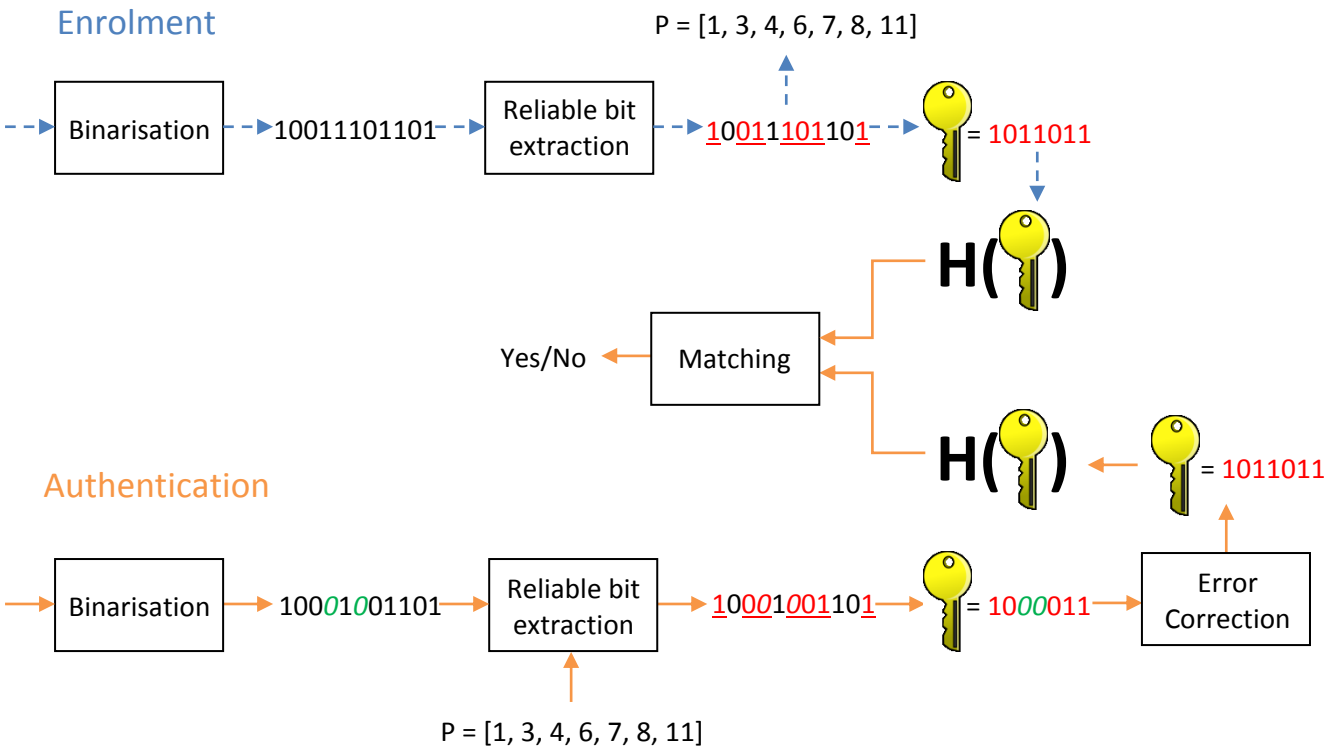


Figure 3.10: The enrolment (indicated by dashed blue arrows) and authentication (indicated by solid orange arrows) stages in an example fuzzy extractor framework.

During enrolment, a user's fingerprint image is captured and the salient minutiae points are extracted. The features are binarised, and then the reliable bits in the resulting binary feature vector are selected and extracted based on some criteria. The *positions* of the reliable bits in the binary feature vector are stored as helper data, P, and the reliable bits are concatenated to form a key (call the key R). The key, R, is hashed via a one-way cryptographic hash function, H, and $H(R)$ is stored in the database along with the helper data, P. During authentication, the user's salient minutiae (from their newly scanned fingerprint) are extracted and binarised, and the corresponding [to the user's claimed identity] helper data, P, and hashed key, $H(R)$, are retrieved from the database. The helper data, P, is used to indicate the positions of the reliable bits in the new binary feature vector. The reliable bits are

then extracted to form an erroneous key, *R''*, which is corrected using an error correcting code to produce *R'*. If the query fingerprint is very similar to the one used during enrolment, then *R''* will be able to be corrected such that the resulting *R'* is the same as *R*. If this is the case, then $H(R)$ and $H(R')$ should be equivalent, in which case authentication will be successful.

Dodis et al. [183], Smith [9], and Dodis et al. [184] went on to propose constructions of secure sketches and fuzzy extractors for three different distance metrics, namely, Hamming distance, set difference, and edit distance, where the distance metrics are used to evaluate the closeness between an enrolled biometric template and a query feature set. Hamming distance considers the number of bit positions that differ between the two templates. Set difference measures the closeness of two biometric templates that are represented as sets. Edit distance evaluates the similarity between two biometric templates depending on the number of insertions and deletions that would be needed in order to convert one template into the other. An advantage of the secure sketch and fuzzy extractor framework is that the whole biometric template need not be stored in the database; only a hashed version of the generated key and some helper data that does not reveal much information about the unprotected biometric data are stored. This is beneficial both in a security sense (the cryptographic hash function is a provably secure protection mechanism, and the helper data is minimally revealing), as well as being advantageous in terms of conserving storage space in large databases (a hash function produces a short summary of a large chunk of data, and the helper data need not be large).

Secure sketch and fuzzy extractor constructions have been proposed for several biometric modalities; for example, fingerprint templates [182, 187, 188], a multimodal biometric system consisting of fingerprint minutiae and facial features [189], 2D face [190], 3D face data [191], and acoustic ear identification [192].

Several weaknesses of secure sketch schemes have been analysed. Smith [9] showed that, due to the inherently noisy nature of biometric measurements, a secure sketch (or fuzzy extractor) generated for biometric data must always leak some information about the biometrics it aims to protect. Boyen [193] showed that a secure sketch scheme that is provably secure may become insecure when noisy versions of the same biometric template are used to generate multiple sketches. Simoens et al. [194] found that, if a sketch leaks a greater amount of information about the biometric template than is necessary to correct the errors arising from noisy biometric measurements, an adversary is easily able to identify protected templates originating from the same person. Furthermore, they demonstrated that an adversary with access to two or more sketches generated from the same person using different sketching functions can obtain more information about the original biometric template than they would have been able to with the availability of only a single sketch.

Since the secure sketch (and fuzzy extractor) framework is a general primitive, most biometric template protection schemes that fall into the key generation category are naturally based on the secure sketch (or fuzzy extractor) method.

3.4 HYBRID PROTECTION SCHEMES

Several biometric template protection techniques that make use of a combination of the basic approaches discussed above have also been presented. Such biometric template protection approaches have been referred to as *hybrid* schemes [2]. Hybrid protection schemes aim to merge the benefits of a number of different biometric template protection approaches, particularly focusing on coalescing feature transformations with biometric cryptosystems. Several hybrid protection examples are available in the literature, some of which even incorporate traditional cryptographic hashing functions into the hybrid protection scheme. Examples include: hardening a fingerprint-based fuzzy vault with a user-specific password (*key binding* combined with *salting*) [174]; an application-specific key release scheme that retrieves a cryptographic key bound to a BioHashed fingerprint (*salting* combined with *key binding*) [195]; applying a cryptographic one-way hash function to a discretized cancellable face template, where the discretization is applied in a similar way to the BioHashing approach (*non-invertible transform* combined with *salting*, which is finally hashed via traditional cryptographic means) [196]; the application of secure sketches to cancellable biometrics, using fingerprints as a case study (*non-invertible transform* combined with *key generation*) [197]; generating irrevocable cryptographic keys from cancellable fingerprint templates, where the key generation process is approached mathematically and not related to the secure sketch scheme (*non-invertible transform* combined with *key generation*) [198-200]; and applying a transformation to biometric data to split it into two parts, one of which is encrypted and used to secure the biometric template, and the other one of which is left unencrypted and used for robust distance matching (*non-invertible transform* combined with *key generation*) [68, 201]. The advantage of hybrid protection schemes is that they can combine the high revocability and diversity properties characteristic of feature transformation approaches with the high security offered by biometric cryptosystems [2].

3.5 SUMMARY

This chapter began by summarising the four agreed-upon characteristics of an ideal biometric template protection scheme: non-invertibility, cancellability, diversity, and performance. The

existing categories of biometric template protection schemes (i.e., *feature transformations* and *biometric cryptosystems*) were then explored.

The main strength of biometric template protection schemes that fall into the *feature transformations* category was attributed to their ability to produce cancellable fingerprint templates. Feature transformations can be further classified into *salting* and *non-invertible transforms*. The main advantage of the *salting* approach is its ability to achieve essentially perfect recognition accuracy due to the incorporation of external user-specific information into the fingerprint template. The main disadvantage is that the protected template is invertible with revelation of the user-specific transformation key (i.e., the *salt*). The chief advantage of the *non-invertible transforms* approach is that the protected template is secure even if the transform is known to an adversary. The chief disadvantage is that there is a trade-off between non-invertibility and recognition accuracy.

Their use of highly secure cryptographic frameworks was noted to be the main strength of biometric template protection schemes that fall into the *biometric cryptosystems* category. Biometric cryptosystems can be further classified as *key binding* and *key generation* schemes. The main advantage of the *key binding* approach is the ability to use error-correcting codes to deal with intra-class variance among multiple samples of the same fingerprint. The main disadvantage is that matching is done in the encrypted domain, so well-established traditional matching algorithms cannot be applied. The chief advantage of the *key generation* cryptosystem is that it can be used to simultaneously protect a fingerprint and a cryptographic key. The chief disadvantage is that it is difficult to reliably extract the same key from different versions of the same fingerprint.

The review concluded with a brief look at hybrid protection schemes, which aim to combine the high revocability and diversity properties characteristic of feature transformation approaches with the high security offered by biometric cryptosystems.

Overall, this chapter provides the reader with an appreciation of the various types of solutions that have been proposed to tackle the challenging task of securing fingerprint templates during storage in a database. Nevertheless, it must be noted that an agreed-upon solution does not yet exist, which means that the design of an ideal fingerprint template protection scheme remains an open problem.

Chapter 4

Non-invertible Fingerprint Transforms

Chapter 3 discussed the two main categories of fingerprint template protection schemes that have been proposed in the literature: *feature transformations* and *biometric cryptosystems*. This chapter provides a more detailed overview of the *non-invertible transforms* sub-category, which fits under the umbrella of the *feature transformations* category, of fingerprint template protection schemes in the literature.

4.1 INTRODUCTION

The promise of combining the benefits of the *feature transformation* and *biometric cryptosystems* categories fuelled an initial focus for this thesis on developing a new hybrid fingerprint template protection scheme; indeed, a significant amount of effort was invested in this direction. However, it was later realised that there was one particular similarity between essentially all the fingerprint template protection schemes currently in the literature, which, if challenged, could potentially present an entirely new solution to the problem of fingerprint template protection. It thus became evident that going down this path may result in a more significant contribution to this field of research. Consequently, the focus switched to proposing a fingerprint template protection scheme based on this new point of view. Although the resulting novel approach (which shall be presented in Chapter 5) is thus difficult to classify into any of the categories discussed in Chapter 3, the closest match is the *non-invertible transforms* sub-category within the umbrella of the *feature transformations* category. To help place our new fingerprint template protection scheme in the context of the relevant literature, the current chapter presents a more detailed literature review of *non-invertible transforms* as applied to the protection of fingerprint templates.

In Chapter 3, it was established that a fingerprint template protection scheme is shelved into the *non-invertible transforms* category if it secures a fingerprint template by transforming it using a non-invertible function. A function is considered to be *non-invertible* if it is easy to

perform the forward transform (i.e., converting a fingerprint template to a protected version of its former self), but it is computationally difficult to invert the function to perform the reverse transform (i.e., obtaining the original fingerprint template from its protected counterpart). The promising security benefits and the challenge of designing such a transform have motivated the focus of this thesis on developing a new non-invertible fingerprint template protection scheme.

Before launching into the proposal of our new non-invertible fingerprint template protection scheme, it is important to understand the nature of the approaches that already exist in this category. The focus of this chapter, therefore, is on presenting a thorough discussion of the types of non-invertible fingerprint template protection schemes that have been proposed in the literature thus far, followed by a consideration of the techniques employed in evaluating the robustness of the proposed schemes. Upon reading this chapter, the reader will be better equipped to understand the context of our new fingerprint template protection scheme, which is proposed in Chapter 5, and the methods adopted for its evaluation in the subsequent chapters of this thesis.

The remainder of this chapter begins by considering what fingerprint information is most commonly used in the development of a non-invertible fingerprint template protection scheme, and what type of intra-class variance is the main focus and how it is dealt with. The nature of the non-invertible fingerprint transforms currently existing in the literature is then investigated. Finally, a discussion of the most common techniques used to evaluate the robustness of non-invertible fingerprint template protection schemes is provided.

4.2 FINGERPRINT INFORMATION USED

In Section 3.2.2, it was mentioned that a non-invertible transform can either be applied to the biometric signal in the signal domain (whereby the signal is distorted directly upon acquisition) or in the feature domain (where the transform is applied to the biometric signal's extracted features). In the context of the fingerprint biometric, the signal domain would refer to a digital fingerprint image, and the feature domain would refer to a set of features extracted from the fingerprint image (e.g., *minutiae* – see Section 1.1). An extensive perusal of non-invertible fingerprint template protection schemes in the literature has revealed that fingerprint *features* are preferred over the entire fingerprint image and that a fingerprint's *minutiae* are by far the most popular features used (e.g., [20, 65, 69, 108, 109, 111, 112, 132-141, 143-147, 151, 202-206]). This may be attributed to the fact that, as mentioned in Chapter 1, a fingerprint is commonly stored in a database in the form of a minutiae template;

therefore, effective strategies for securing those minutiae are urgently required. For this reason, the focus of this thesis shall be on developing a novel minutiae-based non-invertible fingerprint template protection scheme. Likewise, the remainder of this chapter shall discuss minutiae-based non-invertible fingerprint template protection strategies only.

4.3 DEALING WITH INTRA-CLASS VARIANCE

In Section 2.1, it was mentioned that multiple samples of the same fingerprint acquired at different times will always exhibit some amount of intra-class variance. This was identified as the main challenge in designing effective fingerprint template protection schemes. Considering minutiae-based non-invertible fingerprint template protection schemes in the literature, it is evident that the main focus, in terms of intra-class variance, is on dealing with misalignment between a reference and a query minutiae template. This has generally been approached in one of two ways: expressing the (x, y, θ) attributes of each minutia within a template relative to a common global reference point, or else constructing self-aligned local minutiae structures. The former method of dealing with misalignment is more traditional, and the reference point is usually either the core point (e.g., see [65, 108, 132, 136, 144-146]) or else each minutia in the template has a turn at being the reference relative to which the remaining minutiae are expressed (e.g., see [3, 133, 134, 138-141]). The latter method for dealing with misalignment involves constructing a self-aligned local minutiae structure for each minutia in the template, where the structure represents the relationship between a central (or reference) minutia and a small number of neighbouring minutiae (e.g., see [111, 135, 137, 150, 151]).

Using a single reference point for alignment tends to be less memory-intensive than the method based on local minutiae structures (particularly when the core point is used), both in terms of the processing time and the amount of space required to store the final minutiae template. On the other hand, employing local minutiae structures for alignment may generally be expected to be more reliable than using a single reference point, since the former method is less sensitive to missing or inaccurately detected features.

Our new fingerprint template protection scheme, which will be proposed in Chapter 5, uses the core point for alignment. The modified version, proposed in Chapter 11, represents a self-aligned minutiae structure; however, the structure is not *local* in the same sense as the structures used in [111, 135, 137, 150, 151], since it does not necessarily consist of *neighbouring* minutiae, and we use only *one* structure in total as opposed to constructing one structure for *every minutia*.

4.4 TYPES OF EXISTING NON-INVERTIBLE FINGERPRINT TRANSFORMS

This section investigates the types of *non-invertible transforms* that have been used in the literature to convert a fingerprint's minutiae template to a secured version of its former self. There are two main categories of approaches. The techniques in the first category involve perturbing the minutiae locations and/or orientations by some random amounts, such that the protected minutiae template consists of modified locations and/or orientations pertaining to the original minutiae. Non-invertibility is generally achieved by mapping several minutiae to the same location (many-to-one mapping) and/or by superimposing several mappings of each minutia into the same space, which makes it impossible to deduce the minutiae's original positions. Cancellability is usually accomplished by changing the amount of random perturbation. Examples of methods in this category include [3, 20, 65, 69, 108, 109, 111, 112, 132-137].

The techniques in the second category focus on transforming the original minutiae template into one or more fixed-length histograms, which essentially denote the number of minutiae present in certain parts of a fingerprint as opposed to storing the actual minutiae location and orientation attributes. Non-invertibility is generally a natural consequence of the information lost in quantising the minutiae for the purpose of binning them into a histogram, and often the histogram bins are binarised to hide the true number of minutiae in each bin. Cancellability is usually accomplished by permuting the resulting histogram bins using an external permutation key. Examples of methods in the literature that focus on this type of approach include [138-150].

The first category of approaches shall henceforth be referred to as *perturbation-based* non-invertible transforms, since the essence of the transform lies in perturbing the minutiae locations and/or orientations. The second category of approaches shall henceforth be referred to as *histogram-based* non-invertible transforms, since the crux of the transform is the quantisation of minutiae into histogram bins. Sections 4.4.1 and 4.4.2, respectively, present examples of *perturbation-based* and *histogram-based* non-invertible transforms in the literature.

A third category of non-invertible transform approaches involves representing the minutiae template in terms of local minutiae structures, from which the original minutiae template is difficult to reconstruct. This type of approach has not yet had a significant amount of focus in the literature, so Section 4.4.3 provides the only example of a technique in this category that we are aware of: [151].

Some general conclusions about the nature of non-invertible fingerprint template protection schemes existing in the literature are drawn in Section 4.4.4.

4.4.1 Perturbation-based Non-invertible Transforms

The first mention of a *perturbation-based* non-invertible transform appears in [20, 69], where the notion of *cancellable biometrics* was introduced. In particular, the authors suggest randomly perturbing the minutiae point set such that several of the points are mapped to the same location, or mapping the minutiae coordinates onto a high-order polynomial. In both cases, the fact that the forward mapping is many-to-one ensures that the reverse mapping is one-to-many, thereby preventing recovery of the original minutiae template. Furthermore, since the possibility of using a different mapping would allow for the generation of a different transformed minutiae set in the event that the first one is compromised, these schemes are both considered *cancellable*.

The most well-known *perturbation-based* non-invertible transforms are the Cartesian, polar, and functional transforms proposed in [65, 132] for fingerprint minutiae. Since the essence of these transforms was described in detail in Section 3.2.2, we refrain from repeating the details here. However, we note that several researchers (e.g., [3, 133, 134]) have recently suggested interesting modifications to these transforms since their proposal in [65, 132].

Another commonly-cited *perturbation-based* non-invertible transform technique, which is amongst the first fingerprint template protection schemes proposed in the literature, appears in [108]. This method involves the use of a user-specific key, which specifies the angle of a line to be drawn through the fingerprint's core point. All of the minutiae lying below this line are then reflected across the line, while those minutiae already located above the line are left unchanged. Non-invertibility is a natural consequence of the fact that the minutiae are now 'mixed up', so knowledge of the line's angle does not reveal the original minutiae positions. However, the authors note that, since the minutiae above the line are left unperturbed, the resulting protected template leaks some information about the original minutiae template. Cancellability is achievable by changing the user-specific key that specifies the angle of the reflection line.

A different approach was proposed in [109], where, instead of transforming a *set* of minutiae (as in [108]), each minutia is perturbed on its own. The transformation begins by generating a translation- and rotation-invariant value, m, for each minutia. To compute m, k concentric circles are drawn around each minutia, and each circle is uniformly sampled. The orientation of the fingerprint ridge line at each sample point along every circle (in the anticlockwise direction) is calculated relative to the orientation of the central minutia. The

resulting orientation differences form a feature vector, F, which is subsequently normalised by its norm. A user-specific PIN is used as a seed to a random number generator to produce a random number vector, whose dimension is equivalent to the dimension of F. The resulting random number vector is also normalised by its norm, and the dot product between this vector and the normalised F is used as the invariant m for this particular minutia. Next, m becomes the input to two "changing functions": a distance changing function and an orientation changing function. The outputs of these functions indicate the amount by which the corresponding minutia should be translated and rotated. The set of all translated and rotated minutiae constitutes the protected fingerprint template. The non-invertibility of this method is essentially attributed to an attacker's inability to figure out a minutia's invariant value, m, even if the changing functions are known, since m comes from the fingerprint image rather than from the minutiae template. If the range of an invariant value is known, however, then the non-invertibility of the scheme relies on the size of this range and thus the difficulty of guessing the original minutia within that range; in other words, the non-invertibility may be likened to the difficulty of guessing the original minutiae attributes from their quantised counterparts. The cancellability of this scheme is achievable by altering the changing functions.

The *perturbation-based* non-invertible transforms proposed in [108, 109] involve at most one transform per minutia, such that the resulting protected template is no bigger than the original unprotected template. A more popular approach towards designing a *perturbation-based* non-invertible transform is to perturb each minutia several times and then superimpose every transformed version of each minutia into a protected minutiae template that is much larger than the original. For example:

- [111] begins by constructing a 'vicinity' for each minutia, where a vicinity consists of a minutia and its M nearest neighbouring minutiae. Within each vicinity, L minutiae pairs are chosen. The line connecting each of these L minutiae pairs becomes the new x-axis in turn, and the midpoint of each line becomes the new origin. The remaining minutiae within the same vicinity are aligned with respect to each new L and then translated in the x and y directions by randomly generated offsets. After all the minutiae for each L have been offset in this manner, the transformed minutiae are superimposed to generate the protected vicinity. This process is repeated for each minutia's vicinity, and the final protected minutiae template consists of the set of all protected vicinities. Note that a modification of this scheme is proposed in [135], where each protected vicinity is quantised and binarised following the superimposition process.

- [112] uses each minutia in turn as the origin, and the line between the minutia and the core point forms the new x-axis relative to which the remaining minutiae in the template are aligned. The minutiae are then translated along the x and y axes by randomly generated offsets. After this perturbation, only minutiae that fall inside a local disc of radius R around the central minutia are kept. The final protected minutiae template consists of a superimposition of all these local discs.
- In [136], a pseudo-random number generator is first used to generate the (x, y) locations and orientations, θ, of N random reference points. The location of each reference point corresponds to the new origin and its orientation represents the new direction of the x-axis. For each of the N reference points in turn, all the minutiae in the template are translated and rotated so that they are expressed with respect to that particular reference point. A rectangular area is then cropped around the centre of each set of transformed minutiae. Finally, all of the minutiae in the cropped areas are superimposed into one area to form the protected minutiae template.

Note that, in [111, 112, 136], the non-invertibility mainly stems from superimposition of the transformed minutiae, which makes it difficult to establish the original minutiae positions and/or orientations even if the random perturbations are known. Cancellability is achieved by generating different random offsets.

Similar to the approaches presented in [111, 112, 136], but differing in the absence of the superimposition step, is the method proposed in [137]. This method begins by constructing a circular region of radius R around each minutia. In every region, the x and y coordinates of each neighbouring minutia are then expressed in terms of polar coordinates relative to the central minutia, and the difference between the orientations, θ, of the central minutia and each neighbour is also computed. The resulting 3-tuple feature set is quantised, and the set of all quantised 3-tuple feature sets is referred to as a *MinuCode* for that particular region. Each feature vector in a MinuCode is then transformed separately by performing a set of additions and multiplications with other features in the same feature vector, randomly-generated integers, and an application-specific parameter. The set of all transformed feature vectors results in a protected MinuCode, and the set of all protected MinuCodes constitutes the final protected minutiae template. Since the quantised 3-tuple feature sets in a MinuCode can be determined if we assume that the attacker has access to all the necessary external information (such as the application-specific parameter), the non-invertibility of this scheme is mainly the result of the quantisation of the minutiae attributes. Cancellability is achievable by changing the application-specific parameter used to generate each MinuCode.

All of the *perturbation-based* non-invertible transforms discussed in this section are certainly interesting and creative. Their most important limitation, however, is that the level of non-invertibility is generally quite limited under the assumption that an attacker has access to the transform and any associated external parameters. Furthermore, the methods that process each minutia in turn are quite computationally intensive, and the approaches involving superimposition of perturbed minutiae often result in a protected template that is much larger than the original, which necessitates a larger amount of storage space in the database. Finally, changing the locations and orientations of minutiae may be dangerous, because the transformation could accidentally produce a template that is similar to that of an impostor.

4.4.2 Histogram-based Non-invertible Transforms

A *histogram-based* non-invertible transform involves representing the minutiae template in terms of one or more histograms. A histogram essentially denotes the *number* of minutiae in certain parts of the fingerprint, rather than the actual x, y, θ attributes of the individual minutiae (as in the *perturbation-based* non-invertible transforms discussed in Section 4.4.1). Section 4.4.2.1 provides several examples of *histogram-based* non-invertible transforms that focus on generating a separate histogram for each minutia in the template. Section 4.4.2.2 presents examples of *histogram-based* non-invertible transforms whose goal is to produce a single histogram to describe the entire minutiae template in a secure way.

4.4.2.1 Approaches that Produce a Separate Histogram for Each Minutia

This section provides examples of *histogram-based* non-invertible transforms that produce protected minutiae templates consisting of a separate histogram generated for each minutia in the original template. The first class of approaches in this category generate a minutia's histogram by counting the number of minutiae that fall within a certain region of that minutia in a 2D space and then quantizing the results to form a fixed-length 1D histogram. For example:

- In [138], a minutia is selected as the reference minutia, and all the other minutiae in the original template are translated and rotated so that their location coordinates and orientations are expressed relative to the reference minutia. The aligned minutiae lie in a 2D area that is twice the height and width of the original fingerprint image. A user-specific key specifies the vertices of a number of triangles. These triangles are overlaid onto the aligned minutiae template, and the number of minutiae inside each triangle is counted. Furthermore, for each triangle, the number of minutiae in each of six possible

angle ranges is recorded; so, a 6-dimensional feature vector is used to represent each triangle. The feature vectors of all the triangles are then concatenated to form the final "hash" vector. Each decimal value in the hash vector is next binarised using a scheme termed "Bit-Block Coding". This procedure is repeated using each minutia as the reference minutia in turn, and the final protected minutiae template consists of all the resulting hash vectors.

- In [139, 140], once again each minutia has a turn at being the reference minutia, and all the other minutiae are expressed relative to the reference minutia. After this alignment, all minutiae are shifted such that they lie in a new coordinate system that is twice the width and height of the original fingerprint image. The new coordinate system is then converted into a polar grid centred at the reference minutia, such that each minutia is now represented in terms of polar coordinates with its orientation relative to the reference minutia. The resulting attributes of each minutia are next quantised. Finally, a histogram is created, which keeps a count of how many minutiae are located inside each polar sector. The histogram is then binarised, such that 0 is allocated to a bin that contains no minutiae and 1 is allocated to a bin that contains one or more minutiae, thereby achieving a many-to-one mapping. The final string is permuted by a user-specific key. This process is repeated using each minutia as the reference minutia in turn, and the set of all the resulting permuted bit-strings constitutes the protected fingerprint template.

Note that, under the assumption that an attacker has access to the transform and any associated external parameters, the non-invertibility of the schemes in [138-140] is mainly the result of hiding the minutiae locations and orientations by recording only the *number* of minutiae present in a particular area, as well as the additional information loss incurred in the histogram binarisation. Cancellability is achievable by changing the external user-specific keys.

The second class of approaches in this category is similar to the first class; however, this time, the minutiae are mapped to a 3D space instead of a 2D space. For example:

- In [141], once again each minutia has a turn at being the reference minutia, relative to which all the other minutiae are aligned. After alignment, a 3D grid (the width and height of which are twice that of the fingerprint image, and whose depth is 2π) is generated and the aligned minutiae are placed into the cells depending on their new (x, y) coordinates and angles. The number of minutiae in each cell is counted: if the count is 0, the value stays 0, and if the count is 1 or more, then the value of that cell is set to 1. A binary string is then generated by sequentially visiting all the cells. The values of the binary string are

next permuted, where the permutation is based on a user-specific PIN and the type[4] of the reference minutia. Then, the addresses corresponding to the positions of the bit-string that have a value of 1 are hashed using a cryptographic hash function. This process is repeated using each minutia as the reference minutia in turn, and the final protected minutiae template consists of the set of permuted hashed binary strings. The authors attribute the non-invertibility of this scheme mainly to the use of the cryptographic hash function, because they claim that, without it, an attacker can obtain the quantised minutiae positions and orientations from the protected template. The cancellability of this scheme is achievable by changing the user-specific permutation PIN.

- The approach in [150] involves the construction of a 3D cylinder centred at each minutia in turn. The number of neighbouring minutiae that fall within each cell of the 3D cylinder are counted, and the resulting fixed-length vector is referred to as a Minutia Cylinder-Code (MCC). Each MCC is then subjected to a dimensionality reduction and quantisation procedure, upon which the entire set of protected MCCs constitutes the final protected minutiae template. The non-invertibility of this scheme is mostly the result of the dimensionality reduction and quantisation, since the authors demonstrate that the lack of these additional processes (such as in the original MCC method proposed in [207]) would enable an attacker to recover most of the original minutiae from the set of MCCs. The authors state that, in its current form, the proposed fingerprint template protection scheme is not cancellable, but they plan to add this property in the future with the help of a user-specific secret key.
- The approach in [142] involves passing a line[5] through each minutia in turn, where the orientation of the line is equal to the orientation of the minutia. A number of equally-spaced samples along the line are taken and a circular area centred at each sample point is considered. For every circle, the number of minutiae falling inside each angle sector[6] is counted to generate a line code, which is then permuted by an external key. The authors attribute the non-invertibility of this scheme to the many-to-one mapping resulting from storing a minutia *count* instead of the actual minutiae locations and orientations. The

[4] Recall, from Section 1.1, that the two minutiae types most commonly adopted in fingerprint recognition are the *bifurcation* and the *termination*. These are the two types considered here also.

[5] Note that the authors subsequently decided to use 3 lines through each minutia to improve performance.

[6] The angle in this case is the difference in orientation between the reference minutia and each neighbouring minutia. The angle sectors are considered to constitute rows of a cylinder, which is why the resulting representation is referred to as being 3D.

cancellability and diversity of this scheme are achievable by changing the user-specific external permutation key.

4.4.2.2 Approaches that Produce a Single Histogram for the Entire Minutiae Template

This section provides examples of *histogram-based* non-invertible transforms that produce a protected minutiae template consisting of a single histogram.

One of the earliest approaches in this category was proposed in [143]. The first step in this method involves establishing the centroid of the (x, y) coordinates pertaining to the entire set of minutiae in the original minutiae template. A circle of radius R is centred on this centroid and, henceforth, only minutiae inside this circle are considered. For every pair of minutiae, if the distance between the two minutiae is greater than a pre-determined threshold, then a line is drawn through both minutiae points until it intersects two points on the circle[7]. Once every minutiae pair has thus been considered, the circle is divided into equally-sized bins and the number of intersection points falling inside each bin is counted. Finally, the resulting bins are concatenated to form a fixed-length feature vector, which represents the protected minutiae template. A modification to this approach was proposed in [144]. Instead of extending the line between each minutiae pair until it intersects the circle at two points, as in [143], the modified approach in [144] involves a perpendicular projection of the line onto the circle at the two points specified by the locations of the two minutiae. Furthermore, additional local features (such as the relative angles between the minutiae pairs) and global features (such as ridge frequency) are fused with the quantised intersection points on the circle to generate the final fixed-length feature vector. The non-invertibility of the schemes proposed in [143, 144] stems from the difficulty of establishing the original minutiae positions from the intersection points on the circle, as well as the additional information loss caused by quantisation of the intersection points into bins, which helps to conceal the exact intersection points on the circle. While cancellability and diversity do not appear to be features of the fingerprint template protection scheme in [143], the modification in [144] incorporates these properties into the protected fingerprint template by using randomly generated parameters in the generation of the fixed-length feature vector.

Another example of a projection-based technique in this category can be found in [145]. In this approach, a line is drawn through the core point at an angle specified by a user-specific key. The minutiae are then projected onto the line in one of two ways. The first way involves two projections per minutia: one projection is parallel to the x-axis, and the other projection is

[7] Note that the two points would be located on opposite sides of the circle.

parallel to the y-axis. The second way involves three projections per minutia: the two aforementioned projections plus another projection parallel to θ. The line is then divided into a number of segments, the segments are indexed by a user-specific key, and the number of projected minutiae in each segment is counted. The resulting fixed-length feature vector constitutes the protected minutiae template. Note that a modification to this scheme is proposed in [146]. In the modified approach, prior to the projection of the minutiae onto the line, the fingerprint image is divided into a number of cells and all the minutiae in a single cell are mapped to the centre of that cell, thereby achieving a many-to-one mapping. The non-invertibility of the schemes proposed in [145, 146] mainly stems from the information lost in projecting the minutiae onto a line, which makes it difficult to determine the original minutiae attributes. Note that the many-to-one mapping of minutiae into cell centres prior to the line projection in [146] adds another layer of non-invertibility. The cancellability of the schemes in [145, 146] are achievable by changing the user-specific permutation key.

Another example of a non-invertible transform that produces a protected minutiae template in the form of a single histogram can be found in [147, 148]. This approach begins by representing each minutiae pair in terms of the distance between their locations, the difference between their orientations, and the difference between the orientation of each minutia and the angle of the line that joins them. Then, each of those attributes is quantised and binarised. The binarised values are concatenated and the decimal equivalent of the concatenated binary string is computed. The number of minutiae pairs that produce that same decimal value is counted: a count of 1 is left unchanged, while counts of 0 or greater than 1 are set to 0. The final binary string is permuted using a user-specific key, and the permuted fixed-length feature vector constitutes the protected minutiae template. Assuming that an attacker has access to the external permutation key, the non-invertibility of this scheme is mainly attributable to the information loss incurred in producing the bit-string. Cancellability is achievable by changing the user-specific permutation key. Note that a modified version of this scheme is proposed in [149], where the fixed-length feature vector resulting from the method in [147, 148] is subsequently convolved with a shorter random finite-duration sequence. The idea behind this "curtailed convolution" is to produce a fixed-length feature vector that is more difficult to invert to obtain the original minutiae template.

The *histogram-based* non-invertible transforms proposed in the literature represent an interesting contrast to the *perturbation-based* non-invertible transforms: the general aim of the latter approach is to achieve non-invertibility by *changing* minutiae locations and/or orientations, while the general premise of the former approach is to *quantise* the minutiae locations and/or orientations in some manner in order to make it difficult to figure out their

original values. In both cases, however, the non-invertibility of the protected fingerprint template becomes limited under the assumption that an attacker has access to the transform and any associated external parameters. Indeed, the chief limitation of most of the *histogram-based* non-invertible transforms in the literature is that, under this assumption, the non-invertibility of the protected fingerprint template relies on the difficulty of establishing the original minutiae distribution from the quantised version of that information in the histograms. While it may be difficult to precisely recover the original minutiae attributes, it seems that an approximation would be feasible. Furthermore, while coarser quantisation would improve the non-invertibility of a *histogram-based* non-invertible transform, excessive histogram quantisation is likely to reduce the discrimination capabilities of the resulting protected template. Finally, many of the *histogram-based* non-invertible transforms discussed in this section are quite computationally expensive, particularly those that involve generating a separate histogram for each minutia in turn.

4.4.3 Non-invertible Transforms based on Local Minutiae Structures

A non-invertible transform based on local minutiae structures essentially involves the representation of a minutiae template in terms of local minutiae structures alone. Note that many of the approaches discussed in the *perturbation-based* and *histogram-based* non-invertible transforms in Sections 4.4.1 and 4.4.2, respectively, also employ local minutiae structures; the difference, however, lies in the fact that those approaches do not rely on local minutiae structures alone to secure the underlying minutiae template (e.g., the local structures are often perturbed, quantised into histograms, etc.). Conversely, non-invertible transforms based on local structures alone, which are discussed in the current section, rely only on the natural non-invertibility provided by these local structures. The general premise is that, since the structures are local, they do not give away the global positioning of the minutiae with respect to the entire fingerprint, thereby effectively preventing the reconstruction of the corresponding minutiae template.

To the best of our knowledge, the only example of a non-invertible transform in this category was proposed in [151]. This method begins with the selection of three neighbouring minutiae, which form the three vertices of a triangle. The circumcentre of the resulting triangle is then calculated, and the orientation of each of the three minutiae is expressed with respect to the line connecting the particular minutia to the circumcentre. Next, the two largest angles between the lines connecting the three minutiae to the circumcentre are determined. Finally, using majority voting, the overall minutia type (i.e., *bifurcation* or *termination*) is established. The (x, y) coordinates of the circumcentre, the five angles, and the decided

minutia type all form the feature vector for this particular set of three minutiae. This process is repeated for every minutia in the original minutiae template, and the resulting protected minutiae template consists of a set of feature vectors describing these local minutiae structures. The non-invertibility of this scheme is attributable to the fact that the local minutiae structures do not give away the global positioning of the minutiae in the fingerprint image, so it is difficult to reconstruct the original minutiae template. The authors seem to imply that cancellability can be achieved by finding different ways to generate each minutiae triplet; however, this is not particularly clear in the associated paper.

While the premise that local minutiae structures do not reveal the original minutiae template seems reasonable, we cannot ignore the fact that each local structure reveals a small part of the fingerprint. Consequently, a protected fingerprint template consisting of several local minutiae structures may be used to recover important information pertaining to the original minutiae template.

4.4.4 Conclusions on the Types of Existing Non-invertible Fingerprint Template Protection Schemes

The discussion in Sections 4.4.1 to 4.4.3 demonstrates the existence of a wealth of interesting non-invertible fingerprint template protection schemes in the literature. The one aspect that all these schemes have in common, however, is that they employ the *entire* minutiae template in the generation of the protected template. While the idea behind this approach would be to incorporate as much discriminatory information as possible into the resulting protected template, the downside is that this strategy effectively *puts all one's eggs in one basket*. In other words, if a single protected template is compromised (e.g., stolen from the database), then the difficulty of recovering the original minutiae template depends only on the level of non-invertibility provided by the employed transform. Since it was noted that the non-invertibility of most of the non-invertible fingerprint template protection schemes in the literature becomes limited in the event that the transform and any associated external parameters are known to an attacker, it seems very risky to place the security of an entire fingerprint template on the design of a single non-invertible transform. This point is particularly important due to the fact that it is extremely difficult to design a good non-invertible transform.

In light of this observation, an alternative approach to securing a fingerprint template could be to achieve non-invertibility by using only a *small portion* of the entire minutiae template to generate the protected template. In this way, the non-invertibility of the protected template relies on the simple fact that most of the information required to reconstruct the

original minutiae template is literally missing from the protected version. This seems to be a safer way of achieving non-invertibility, since it removes the danger of recovering the original template as a result of a poorly-designed non-invertible transform applied to an entire minutiae template. This concept was applied in the development of our new non-invertible fingerprint template protection scheme, which is proposed in Chapter 5 and is the main contribution of this thesis.

4.5 TECHNIQUES USED TO EVALUATE NON-INVERTIBLE FINGERPRINT TEMPLATE PROTECTION SCHEMES

Section 3.1 presented four requirements for an ideal biometric template protection scheme, which have generally been agreed upon in the associated literature. In particular, it was stated that an ideal biometric template protection scheme should ensure that the protected biometric template is *non-invertible* (such that the original template cannot be obtained from the protected template), *cancellable* (such that a compromised protected template can be replaced by a different protected template generated from the same original template), *diverse* (such that different protected templates, generated from the same original template, can be used to enrol across different applications without the risk of the user being tracked), and that the *performance* (i.e., recognition accuracy) provided by the protected template is comparable to the performance resulting from using the unprotected template. Sections 4.5.1 to 4.5.4, respectively, consider the techniques used to evaluate *non-invertibility, cancellability, diversity* and *performance* of the non-invertible fingerprint template protection schemes currently in the literature.

4.5.1 How is Non-invertibility Measured?

In mathematical terms, a function is either *invertible* or *non-invertible*. An *invertible* function (or a *bijection*) maps every input to a *different* output, i.e., the mapping is one-to-one. Consequently, if we know the outputs and we have access to the function used to generate them, we can easily determine the corresponding inputs. On the other hand, a *non-invertible* function (or a *surjection*) maps several inputs to the *same* output, i.e., the mapping is many-to-one. In this case, even if we know the outputs and the function used to generate them, we cannot determine the *unique* set of inputs since the reverse mapping would be one-to-many.

In the arena of fingerprint template protection, we talk about the *non-invertibility* of a transform used to generate a protected template from its unprotected counterpart. In the literature, this term is used rather loosely to refer to two separate concepts: the first is a proof

that the transform is, in fact, *non-invertible*, and the second is an estimation of *how* non-invertible the transform is. Referring to the definition of a *non-invertible* function above, we may deduce that, the more inputs that are mapped to the same output, the more difficult it will be to infer the original set of inputs from the outputs; consequently, the difficulty of obtaining the inputs from the outputs may be used to assess *how* non-invertible the function is. Quantifying the *non-invertibility* allows us to measure the effectiveness of a *non-invertible* fingerprint template protection scheme by evaluating the difficulty of reconstructing the original fingerprint template from the protected version.

It turns out that measuring the *non-invertibility* of a fingerprint template protection scheme is a non-trivial task, and this, combined with the relative infancy of the fingerprint template protection research field, means that standard techniques for evaluating this *non-invertibility* have not yet been established. For this reason, the *non-invertibility* of fingerprint template protection schemes in the literature is generally either not formally assessed or else is evaluated in whichever way seems most fitting to a particular method. In this section, we investigate what techniques researchers in this field have adopted for measuring the *non-invertibility* of their proposed non-invertible fingerprint template protection schemes.

Upon perusing the literature concerning minutiae-based non-invertible fingerprint transforms, it appears that researchers in this field generally take one of two approaches for convincing the reader of the non-invertibility of their proposed scheme. The first approach involves providing a proof that the proposed method is *non-invertible* by showing that the forward mapping is many-to-one and thus that the reverse mapping is one-to-many. Although this is loosely referred to as a *non-invertibility* analysis, no estimation of the *non-invertibility* is actually provided (for example, see [3, 20, 65, 69, 109, 112, 138, 139, 141, 142, 144, 151]). The second approach involves quantifying the *non-invertibility* by estimating the number of guesses required to recover the original minutiae template from the protected minutiae template via brute force (for example, see [111, 132, 134-137, 143, 145-147, 149]). The general idea is to estimate the number of guesses required to figure out the m input minutiae that were used to generate the n output minutiae[8]. However, since the exact method of calculating the brute-force complexity depends on the nature of the fingerprint template protection scheme (e.g., parameters employed, number of minutiae that need to be recovered, accuracy of the recovered minutiae, etc.) and/or the size of the fingerprint images used, it is

[8] Note that, formally, a *non-invertible* function produces fewer outputs than inputs, such that $n < m$. However, in some cases (e.g., recall the discussion on the superimposition of transformed minutiae in Section 4.4.1), a fingerprint template protection scheme might result in a protected template that is larger than the unprotected template. Therefore, the exact relationship between n and m will depend on the nature of the non-invertible transform applied.

not computed in a consistent manner across all fingerprint template protection schemes in the literature. The inconsistencies in evaluating the brute-force attack complexity mean that it is generally difficult to quantitatively compare the *non-invertibility* of different fingerprint template protection schemes, so this is often avoided in the literature.

A third approach for evaluating the non-invertibility of a non-invertible fingerprint transform involves computing the percentage of the original minutiae template that remains unrecoverable in the event that the protected template is compromised. While such an evaluation is, in our opinion, more intuitive than a brute-force complexity analysis, to the best of our knowledge there are only two methods for which such an evaluation is conducted: [132, 150].

To gain as much insight as possible into the non-invertibility of our new fingerprint template protection scheme, which is proposed in Chapter 5, we adopted all three evaluation techniques discussed above (for details, see Chapter 10). More specifically, we first consider the non-invertibility of our fingerprint template protection scheme from an intuitive perspective. Non-invertibility is then quantified in terms of the proportion of minutiae that remains unrevealed in the event of compromise, as well as in terms of the brute-force attack complexity.

Regardless of which of the above methods is adopted in evaluating the *non-invertibility* of a fingerprint template protection scheme, it is important to note that the *non-invertibility* must be evaluated **under the assumption that an attacker has access to the transform as well as any external parameters used in the transform.** So, for example, a technique that simply permutes a minutiae template using an external permutation key must be considered *invertible*, since it cannot be assumed that the attacker will not have access to that key. This is because the most important feature of a *non-invertible transformation* is that the resulting template is non-invertible even when both the transform and its parameters are known. Most non-invertible fingerprint template protection schemes that have been published in the literature have been analysed with this point in mind. Often, however, a complementary analysis on the improved non-invertibility as a result of keeping the transform and its parameters secret has also been included, followed by a recommendation that the transform and its parameters should be kept secret in practice in order to increase the difficulty of reconstructing the original minutiae template from its protected counterpart.

Note that the *non-invertibility* analysis for a non-invertible fingerprint template protection scheme in the literature generally focuses on the difficulty of recovering the original minutiae template from a *single* protected template. While this type of analysis is certainly important, [173] pointed out that it may be possible to correlate multiple protected templates originating

from the same unprotected template to recover the unprotected template in what was termed a *Record Multiplicity Attack* (ARM). In essence, [173] effectively suggested that the concept of applying different transforms to the *same* biometric template is a misuse of the "one-time pad" model in classical cryptography. The one-time pad model essentially guarantees 'perfect security' under the strict requirement that the "pad" is used only once; for example, the same message is never encrypted using different encryption keys. The fingerprint template protection research community started taking notice of this potential problem when it was shown in [173] and later empirically justified in [175] that one of the most popular fingerprint template protection schemes in the literature, the Fuzzy Vault scheme, is vulnerable to this type of attack. For example, [175] showed that the correlation of only *two* fuzzy vaults generated from the same fingerprint may be sufficient to reveal the original minutiae template. This is because two different fuzzy vaults from the same fingerprint are generally constructed using the *same* set of minutiae but different chaff points. Correlating two or more fuzzy vaults from the same fingerprint thus makes it possible to identify the true minutiae points from the randomly generated chaff points. Indeed, since current fingerprint template protection schemes generally construct a protected fingerprint template by modifying the *entire* original fingerprint template in some manner, it makes sense to expect a possible correlation between multiple protected templates originating from the same fingerprint.

In the literature pertaining to the *non-invertible transforms* approach to fingerprint template protection, which is the focus of the current chapter, a dedicated analysis of each scheme's susceptibility to a Record Multiplicity Attack is currently lacking. While it is common to evaluate the correlation between multiple protected templates originating from the same fingerprint in terms of the probability of them matching[9], this type of analysis is insufficient to reveal a correlation that may be present at a deeper level, particularly if the matching does not consider the case where an external factor used in the generation of the protected template (e.g., a permutation key) is known to the attacker. If conducted properly, we believe that this type of analysis would be likely to reveal that essentially *every* currently-existing non-invertible fingerprint template protection scheme is susceptible to a Record Multiplicity Attack. This is because, as pointed out in [173], a collection of protected templates generated by applying different transforms to the *same* biometric template cannot be expected to *not* exhibit correlations capable of revealing the original template. Indeed, a thorough analysis on a set of four non-invertible fingerprint template protection schemes in

[9] See the discussion on evaluating cancellability in Section 4.5.2.

the literature (i.e., those proposed in [109, 134, 144, 208]) has recently been conducted in [209], where it was shown that all four schemes are susceptible to a Record Multiplicity Attack. More of this type of work is needed in the literature.

The susceptibility of our new non-invertible fingerprint template protection scheme, which is proposed in Chapter 5, to a Record Multiplicity Attack is thoroughly investigated in Chapter 11.

4.5.2 How is Cancellability Measured?

A fingerprint template protection scheme is considered to be *cancellable* if it is possible to generate multiple different protected templates from the same unprotected template. The idea is to find out whether it is possible to revoke a compromised protected template and replace it with a *different* protected template originating from the same unprotected template. This section considers the techniques used in the literature to evaluate the *cancellability* of non-invertible fingerprint template protection schemes.

In the literature, there are two general approaches towards the *cancellability* analysis for a non-invertible fingerprint template protection scheme. The first approach, which has become less common with the advancement of this field of research, involves simply stating that a fingerprint template protection scheme is cancellable due to the possibility of changing the non-invertible transform or a set of external parameters used in the generation of the protected template. In this case, the *cancellability* is assumed to be rather self-explanatory. Examples of publications in which this approach is adopted include [3, 20, 69, 137, 144]. The second, more dominant approach, involves evaluating a method's *cancellability* by generating multiple protected templates from the *same* unprotected template and then attempting to match them. If the probability of achieving a match in this case is low (e.g., approximately equal to the probability of matching two protected templates originating from *different* unprotected templates), this serves as proof that the fingerprint template protection scheme is effectively capable of establishing different identities from the same fingerprint; consequently, it is concluded that the fingerprint template protection scheme is *cancellable*. Examples of publications in which this approach is adopted include [65, 108, 109, 133, 134, 138, 139, 141, 145-147, 149].

In the evaluation of our new fingerprint template protection scheme, which is proposed in Chapter 5, we adopt both evaluation techniques discussed above. In particular, we first consider the *cancellability* of our method from an intuitive perspective, and we then prove its *cancellability* in terms of the probability that two protected templates originating from the same fingerprint would match (for more details, see Chapter 12).

An important consideration in the evaluation of a fingerprint template protection scheme's cancellability is the number of times that a compromised protected template can be cancelled and replaced with a new protected template from the same unprotected template. Currently, however, this point is rarely taken into account in the literature, which means that, at present, one is unlikely to see a quantitative comparison of the *cancellability* of different fingerprint template protection schemes. The only two publications on non-invertible fingerprint transforms that we have come across, in which an attempt is made to quantify cancellability in these terms, are [109, 142]. In [142], the number of different templates that can be generated from a single fingerprint is estimated in terms of the number of possible permutations of a fingerprint's multi-line code, and [109] relates the number of different templates from the same fingerprint to the number of changing functions that can be generated. While [109, 142] must be commended on their efforts, perhaps the reason that similar cancellability evaluations are lacking in the literature is that the definition of "the number of different protected templates that can be generated from the same unprotected template" is somewhat ill-defined. For example, at what point would we say that the limit has been reached: when enough of a person's protected templates have been collected and correlated to reveal the original template[10], or when we run out of non-invertible transforms/external parameters that incorporate diversity into the protected templates? In the *cancellability* analysis of our new fingerprint template protection scheme, which is proposed in Chapter 5, we adopt the latter approach. This is because the nature of our method ensures that the protected template remains cancellable even if a person's entire fingerprint is revealed to an attacker (for more details, see Chapter 12).

4.5.3 How is Diversity Measured?

A fingerprint template protection scheme is considered to possess the *diversity* property if it is capable of generating multiple uncorrelated protected templates from the same unprotected template. The idea is that, if a person uses different protected templates generated from the same unprotected template to enrol into different applications, it should be impossible to cross-match that person across those applications' databases, thereby preventing tracking. Due to its similarity to the *cancellability* property, in the literature *diversity* is commonly regarded as synonymous with *cancellability*. Consequently, the *same* analysis (i.e., that described in Section 4.5.2) is conducted to prove that a fingerprint template protection scheme satisfies both the *cancellability* and *diversity* requirements.

[10] See the discussion on the Record Multiplicity Attack in Section 4.5.1.

When evaluating the *diversity* of our new fingerprint template protection scheme, which is proposed in Chapter 5, we take the analysis a step further. In particular, we adopt a stricter definition of *diversity*, which states that two protected templates originating from the same fingerprint can only be considered truly *diverse* if they are *fully unlinkable*, where *unlinkability* is considered on a deeper level than that implied by a direct match (for more details, see Chapter 12).

4.5.4 How is Performance Measured?

The final of the four requirements for an ideal fingerprint template protection scheme concerns the performance of the recognition system in which this scheme is implemented. More specifically, it is important that the use of the protected fingerprint templates does not significantly degrade the recognition accuracy that is attainable when matching in the original, unprotected domain.

Note that, in the design of a non-invertible fingerprint template protection scheme, there is a well-known trade-off between recognition accuracy and non-invertibility: the higher the non-invertibility, the less *unique* the protected template becomes and thus the worse the recognition accuracy. An extreme example would be a non-invertible transform that maps *every* fingerprint template to the *same* protected template: While it would be essentially impossible to determine which original template a particular protected template came from, the recognition accuracy would be extremely poor since every person's protected template would be the same as every other person's. Because of this trade-off, it is expected that the performance of a recognition system employing protected templates that were generated by a non-invertible transform would generally be worse than the performance of the original system, which uses unprotected fingerprint templates. The exception to this rule would be a non-invertible fingerprint template protection scheme that incorporates some external information to introduce a higher level of diversity into the protected templates. In this case, assuming that this external information is kept secret, it is possible for the protected templates to be even more discriminative than their unprotected counterparts due to the increase in entropy as a result of the extra information.

The way in which the performance of a non-invertible fingerprint template protection scheme is evaluated is fairly standard in the literature, because it generally follows the procedure used to evaluate the performance of a traditional fingerprint recognition system. In particular, a fingerprint database is chosen and a protected fingerprint template is generated for each fingerprint in the database: the *same* non-invertible transform is applied to all samples of the *same* fingerprint, and a *different* non-invertible transform is applied to each

person. Then, each protected template is compared to every other protected template. The percentage of protected templates that originate from the *same* fingerprint but do *not* match is used to denote the system's False Reject Rate (FRR). The percentage of protected templates that originate from *different* fingerprints but *match* is used to denote the system's False Accept Rate (FAR).

It has recently become common for researchers to provide a complementary evaluation on the performance of their proposed non-invertible fingerprint template protection scheme in the scenario where an attacker is assumed to have access to the transform and any external parameters used to generate the protected template. This is frequently referred to as the stolen-token scenario, which was thus named after [90] showed that the performance of the popular BioHashing scheme [74] becomes worse than the performance of the original fingerprint recognition system when the user's external token is stolen by an attacker[11]. To evaluate this scenario for a non-invertible fingerprint template protection scheme in the literature, the *same* non-invertible transform and any additional external parameters are applied to *every* fingerprint in the database, and the FAR and FRR are then computed. The resulting performance is usually compared to the performance of the method in the *normal* scenario (where an attacker is not privy to the transform or any additional user-specific external information). Sometimes, the performance resulting from protected fingerprint templates is also compared to the performance obtained when the templates are matched in their unprotected format; however, this is not always the case, since, as stated earlier, we would already expect the performance in the protected case to be worse. Consequently, often the performance of the protected templates is left to stand on its own merits.

Our new fingerprint template protection scheme, which is proposed in Chapter 5, is evaluated both in the normal and "stolen-token" scenarios (see Chapters 5, 7, 8, and 9). Furthermore, the recognition accuracy of our method is compared to the recognition accuracy attainable using unprotected minutiae templates when a standard fingerprint recognition algorithm is used (see Chapter 5).

It must be noted that, while the method used to evaluate the performance of a non-invertible fingerprint template protection scheme in the literature is generally consistent across the board, there are often inconsistencies in the choice of the fingerprint database used for testing and the employed fingerprint feature extractor. For this reason, comparing different non-invertible fingerprint template protection schemes on the basis of their reported performance results is generally unfair and may result in misleading conclusions being drawn

[11] See Section 3.2.1 for a detailed discussion of the BioHashing scheme.

on the benefits of one method over another. While this potential danger is sometimes unfortunately ignored in comparisons made in the literature, we believe that such comparisons are unproductive and should be avoided unless the experimental conditions in the methods being compared are as similar as possible. In Chapter 9, we compare the performance of our new fingerprint template protection scheme to the performance of other non-invertible fingerprint template protection schemes in as fair a scenario as we could create.

4.6 SUMMARY

This chapter investigated the types of non-invertible transforms proposed in the literature for fingerprint template protection, and a discussion on the techniques used to evaluate the robustness of these methods was provided.

A perusal of the associated literature revealed that the most common features used in the development of non-invertible fingerprint template protection schemes are a fingerprint's minutiae. The focus of the rest of the chapter, and indeed the thesis overall, was then narrowed to minutiae-based non-invertible fingerprint template protection mechanisms.

It was established that misalignment is the main type of intra-class variance focused on in the development of non-invertible fingerprint template protection schemes, and that there are two main approaches for dealing with misalignment: establishing a global reference point (such as the core or a minutia) relative to which all the minutiae in a fingerprint are expressed, or else using self-aligned local minutiae structures.

The existing non-invertible fingerprint template protection schemes were classified into two main categories: *perturbation-based* non-invertible transforms and *histogram-based* non-invertible transforms. Perturbation-based techniques achieve non-invertibility by mapping multiple minutiae to the same location and/or superimposing several minutiae mappings into the same space to effectively hide the original minutiae locations. Histogram-based techniques transform the original minutiae template into one or more fixed-length histograms, which hide the original minutiae attributes by storing only the (often binarised) *number* of minutiae present in certain parts of a fingerprint, instead of the minutiae locations and orientations. A third category of non-invertible approaches involves representing the minutiae template in terms of local minutiae structures only; however, it was found that this type of approach has not yet had a significant amount of focus in the literature.

Contemplating the nature of the existing non-invertible fingerprint template protection schemes in general, it was established that they all use the *entire* minutiae template in generating the protected template. In this case, the security of the protected template relies

upon the level of non-invertibility provided by the transform, which can be quite limited. An alternative approach, which achieves non-invertibility by using only a *small portion* of the entire minutiae template, was suggested, and it was stated that this concept is employed in the development of our new fingerprint template protection scheme in Chapter 5.

Finally, this chapter considered the techniques used to evaluate the robustness of the existing non-invertible fingerprint template protection schemes in terms of the four characteristics of an ideal biometric template protection scheme: non-invertibility, cancellability, diversity and performance. It was found that non-invertibility is generally either simply stated or proven but not quantified, or else it is quantified in terms of an estimation of the complexity of reconstructing the original minutiae template from its protected counterpart via brute force. A related evaluation concerns the possibility of reconstructing the original minutiae template from multiple protected templates, which, although important, was found to be often neglected in the literature. It was determined that cancellability and diversity are generally evaluated in the same way, which involves either simply stating the fact or else generating multiple protected templates from the same fingerprint and then matching them in an attempt to prove that the protected templates are sufficiently different from each other. While it would be useful for cancellability and diversity to also be quantified in terms of the number of possible cancellations and replacements, this seems to be very uncommon in the literature. Performance was found to be evaluated in terms of the FAR and FRR of the protected templates, which is sometimes compared to the FAR and FRR of the original, unprotected minutiae templates. It was further discovered that it has recently become common to provide an additional performance evaluation in the "stolen-token scenario", which assumes that the non-invertible transform and any associated external parameters are known to an attacker.

Chapter 5

A Non-invertible Cancellable Fingerprint Construct based on Compact Minutiae Patterns

This chapter proposes a new fingerprint template protection scheme, which, although fundamentally different from the currently existing techniques, most snugly fits into the *non-invertible transforms* category of fingerprint template protection approaches. In Chapter 4, it was established that existing non-invertible fingerprint template protection schemes generally use the *entire* minutiae template in generating the protected fingerprint template. Since it is difficult to design a good non-invertible transform, it was concluded that a safer approach might be to achieve non-invertibility by using only a *small portion* of the entire minutiae template in generating the protected template. The non-invertibility of our new fingerprint template protection scheme, which is proposed in this chapter, relies on this concept of minimising the amount of fingerprint information used. This new scheme and its associated analysis form the main contributions of this thesis.

Note: *A shortened version of the material presented in this chapter has been published in [21].*

5.1 INTRODUCTION

In fingerprint recognition, it is generally considered that, the more information about two fingerprints that we have, the easier it is to determine whether or not they come from the same finger. For this reason, current non-invertible fingerprint template protection schemes, which were reviewed in Chapter 4, focus on transforming a fingerprint's *entire* minutiae template into a protected version of its former self. While this makes sense from the point of view of recognition accuracy, basing the protected template on the entire original template is dangerous in terms of securing the original template. This is because the complexity of

reconstructing the original minutiae template from the protected template depends only on the level of non-invertibility provided by the transform used to generate the protected template. Since it is difficult to design a good non-invertible transform, the resulting protected template often has limited non-invertibility in the event that the transform and any associated external parameters are known to an attacker. For this reason, the focus of this chapter, and this thesis in general, is on developing a new fingerprint template protection scheme, whose non-invertibility is the result of using only a *small portion* of the entire minutiae template, such that the majority of the original fingerprint remains unrevealed in the event of compromise.

This chapter proposes a new fingerprint construct, which is intrinsically a non-invertible fingerprint template protection scheme. The crux of our scheme entails the representation of a fingerprint by a single Pattern constructed using a small subset of minutiae from the corresponding minutiae template. The sparsity of the resulting Pattern makes it impossible to reconstruct the original fingerprint template and it ensures that the Pattern is cancellable in the event of compromise. This construct was specifically designed for deployment in cooperative-user civilian fingerprint recognition applications, *not* for forensics.

The remainder of this chapter is dedicated to introducing our new fingerprint construct and on gauging its potential for use as a fingerprint template protection scheme in practice, with a specific focus on the attainable recognition accuracy.

5.2 THE PROPOSED FINGERPRINT CONSTRUCT

The proposed fingerprint construct consists of a single Pattern generated using a small subset of minutiae from the corresponding minutiae template. The inspiration for this new fingerprint construct came from the pattern-based mechanism that can be used as a password to unlock an Android smartphone: The smartphone user is provided with a matrix of dots, from which they select a certain number of these dots and connect them in a particular order. Authentication is successful, and the user's phone is therefore unlocked, if the correct Pattern (i.e., the one chosen during password set-up) is entered. The difference with our idea lies in that, instead of using a standard matrix of dots from which a pattern must be chosen, a fingerprint's minutiae are used to form the dots instead. This way, each fingerprint has its own, unique dot 'alphabet' to work with.

A Pattern that consists of N minutiae is referred to as an N-node Pattern, and it is described by two sets of features: local features and global features. The local feature set characterises the *shape* of the Pattern, while the global feature set represents both the *location* and *orientation* of the Pattern relative to the fingerprint core.

5.2.1 Rules for Pattern Formation

A Pattern consists of a few minutiae (henceforth referred to as *nodes*) connected in a particular order via straight lines. There are two constraints on the types of Patterns that may be formed. Firstly, each node must have exactly one line entering it and one line exiting from it, such that the latter line immediately follows the former line. Secondly, the Pattern must form a closed shape, which means that the last line must finish at the first node. In other words, the Pattern is required to satisfy the definition of a polygon. This means that an *N*-node Pattern is essentially an *N*-gon (i.e., a polygon with *N* sides). Figure 5.1(a) shows a valid type of Pattern, and Figures 5.1(b) – 5.1(d) show Patterns that are invalid due to their noncompliance with these constraints.

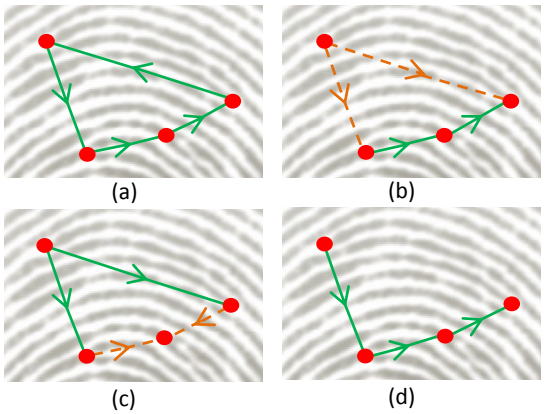


Figure 5.1: Valid and invalid 4-node Patterns, where the nodes (minutiae) are labelled by red dots: (a) VALID; (b) INVALID: The two dashed orange lines are exiting from the same node; (c) INVALID: The two dashed orange lines are entering the same node; (d) INVALID: Does not form a closed shape.

Note that the reliance of our proposed fingerprint construct on minutiae Patterns may initially liken our method to non-invertible fingerprint template protection schemes that employ local minutiae structures[12]. There are two important differences between our fingerprint construct and these other techniques, however. Firstly, our approach involves the construction of only a *single* Pattern using a small subset of minutiae from the entire minutiae template, while most of these other methods create a structure to describe *each minutia* in the template. Consequently, the fingerprint template resulting from our proposed fingerprint construct is intuitively more secure, since it employs only a small subset of the available minutiae. The sparsity of our *N*-node Patterns suggests that our proposed fingerprint construct is also less demanding in terms of the required amount of storage space. Secondly, most existing non-invertible fingerprint template protection schemes that employ minutiae structures require that each structure consists of *neighbouring* minutiae. On the other hand,

[12] See Sections 4.4.1 and 4.4.2 for examples.

the N minutiae used in the generation of an N-node Pattern resulting from our proposed fingerprint construct can essentially come from anywhere in the fingerprint. Consequently, our proposed fingerprint construct allows for greater diversity in the resulting minutiae structure.

5.2.2 Local Features

The local[13] feature set of an N-node Pattern consists of the following attributes, which are illustrated in Figure 5.2 for a 4-node Pattern:

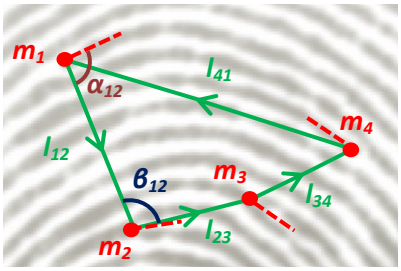


Figure 5.2: Local features of the Pattern from Figure 5.1 (a). The dashed red lines represent the corresponding minutiae orientations, θ. To avoid cluttering, α and β are labelled only for the first connection line.

- l_{ij}: The length of the connection line from minutia m_i to minutia m_j, which is the Euclidean distance between the locations of m_i and m_j. The location of a minutia is expressed in terms of the (column, row) indices of the corresponding pixel in the fingerprint image plane. Since the origin in an image is commonly placed at the top left corner, this means that the column indices increase to the right and the row indices increase downwards. We may thus consider the image plane as a Cartesian coordinate system in which the column indices represent x-coordinates and the row indices represent y-coordinates. Let (x_i, y_i) denote the location of minutia m_i and (x_j, y_j) denote the location of minutia m_j in the fingerprint image; then l_{ij} is calculated using Equation (5.1), where $i = 1, 2, \ldots, N$, and $j = 2, 3, \ldots, N, 1$.

$$l_{ij} = \sqrt{(x_j - x_i)^2 + (y_j - y_i)^2} \tag{5.1}$$

[13] Note that the word "local" in this context should not be confused with the word "local" used to describe *local* minutiae structures employed in several fingerprint template protection schemes discussed in Sections 4.4.1 and 4.4.2. In the case of our proposed fingerprint construct, the word "local" is used to refer to features derived from the relationships *between minutiae* (as opposed to between minutiae and the core point, which are referred to as "global" features – see Section 5.2.3). In the context of "local" minutiae structures, however, the word "local" is generally used to refer to the fact that the minutiae in a structure are neighbours or occur in the same (small) part of the fingerprint.

- α_{ij} and β_{ij}: Let φ_{ij} denote the angle of the connection line from minutia m_i to minutia m_j. If we continue to represent the locations of minutiae m_i and m_j using (x_i, y_i) and (x_j, y_j), respectively, then φ_{ij} is calculated via Equation (5.2), where $i = 1, 2, \ldots, N$, and $j = 2, 3, \ldots, N, 1$.

$$\varphi_{ij} = \tan^{-1}\left(\frac{y_j - y_i}{x_j - x_i}\right) \tag{5.2}$$

Further, let θ_i and θ_j represent the orientations of minutiae m_i and m_j, respectively. Note that φ_{ij}, θ_i and θ_j all increase in the clockwise direction from the horizontal and lie in the range [0°, 360°). Then α_{ij} is the angle difference between φ_{ij} and θ_i, which is calculated using Equation (5.3), and β_{ij} is the angle difference between φ_{ij} and θ_j, which is calculated using Equation (5.4):

$$\alpha_{ij} = min\left(\left|\varphi_{ij} - \theta_i\right|, 360° - \left|\varphi_{ij} - \theta_i\right|\right) \tag{5.3}$$

$$\beta_{ij} = min\left(\left|\varphi_{ij} - \theta_j\right|, 360° - \left|\varphi_{ij} - \theta_j\right|\right) \tag{5.4}$$

Note that the use of Equations (5.3) and (5.4) results in a constriction of α_{ij} and β_{ij} to within the range [0°, 180°), even though φ_{ij}, θ_i and θ_j all lie in the range [0°, 360°), thereby destroying the information pertaining to which of the two angles being compared (i.e., φ_{ij} or θ_i in Equation (5.3), and φ_{ij} or θ_j in Equation (5.4)) is larger. This was a conscious design decision in order to deal with the possibility of φ_{ij} and θ_i or φ_{ij} and θ_j accidentally *swapping* in practice. For example, if φ_{ij} and θ_i are very close, then a slight error in the detection of the locations of minutiae *i* and *j*, which are used to calculate φ_{ij}, or an error in the determination of θ_i, could essentially cause φ_{ij} and θ_i to *swap*, such that the larger angle now becomes the smaller angle, and vice-versa. In the absence of the constraints incurred by the use of Equation (5.3), the effect of this *swap* could be that the pre-swap α_{ij} does not match the post-swap α_{ij}; similarly for the β_{ij} attributes. In this case, an α_{ij} whose value is 359°, for example, would not match an α_{ij} with a value of 1°, whereas, in reality, these should be considered to match. The use of Equations (5.3) and (5.4) ensure that the aforementioned scenario is avoided in practice[14].

[14] Note that this situation is actually dealt with during Pattern matching (see Section 5.2.5), so, in hindsight, the constrictions placed on α and β as a result of Equations (5.3) and (5.4) could have been removed during the computation of these attributes. Section 8.5 considers an improvement to our proposed fingerprint construct, in which this scenario is considered.

The fact that Equations (5.3) and (5.4) destroy the information pertaining to which of the two angles being compared (i.e., φ_{ij} or θ_i in Equation (5.3), and φ_{ij} or θ_j in Equation (5.4)) is larger implies that the use of these equations incurs an information loss in the ensuing α and β attributes, which may negatively impact upon Pattern *uniqueness* and thus decrease the recognition accuracy of our proposed fingerprint construct. In particular, it would appear as if there is ambiguity in the signs of α_{ij} and β_{ij}. For example, considering Equation (5.3), we know that:

$$\varphi_{ij} - \theta_i = \pm\alpha_{ij} \ ,$$

which means that:

$$\theta_i = \begin{Bmatrix} \varphi_{ij}, & if\,\alpha_{ij} = 0° \\ \varphi_{ij} - \alpha_{ij}, & if\,\alpha_{ij} > 0° \\ \varphi_{ij} + \alpha_{ij}, & if\,\alpha_{ij} < 0° \end{Bmatrix}, \qquad 0° \le \theta_i < 360°$$

Thus, we may conclude that, for a particular φ_{ij}, we would be able to use Equation (5.3) on its own to determine a *unique* value for θ_i only when $\alpha_{ij} = 0°$, which would happen when $\theta_i = \varphi_{ij}$. However, in practice it is more likely that $\theta_i \neq \varphi_{ij}$; therefore, we may surmise that, more often than not, using Equation (5.3) on its own would be expected to result in *two* possible values for θ_i: if α_{ij} is positive then $\theta_i = \varphi_{ij} - \alpha_{ij}$, and if α_{ij} if negative then $\theta_i = \varphi_{ij} + \alpha_{ij}$. Consequently, for a single value of φ_{ij}, *either* of the two possible values for θ_i would produce the same α_{ij}, suggesting that, in the case where $\theta_i \neq \varphi_{ij}$, Equation (5.3) cannot be used on its own to determine the *true* value of θ_i, since that information is lost as a result of losing the sign for α_{ij}. In practice, however, the *true* θ_i can actually be determined by considering the value of the β attribute corresponding to the *same minutia*. For example, considering Figure 5.2, we could use Equations (5.3) and (5.4) to examine α_{12} and β_{14} simultaneously in order to uniquely determine the orientation of minutia m_1, i.e., θ_1, as follows:

Use Equations (5.5) and (5.6), which are based on Equations (5.3) and (5.4), to determine α_{12} and β_{41}, respectively:

$$\alpha_{12} = min(|\varphi_{12} - \theta_1|, 360° - |\varphi_{12} - \theta_1|) \quad (5.5)$$

$$\beta_{41} = min(|\varphi_{41} - \theta_1|, 360° - |\varphi_{41} - \theta_1|) \quad (5.6)$$

From Equation (5.5):

$$\varphi_{12} - \theta_1 = \pm\alpha_{12} \therefore \theta_1 = \begin{Bmatrix} \varphi_{12}, & if\alpha_{12} = 0° \\ \varphi_{12} - \alpha_{12}, & if\alpha_{12} > 0° \\ \varphi_{12} + \alpha_{12}, & if\alpha_{12} < 0° \end{Bmatrix}, \quad 0° \leq \theta_1 < 360° \quad (5.7)$$

From Equation (5.6):

$$\varphi_{12} - \theta_1 = \pm\beta_{41} \therefore \theta_1 = \begin{Bmatrix} \varphi_{12}, & if\beta_{41} = 0° \\ \varphi_{12} - \beta_{41}, & if\beta_{41} > 0° \\ \varphi_{12} + \beta_{41}, & if\beta_{41} < 0° \end{Bmatrix}, \quad 0° \leq \theta_1 < 360° \quad (5.8)$$

Since a *unique* solution for θ_1 can be established in the case where $\alpha_{12} = 0°$ or $\beta_{41} = 0°$, henceforth it is only necessary to consider the cases where $\alpha_{12} > 0°$ or $\alpha_{12} < 0°$, and $\beta_{41} > 0°$ or $\beta_{41} < 0°$. So, from Equations (5.7) and (5.8), respectively, we obtain the following:

Solution Set 1: $\theta_1 = \{\varphi_{12} - \alpha_{12},\ \varphi_{12} + \alpha_{12}\}$

Solution Set 2: $\theta_1 = \{\varphi_{41} - \beta_{41},\ \varphi_{41} + \beta_{41}\}$

Since we have two solution sets for a single variable, θ_1, it is possible to *uniquely* determine θ_1 for a given φ_{12} and φ_{41}, i.e., a single solution from Solution Set 1 will be the *same* as a solution from Solution Set 2, and this will be the *true* θ_1. The only case in which this would *not* be possible would be when $\varphi_{12} = \varphi_{41}$, since the two solution sets would then be the same and the sets would intersect at two points instead of one.

From the analysis above, we may conclude that, in general, restricting α_{ij} and β_{ij} to the range [0°, 180°) via the use of Equations (5.3) and (5.4) would not incur an information loss in practice. While some information loss may occur when a line entering a minutia is collinear with the line exiting the same minutia, it is unlikely that such an occurrence will be frequent in practice. Therefore, we may expect that, usually, the signs of the α and β

attributes will be disambiguated in practice[15]. Consequently, we may conclude that the use of Equations (5.3) and (5.4) to generate the α and β attributes would be expected to have a negligible effect on the recognition accuracy of our proposed fingerprint construct. This claim is empirically evaluated in Section 8.5.

5.2.3 Global Features

A Pattern's global features consist of the following attributes, which are illustrated in Figure 5.3(a) and Figure 5.3(b), respectively:

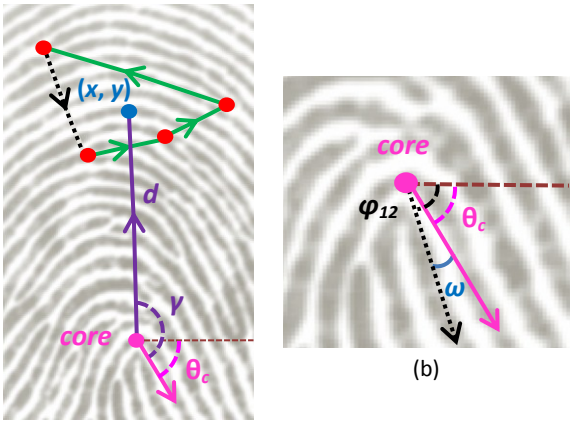


Figure 5.3: Global features of the Pattern from Figure 5.2: (a) Cartesian, (x, y), and polar, (d, γ), coordinates depicting the location of the Pattern centroid relative to the core; (b) Pattern orientation, ω, is the difference between the orientation of the first connection line (coloured in black and dotted in (a) and (b)), φ_{12}, and the core angle, θ_c.

- (x, y): The location of the Pattern's centroid. Once again allowing (x_i, y_i) to denote the location of minutia m_i in the fingerprint image plane, the x and y coordinates of the centroid of an N-node Pattern (i.e., a Pattern consisting of N minutiae) are calculated using Equations (5.9) and (5.10), respectively:

$$x^{initial} = \frac{1}{N}\sum_{i=1}^{N} x_i \tag{5.9}$$

$$y^{initial} = \frac{1}{N}\sum_{i=1}^{N} y_i \tag{5.10}$$

The superscript "*initial*" next to the outputs of Equations (5.9) and (5.10) indicate that these values do not represent the final location of the Pattern's centroid. In order to deal

[15] While it may be possible to derive a contrived example of a Pattern in which this is not true, the resulting Pattern is likely to be unnatural and therefore highly unlikely to exist in anyone's fingerprint.

with translational and rotational differences between the acquired fingerprint images, the initial location of the Pattern's centroid is next expressed in terms of polar coordinates relative to the fingerprint core. Let (x_c, y_c) denote the location of the core point in the corresponding fingerprint image, and let θ_c denote the core angle. Then the radial distance of the Pattern's centroid relative to the core is calculated using Equations (5.11) and (5.12), and the radial angle of the Pattern's centroid relative to the core is calculated using Equation (5.13):

$$d = \sqrt{(x^{initial} - x_c)^2 + (y^{initial} - y_c)^2} \tag{5.11}$$

$$\theta_l = \tan^{-1}\left(\frac{y^{initial} - y_c}{x^{initial} - x_c}\right) \tag{5.12}$$

$$\gamma = min(|\theta_l - \theta_c|, 360° - |\theta_l - \theta_c|) \tag{5.13}$$

Note that θ_l and γ both increase in the clockwise direction from the horizontal, but θ_l lies in the range [0°, 360°) while γ lies in the range [0°, 180°). Expressing the Pattern's location in terms of these polar coordinates, (d, γ), would be problematic for Patterns whose centroid is located near the core. If any of the minutiae making up a Pattern are detected in a slightly different location in another sample of the same fingerprint, the Pattern's centroid may occasionally fall onto a different side of the core. This would cause the γ value to vary by considerable amounts. Since comparison of two Patterns' locations would require checking the differences between their d and γ attributes separately, two Patterns that are close to their corresponding cores (in terms of d) but located on different sides of the core would not match (even though they should) on account of their γ values being very different. In order to avoid this issue, the polar coordinates are converted to Cartesian coordinates via Equation (5.14) and Equation (5.15) to produce the final (x, y) coordinates of the pattern's centroid:

$$x = d\cos\gamma \tag{5.14}$$

$$y = d\sin\gamma \tag{5.15}$$

Cartesian coordinates provide more tolerance for location differences in Patterns whose centroids lie closer to the core. This is because the range of *cos* and *sin* functions is between -1 and 1, so the larger the d value the greater the magnitudes of the resulting x- and y-coordinates. Since the comparison of two Patterns' locations is now calculated as

the Euclidean distance between their (x, y) coordinates (see Section 5.3.6), then Patterns whose centroids lie closer to the core would have smaller x- and y-coordinates and thus the Euclidean distance between them would be smaller than for Patterns that are further away from the core.

Note that Equation (5.13) was used to restrict γ to the range [0°, 180°) for a similar reason as to why Equations (5.3) and (5.4) were used to restrict α_{ij} and β_{ij}, respectively, to the range [0°, 180°), i.e., to ensure that an accidental *swap* between θ_l and θ_c in practice does not impact on the resulting γ. Because of this restriction, however, the y-coordinate pertaining to the location of the Pattern's centroid (see Equation (5.15)) can only take on positive values (since $0 \leq \sin\gamma \leq 1$), which means that we cannot be sure whether the *true* Pattern location is at $+y$ or at $-y$. A disadvantage of this ambiguity may be a slight decrease in Pattern *uniqueness* and thus the recognition accuracy of our proposed fingerprint construct; however, this would only occur in the extremely unlikely scenario in which there happen to exist two N-node Patterns that are the exact mirror images of each other, with one of the Patterns residing at $+y$ and the other at $-y$. We may thus reason that the ambiguity in the true Pattern location introduced as a result of using Equation (5.13) would be expected to have a negligible effect on the recognition accuracy of our proposed fingerprint construct. This claim is empirically evaluated in Section 8.5.

- ω: The orientation of the Pattern. The orientation of the Pattern's *first* connection line is used to represent the orientation of the Pattern as a whole. Let φ_{12} denote the angle of the first connection line (i.e., from minutia 1 to minutia 2) relative to the horizontal, where φ_{12} increases in the clockwise direction and lies in the range [0°, 360°). Note that φ_{12} is calculated via Equation (5.16), where (x_1, y_1) and (x_2, y_2) denote the locations of minutia 1 and minutia 2, respectively, in the fingerprint image.

$$\varphi_{12} = \tan^{-1}\left(\frac{y_2 - y_1}{x_2 - x_1}\right) \tag{5.16}$$

Then the Pattern's orientation, ω, is calculated as the difference between φ_{12} and the core angle, θ_c, using Equation (5.17):

$$\omega = min(|\varphi_{12} - \theta_c|, 360° - |\varphi_{12} - \theta_c|) \tag{5.17}$$

Note that Equation (5.17) was used to restrict ω to the range [0°, 180°) for a similar reason as to why Equations (5.3) and (5.4) were used to restrict α_{ij} and β_{ij}, respectively, to the range [0°, 180°), and why Equation (5.13) was used to restrict γ to the range [0°, 180°), i.e., to ensure that an accidental *swap* between φ_{12} and θ_c in practice does not impact on the resulting ω. Due to this restriction, Equation (5.17) introduces ambiguity as to the Pattern's *true* orientation. In particular, forcing ω to lie in the range [0°, 180°), instead of allowing it to lie in the range [0°, 360°), means that two *N*-node Patterns that are the exact mirror images of each other, with the mirror line being the core angle, θ_c, would match. This would only occur, however, if all the Pattern attributes in these two *N*-node Patterns were the same, except for that fact that one of the Patterns had an orientation equal to $\varphi_{12} - \theta_c$ and the other one had an orientation equal to $\varphi_{12} + \theta_c$. Since such a scenario is unlikely to occur in practice, we may reasonably conclude that this loss of information concerning a Pattern's true orientation would, in practice, have a negligible effect on the recognition accuracy of our proposed fingerprint construct. This claim is empirically evaluated in Section 8.5.

5.2.4 Constructing the Pattern's Feature Vector

Once a Pattern's validity (in terms of the constraints outlined in Section 5.2.1) has been verified, its local and global attributes are collated into a feature vector, which is stored in the database for verification purposes. The feature vector of person *P*'s *N*-node pattern has the following format:

$$v^P = [l_{12}^P, \alpha_{12}^P, \beta_{12}^P, l_{23}^P, \alpha_{23}^P, \beta_{23}^P, \ldots, l_{N1}^P, \alpha_{N1}^P, \beta_{N1}^P, x^P, y^P, \omega^P]$$

Note the following two points:

- Since the Pattern's features are all relative either to each other or to the fingerprint core, the resulting feature vector is invariant to translation and rotation of the underlying fingerprint image.
- The order in which the Pattern's attributes are placed in the resulting feature vector is important for matching two feature vectors (see Section 5.2.5). The ordering of the attributes depends on the order in which a Pattern's *N* constituent minutiae are connected; therefore, the order in which the minutiae (nodes) are connected is important for differentiating two Patterns. For example, if the nodes in Figure 5.2 were connected in the

clockwise, instead of the anticlockwise, direction, the resulting Pattern would be considered different to the one currently portrayed by Figure 5.2. Similarly, even if the nodes were connected in the same direction but the starting minutiae were different, the two Patterns would be deemed different.

Considering the feature vector of an N-node Pattern, let us estimate the number of bits that would be required to store a person's reference N-node Pattern in a database[16]. Since, as established in Sections 5.2.2 and 5.2.3, the α, β and ω attributes all lie in the range [0°, 180°), we need a maximum of 180 bits to represent each of these attributes in the Pattern's feature vector; therefore, 1 byte would be sufficient to store each α, β and ω attribute. Since the l, x and y attributes are all derived from the (column, row) pixel indices corresponding to the locations of minutiae and/or core points in the underlying fingerprint image, the number of bits required to represent each of these attributes essentially depends on the image size. The size of the fingerprint image in turn depends on the size of the scanning surface used to acquire the fingerprint image. For example, the Futronic FS88 optical scanner [210], which was used to acquire the fingerprint images for our cooperative-user fingerprint database in Chapter 6, produces fingerprint images with a width of 320 pixels and a height of 480 pixels. The *maximum* possible range of minutiae and core x-coordinates in this image would thus be [0, 320], and the *maximum* possible range of minutiae and core y-coordinates would be [0, 480]. Consequently, 1 byte ($2^8 = 256$ possible values) may be insufficient for storing the l, x and y Pattern attributes; however, 2 bytes ($2^{16} = 65{,}536$ possible values) would be more than enough for this purpose, regardless of which fingerprint scanner is used[17]. We may thus estimate that the total amount of storage space required for the feature vector of an N-node Pattern is $2N + 1N + 1N + 2 + 2 + 1 = 4N + 5$ bytes (i.e., 2 bytes for each of the N l attributes + 1 byte for each of the N α attributes + 1 byte for each of the N β attributes + 2 bytes for the x attribute + 2 bytes for the y attribute + 1 byte for the ω attribute). Table 5.1 compares the number of bytes needed to store the feature vector of a single N-node Pattern as the Pattern size, N, increases from 3 to 5.

[16] Note that, in practice, the number of bits used to represent a Pattern's feature vector would depend on several factors, including the desired precision of each attribute, the available data types in a software implementation of our proposed fingerprint construct, etc. However, here we are only interested in providing the reader with a theoretical approximation of the amount of storage space required.

[17] Most modern fingerprint scanners used in practice would not be expected to produce a fingerprint image whose resolution exceeds 65,536 pixels.

Table 5.1: An estimation of the number of bytes used to store an *N*-node Pattern.

	Number of Bytes Needed to Store Feature Vector
$N = 3$	$(4 \times 3) + 5 = 17$
$N = 4$	$(4 \times 4) + 5 = 21$
$N = 5$	$(4 \times 5) + 5 = 25$

From Table 5.1, we can see that, as the Pattern size, N, increases by 1, the number of bytes required to store the resulting feature vector increases by 4. Since the smallest possible Pattern consists of 3 minutiae, we may conclude that the *minimum* amount of storage space required to store a Pattern would be **17 bytes**. Similarly, since the largest Pattern we propose consists of 5 minutiae, we may conclude that the *maximum* amount of storage space required in practice would be **25 bytes**.

Let us consider how the amount of storage space required by our proposed fingerprint construct compares with the number of bytes needed to store a fingerprint's entire minutiae template, which is the most common way of representing a fingerprint in practice today. We shall consider the typical case where each minutia is represented by 3 pieces of information: 2 location attributes and 1 orientation attribute (i.e., $m = \{x, y, \theta\}$). Since, as for the l, x and y Pattern attributes, the range of a minutia's x and y attributes depends on image size, we may conclude that 2 bytes is sufficient to represent each of those coordinates. The orientation, θ, is commonly represented either in the range [-180°, 180°) or [0°, 360°); in either case, 2 bytes are required to represent the θ attribute. Let us denote the total number of minutiae in the entire minutiae template by T. We may thus conclude that the total amount of storage space required to store a T-minutiae template is $6T$ bytes, i.e., $2T$ bytes for each minutia's x-coordinate + $2T$ bytes for each minutia's y-coordinate + $2T$ bytes for each minutia's θ. Since the number of minutiae in a fingerprint would generally be much larger than 3, 4 or 5, we can expect $T >> N$; therefore, $6T >> 4N + 5$. For example, in our cooperative-user fingerprint database (see Chapter 6), the *minimum* number of minutiae detected in a fingerprint was 25 and the *maximum* number[18] was 96. In this case, the *minimum* amount of storage space required to store an entire minutiae template would be $6T = 6 \times 25 =$ **150 bytes**, and the *maximum* amount of required storage space would be $6T = 6 \times 96 =$ **576 bytes**. Compare these numbers to the *minimum* of **17 bytes** required to store a single 3-node Pattern and the *maximum* of **25 bytes** needed to store a single 5-node Pattern (see Table 5.1). So, it is evident

[18] Since these minutiae were extracted *manually*, we can be confident that the total number of detected minutiae includes only *true* minutiae (i.e., no spurious minutiae). For more details on the minutiae extraction process, see Chapter 6.

that our proposed fingerprint construct incurs a significant reduction in the amount of storage space required to represent each user of a fingerprint recognition system in the system's database. Furthermore, while the number of bytes required to store a Pattern of a particular size remains the same for each person, the number of bytes required to store an entire minutiae template depends on the number of minutiae available in each person's fingerprint. This is another advantage of our proposed fingerprint construct, since allocation of memory for each person in a database is made easier by the fact that the amount of storage space required is predictable.

5.2.5 Comparing Pattern Feature Vectors during Matching

Our proposed fingerprint construct lends itself to two different authentication methods. The first method may be likened to the pattern-based unlocking functionality provided in Android smartphones. This method requires a user to choose a reference *N*-node Pattern during enrolment and 'draw' it on their reference fingerprint by connecting a certain set of minutiae from that fingerprint in a particular order. The user must then remember their chosen Pattern and, during authentication, they must 're-draw' the Pattern onto their query fingerprint using the minutiae from that fingerprint.

The second authentication method does not require the user to remember their reference Pattern or present it during authentication; rather, the user selects their reference *N*-node Pattern during enrolment (or they may elect the recognition system to choose a Pattern on their behalf), and the recognition system then searches for the same Pattern in the user's query fingerprint by trying out all possible *N*-node Patterns from the query minutiae until a matching Pattern is found.

We shall henceforth refer to the former authentication mechanism as *Two-Factor Authentication*, since the user is required to present both their fingerprint and the correct *N*-node Pattern for authentication purposes. Conversely, the latter method shall be referred to as *Single-Factor Authentication*, since the user is required to present only their fingerprint for authentication purposes.

Figure 5.4 illustrates the enrolment process for our Pattern method. Figures 5.5 and 5.6, respectively, depict *Two-Factor Authentication* and *Single-Factor Authentication*.

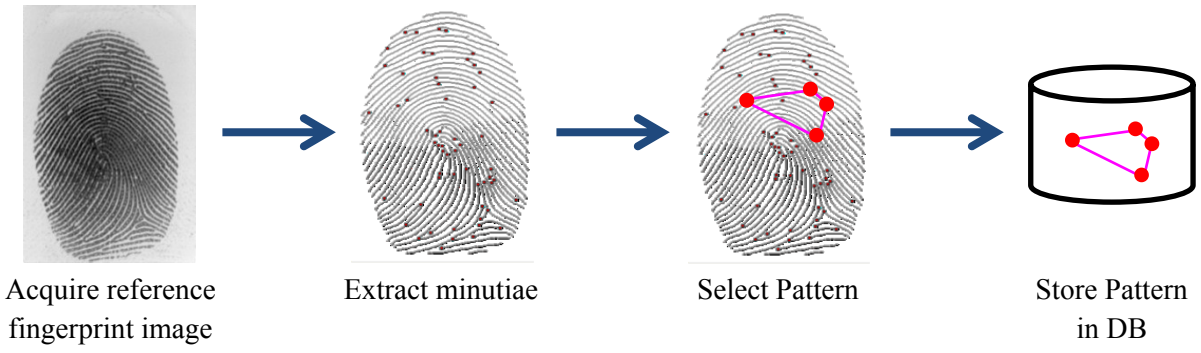


Figure 5.4: Enrolment of a user's (4-node) reference Pattern into the recognition system's database.

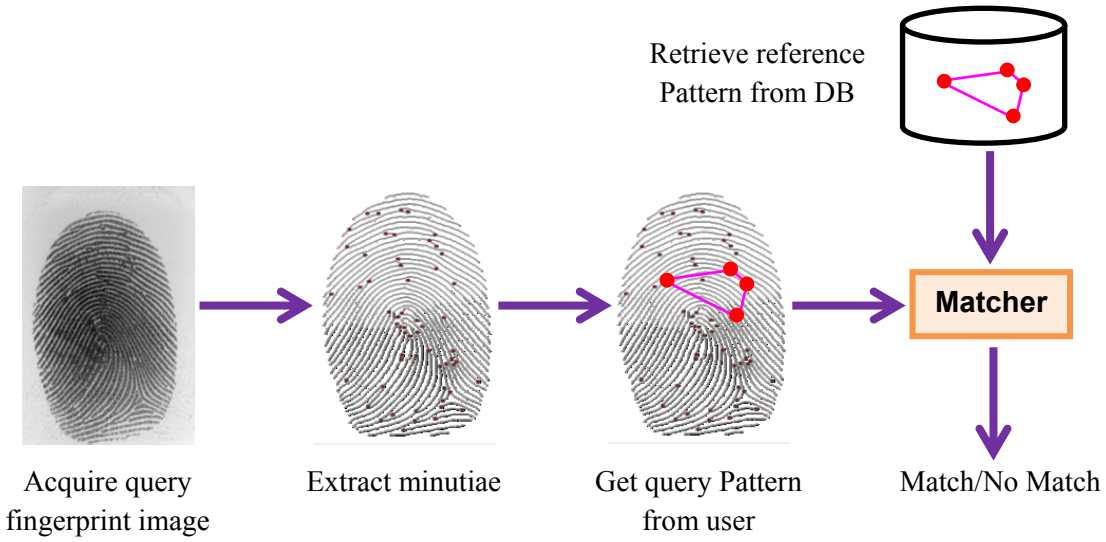


Figure 5.5: Two-Factor Authentication.

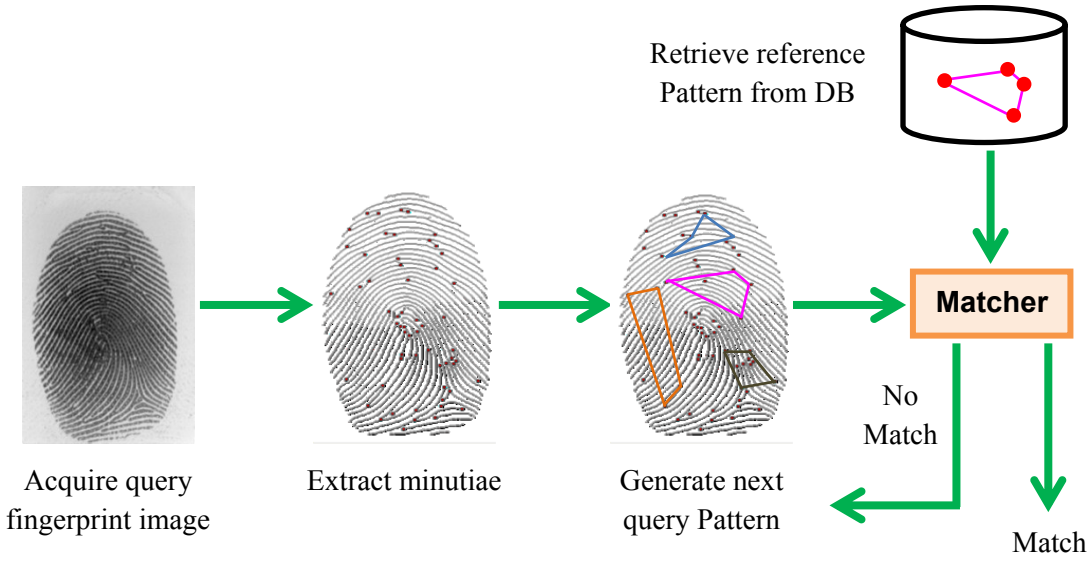


Figure 5.6: Single-Factor Authentication.

Regardless of which authentication method is used, however, the actual comparison between a reference and query Pattern pair remains the same. Let v^A and v^B denote the feature vectors corresponding to Patterns A and B, respectively. If the two Patterns consist of a different number of nodes, then verification will fail immediately; so, let us consider the case where both Patterns consist of N nodes. To compare v^A with v^B, we calculate the differences between the corresponding vector elements and check whether those differences are below the pre-determined thresholds required for a match: if so, then the features are said to match; otherwise, they do not match and verification fails. Note that *all* the features must match in order for the verification to be successful; any *single pair* of corresponding features that do not match will result in a failed verification attempt. Expressing this mathematically, verification will be successful if *all* of the following conditions are satisfied, and it will fail as soon as *any* of these conditions are violated:

$$|l_{ij}^A - l_{ij}^B| \leq \tau_l \tag{5.18}$$

$$\min(|\alpha_{ij}^A - \alpha_{ij}^B|, 360° - |\alpha_{ij}^A - \alpha_{ij}^B|) \leq \tau_{\alpha\beta} \tag{5.19}$$

$$\min(|\beta_{ij}^A - \beta_{ij}^B|, 360° - |\beta_{ij}^A - \beta_{ij}^B|) \leq \tau_{\alpha\beta} \tag{5.20}$$

$$\sqrt{(x^A - x^B)^2 + (y^A - y^B)^2} \leq \tau_{loc} \tag{5.21}$$

$$\min(|\omega^A - \omega^B|, 360° - |\omega^A - \omega^B|) \leq \tau_\omega \tag{5.22}$$

Note that, in Equations (5.18) to (5.22), $i = 1, 2, \ldots, N$; $j = 2, 3, \ldots, N, 1$; and τ_l, $\tau_{\alpha\beta}$, τ_{loc} and τ_ω are pre-determined thresholds for Pattern attributes l, α and β, (x, y), and ω, respectively. Threshold selection is discussed in Section 5.3.1.4.

5.3 SUITABILITY OF PROPOSED FINGERPRINT CONSTRUCT AS A FINGERPRINT TEMPLATE PROTECTION SCHEME

This section provides a preliminary analysis on the suitability of our proposed fingerprint construct to serve as a fingerprint template protection scheme. Recall from Section 3.1 that an ideal fingerprint template protection scheme must satisfy four requirements: non-invertibility, cancellability, diversity, and performance.

A fingerprint template protection scheme is considered to be *non-invertible* if the protected template cannot be *inverted* to reveal the original minutiae template and, thereby, reconstruct the underlying fingerprint. Since our proposed fingerprint construct entails the representation of a fingerprint by a single Pattern consisting of either 3, 4, or 5 minutiae, it is intuitively evident that a fingerprint's entire minutiae template *cannot* be reconstructed from

this Pattern. This is because the total number of minutiae available in a full minutiae template is generally much larger than 3, 4, or 5. A full analysis on the *non-invertibility* of our proposed fingerprint construct is presented in Chapters 10 and 11.

A fingerprint template protection scheme is considered to be *cancellable* if the protected template can be cancelled (or revoked) and replaced with a new template *from the same fingerprint*, in the event that the original template is compromised (e.g., stolen from the database). This is important for ensuring that a compromised fingerprint template does not render the underlying fingerprint useless for future authentication purposes, thereby effectively allowing a user to generate multiple 'passwords' from a single fingerprint. The fact that multiple *N*-node Patterns exist in a single fingerprint suggests that our proposed fingerprint construct indeed satisfies the *cancellability* property. A related property, which is also satisfied by our proposed fingerprint construct, is *diversity*. This property enables a user to enrol into different applications using a different *N*-node Pattern from the same fingerprint in each application, thereby mitigating the danger of abusing a person's privacy by cross-tracking them across different applications into which they have enrolled with the same fingerprint. A full analysis on the *cancellability* and *diversity* of our proposed fingerprint construct is presented in Chapter 12.

While the sparsity of our proposed fingerprint construct makes it intuitively evident that the non-invertibility, cancellability, and diversity properties of an ideal fingerprint template protection scheme are satisfied, the same characteristic of the construct may raise concerns regarding its ability to satisfy the fourth property: performance. Since it has traditionally been believed that there must exist a minimum of 12 matching minutiae between two fingerprints in order to consider the fingerprints as matching (e.g., see [14]), the use of only 3, 4, or 5 minutiae in an *N*-node Pattern generated using our proposed fingerprint construct may cast doubt on the attainable recognition accuracy. For this reason, the main focus of this chapter, besides introducing the new fingerprint construct, is to present results from a preliminary investigation conducted into evaluating its performance.

The remainder of this section presents the methodology and results of three experiments designed to provide some preliminary ideas on the suitability of our proposed fingerprint construct for recognition purposes in practice. The aim of the first experiment was to evaluate Pattern *uniqueness* in terms of the discriminability of different fingerprints using a single *N*-node Pattern. The aim of the second experiment was to measure the recognition accuracy of a standard fingerprint recognition algorithm, which uses full minutiae templates, and compare this to the performance of the proposed fingerprint construct from the first experiment. The

aim of the third experiment was to estimate the amount of time it takes to verify a person using the proposed fingerprint construct.

5.3.1 Experiment 1: Pattern Uniqueness

The aim of this experiment was to evaluate the recognition accuracy attainable by our proposed fingerprint construct. In particular, we were interested in gauging the *uniqueness* of small Patterns consisting of 3, 4, and 5 nodes to determine whether such Patterns could be used as an alternative to full minutiae templates in civilian fingerprint recognition applications. Pattern *uniqueness* was judged based on a Pattern's ability to discriminate between a genuine user's fingerprint and an impostor's fingerprint. Discrimination was considered to be successful if an *N*-node Pattern generated from fingerprint *A* was not found in fingerprint *B*, when *A* and *B* denote two different fingers. Conversely, discrimination was deemed unsuccessful if an *N*-node Pattern from fingerprint *A* was also found to exist in fingerprint *B*.

Since we were interested in gauging Pattern uniqueness based on the *existence* of a Pattern in a particular fingerprint, in this experiment we opted for the *Single-Factor Authentication* method rather than the *Two-Factor Authentication* method. This is because, as explained in Section 5.2.5, *Single-Factor Authentication* involves searching for a reference Pattern amongst all the possible Patterns in a query fingerprint, which means that, if a matching Pattern *exists* in the query fingerprint, then this method will discover it. Alternatively, *Two-Factor Authentication* relies on the user inputting the correct Pattern using the minutiae in their query fingerprint, which means that Pattern *existence* in this case is additionally dependent upon the genuine user remembering their reference Pattern or the ability of an impostor to guess the genuine user's reference Pattern and locate a matching version in their own fingerprint. While *Two-Factor Authentication* would thus make it much more difficult for an impostor to be falsely accepted as a genuine user, in this experiment we consider the worst-case scenario in which an impostor *knows* the genuine user's Pattern. Both in this case and in *Single-Factor Authentication*, where an impostor does not need to know a genuine user's reference Pattern, Pattern *uniqueness* would depend only upon the reference Pattern *existing* in the query fingerprint; therefore, Pattern *uniqueness* in both cases is covered by evaluating the recognition accuracy in the *Single-Factor Authentication* scenario.

In a nutshell, this experiment involved constructing all possible 3-node, 4-node, and 5-node Patterns from the minutiae in several reference fingerprints, and then searching for each of those reference Patterns in a number of query (test) fingerprints. Authentication was considered to be successful if at least one match for a reference *N*-node Pattern was found in

the query fingerprint. The recognition accuracy of the proposed fingerprint construct was then evaluated in terms of the resulting False Accept Rate (FAR) and the False Reject Rate (FRR). A *False Accept* occurs when the query fingerprint and the reference fingerprint come from different fingers, yet the reference *N*-node Pattern finds a match in the query fingerprint. The lower the FAR, the greater the *uniqueness* of an *N*-node Pattern, since its ability to discriminate between genuine users and impostors is greater. A *False Reject* occurs when the query fingerprint comes from the same finger as the reference fingerprint, but the reference *N*-node Pattern does not find a match in the query fingerprint. This would occur as a result of the intra-class variance between multiple samples of the same fingerprint acquired across multiple authentication attempts. So, the FAR represents the percentage of reference Patterns for which *at least one* match was found to *exist* in an impostor fingerprint sample, and the FRR indicates the percentage of reference Patterns for which *no matches* could be identified in a genuine fingerprint sample.

5.3.1.1 Selecting the Fingerprint Database

The public FVC2002 DB1_A fingerprint database was selected for our experiments. Fingerprints 1_1, 1_2, 1_6, 1_7 and 1_8 were used as the genuine samples to calculate the FRR, because those images have the most overlap in terms of capturing the same fingerprint area. Since our proposed fingerprint construct is intended for a cooperative-user civilian application, it is assumed that the users will be cooperative in placing their finger on the scanner in order to ensure that the necessary fingerprint area is captured. The first samples of the first 49 fingers (i.e., 2_1, 3_1, …, 50_1) were used as the impostor samples to calculate the FAR. Note that each of the genuine samples served as the reference fingerprint (i.e., the fingerprint against which all the other samples would be compared) in turn, which means that we effectively had a total of 20 genuine comparisons and 245 impostor comparisons *per Pattern*.

We did not use a larger number of fingerprints for two reasons: (i) setting up this experiment involved manual minutiae matching to establish ground-truth data, which is very time-consuming and would thus be impractical for a larger database at this stage, and (ii) the experiment took a considerable amount of time to run (it took about 1 week just for this small database when parts of the experiment were run in parallel, and would take over 2 weeks using sequential processing). However, the large number of Patterns tested ensures a meaningful result despite the small number of fingerprints used, as is explained at the end of Section 5.3.1.5.

5.3.1.2 Extracting the Minutiae and Core Points

The locations and orientations of the minutiae and core points were extracted using version 4.5 of the popular VeriFinger Software Development Kit [211]. The locations are expressed in terms of their (column, row) indices in the corresponding fingerprint image. The orientations lie in the range [0°, 360°) and increase in the clockwise direction in the image coordinate system.

5.3.1.3 Constructing the Reference Patterns

The reference Patterns were constructed using the minutiae from each genuine fingerprint sample (i.e., 1_1, 1_2, 1_6, 1_7 and 1_8) in turn. Since the aim of this experiment was to gauge the recognition accuracy attainable by 3-node, 4-node, and 5-node Patterns, we needed a way to evaluate the *uniqueness* of these small Patterns independently from minutiae extraction errors. Therefore, it was assumed that none of the minutiae that were used to construct the reference Patterns would be missing in the other (genuine) samples of the same fingerprint. It is possible to achieve this in practice by asking the user to scan their finger multiple times during enrolment and then using a subset of only those minutiae that appear in *all* the scans for constructing the reference Pattern. Indeed, our investigation into minutiae persistence among multiple samples of the same person's fingerprint, which is discussed in Chapter 6, will prove that this method of filtering out only the most reliable reference minutiae is quite effective at ensuring that the same minutiae will be present in a test sample of the same fingerprint. So, it is reasonable to expect that, more often than not, the reference Pattern's constituent minutiae will be present in subsequent samples of the user's fingerprint acquired during verification, especially since our Patterns use only a small number of minutiae.

The aforementioned assumption was realised by identifying all the corresponding minutiae in fingerprint samples 1_1, 1_2, 1_6, 1_7 and 1_8; there were 24. Next, the 24 minutiae from each of these fingerprint samples were ordered so that the corresponding minutiae appear at the same location in each of the ordered sets. If we let $M^j = \{m_1^j, m_2^j, \ldots, m_{24}^j\}$ denote the *ordered* set of minutiae for fingerprint sample 1_*j* (where *j* = {1, 2, 6, 7, 8}), then minutia m_i^j corresponds to minutia m_i in all the other genuine fingerprint samples. Note that the establishment of minutiae correspondences and the ordering were conducted manually to ensure that they were correct. The ordering was necessary for the threshold selection algorithm (see Section 5.3.1.4).

The 24 minutiae in each of these genuine fingerprints were then used to construct every possible 3-node, 4-node, and 5-node Pattern. Since the order of the minutiae in a Pattern is important, technically "every possible" pattern implies "every *permutation* of N out of 24 minutiae", where N refers to the number of nodes (3, 4, or 5). However, instead of *permutations*, we used every possible *combination* of N out of 24 minutiae, where an arbitrary ordering of the N minutiae in each combination of N nodes constituted one reference Pattern. This decision was primarily based on the amount of time taken to run the experiments. Tables 5.2 and 5.3 show the number of 3-node, 4-node, and 5-node Patterns obtained using permutations and combinations, respectively, and the estimated amount of time taken to sequentially calculate the FAR and FRR for each set of those Patterns using our current MATLAB implementation under the Windows 7 operating system.

Table 5.2: Total number of 3-node, 4-node, and 5-node Patterns obtained using permutations, and the estimated time to sequentially calculate the FRR and FAR for each set of Patterns.

	$N = 3$	$N = 4$	$N = 5$
Total Number of Patterns	12,144	255,024	5,100,480
Total Time to Calculate FRR	4.89 hours	5.70 days	111.45 days
Total Time to Calculate FAR	2.50 days	69.86 days	3.74 years

Table 5.3: Total number of 3-node, 4-node, and 5-node Patterns obtained using combinations, and the estimated time to sequentially calculate the FRR and FAR for each set of Patterns.

	$N = 3$	$N = 4$	$N = 5$
Total Number of Patterns	2,024	10,626	42,504
Total Time to Calculate FRR	48.91 min	5.70 hours	22.29 hours
Total Time to Calculate FAR	9.99 hours	2.91 days	11.38 days

Clearly, the use of combinations instead of permutations results in a much more practical experimental run-time. Furthermore, when the *Single-Factor Authentication Method* is used (as was the case for this experiment), if a particular reference Pattern is (not) identified in a fingerprint, then it may be correctly assumed that all the permutations of that Pattern will also (not) be present, since a different permutation is obtained by simply rearranging the order of the N constituent minutiae. Therefore, it is perfectly reasonable to use combinations instead of permutations for reference Pattern construction in this experiment.

5.3.1.4 Selecting the Thresholds for Pattern Matching

A systematic algorithm was established to select the numerous thresholds required for Pattern matching (see Section 5.2.5 for a description of the Pattern matching procedure). Each of the genuine fingerprint samples served as the reference fingerprint in turn, and each of the

reference fingerprint's Patterns was compared against its corresponding Pattern in all the remaining genuine fingerprint samples. Since the minutiae used to construct each genuine fingerprint's Patterns were in the same order (see Section 5.3.1.3), this meant that Pattern *i* from one genuine fingerprint sample directly corresponded to Pattern *i* in all the other genuine samples, because both Patterns were constructed using the corresponding sets of minutiae. Comparison of two Patterns involved comparing the corresponding attributes of their feature vectors (see Section 5.2.5). The differences between the individual attributes of every pair of feature vectors were collated into an attribute-specific histogram for a particular reference fingerprint. The idea was to find the *maximum* difference obtained for each attribute, such that, if these maximums were used to set the attribute-specific thresholds, then the FRR for that particular reference fingerprint would be 0%[19]. This process was repeated using each of the remaining genuine samples as the reference fingerprint in turn. The maximum attribute-specific differences computed for all the reference fingerprints were averaged to obtain the attribute-specific thresholds for Pattern matching. The average maximums and the thresholds for 3-node, 4-node, and 5-node Patterns are shown in Tables 5.4 and 5.5, respectively.

Table 5.4: Average maximum differences between corresponding attributes of corresponding genuine Patterns.

	$N = 3$	$N = 4$	$N = 5$
Maximum l difference	12.86	12.86	12.86
Maximum α, β difference (combined)	31.28°	31.28°	31.28°
Maximum (x, y) Euclidean distance	23.68	22.18	21.21
Maximum ω difference	27.98°	27.98°	27.98°

Table 5.5: Matching thresholds for 3-node, 4-node, and 5-node Patterns, based on the maximum values in Table 5.4.

Matching Threshold	Selected Value
τ_l	13
$\tau_{\alpha\beta}$	32°
τ_{loc}	24
τ_{ω}	28°

5.3.1.5 Calculating the FRR and FAR

The reference 3-node, 4-node, and 5-node Patterns created from the 24 minutiae in each of the 5 genuine fingerprint samples (see Section 5.3.1.3) were searched for in the remaining

[19] Since our proposed fingerprint construct is intended for deployment in a civilian fingerprint recognition application, the FRR would need to be kept as low as possible in order to make the recognition process as convenient as possible for the genuine users.

genuine and impostor fingerprint samples. If *at least one* match for a particular reference Pattern was found in a certain query fingerprint, the verification result was set to 1, and if *no matches* were found then the verification result was set to 0. Let NSG_i^j and NSI_i^j denote the Number of Successful Genuine verification attempts and the Number of Successful Impostor verification attempts, respectively, for Pattern i from reference fingerprint 1_j, where j = {1, 2, 6, 7, 8}. Further, let T denote the total number of N-node Patterns in each reference fingerprint, where N is 3, 4, and 5, in turn. Then the *average* FRR and FAR across all N-node Patterns was computed using Equations (5.23) and (5.24), respectively:

$$FRR_N = \frac{1}{5T}\sum_j\sum_{i=1}^{T}\left(4 - NSG_i^j\right) \div 4 \times 100\% \tag{5.23}$$

$$FAR_N = \frac{1}{5T}\sum_j\sum_{i=1}^{T} NSI_i^j \div 49 \times 100\% \tag{5.24}$$

5.3.1.6 Results and Discussion for Experiment 1

The FAR and FRR obtained for each Pattern size are recorded in Table 5.6.

Table 5.6: The FRRs and FARs obtained for 3-node, 4-node, and 5-node Patterns, using the matching thresholds in Table 5.5.

	FRR	FAR
$N = 3$	0.3%	6.5%
$N = 4$	0.3%	3.0%
$N = 5$	0.4%	2.1%

The FAR and FRR results in Table 5.6 are extremely encouraging, considering that this construct incorporates only a fraction of the minutiae available in a full minutiae template. The FRR was found to be 0.4% or less for all the Pattern sizes tested. The obtained FARs are highly favourable, because they suggest that, despite the small number of minutiae used, the resulting Patterns are quite *unique* in terms of providing a sufficient level of discrimination between genuine users and impostors. In particular, our FAR results indicate that, in this experiment, 93.5% of a genuine user's reference 3-node Patterns, 97% of their 4-node Patterns, and nearly 98% of their 5-node Patterns are *unique* in the sense that they *do not exist* in an impostor fingerprint.

The obtained results suggest that it is possible to achieve a recognition accuracy to suit many civilian applications by using only a small proportion of the full minutiae template. The FRR of 0.4% or less implies that genuine users are unlikely to be inconvenienced by an annoyingly high rejection rate, provided that users are cooperative and thus consistent at capturing approximately the same fingerprint area during each verification attempt. At the same time, the FARs are low enough for applications where the acceptance of a few impostors would not be catastrophic (e.g., in gyms and schools). In practice, the FAR and FRR may be optimised according to the requirements of the application in which our proposed fingerprint construct is employed.

The decreasing trend in the FAR as more nodes are added is expected, because incorporating more minutiae into the Pattern provides a higher degree of uniqueness or distinguishability, which in turn makes it more difficult for an impostor to be mistaken for a genuine user. Patterns made up of 5 minutiae were thus shown to be considerably more unique than 3-node patterns.

The performance evaluation recommends using a 5-node Pattern as the best choice out of the three Pattern sizes tested. This is because a 5-node Pattern was able to achieve the lowest FAR whilst increasing the FRR by only about 0.1% compared to 3-node and 4-node Patterns. Having made this recommendation, it is important to keep in mind that the Pattern size chosen in practice should be based on the performance and security requirements of the particular application.

Note that the fact that the FRR and FAR were calculated across a large number of patterns (see Table 5.3) provides sufficient legitimacy to the obtained results, despite the small size of the database employed. Nevertheless, it would be useful to extend this investigation to a larger fingerprint database. Recall that our proposed fingerprint construct is intended for deployment in a cooperative-user civilian fingerprint recognition application. For this reason, most public fingerprint databases, such as FVC2002 DB1_A, which were deliberately constructed to reflect uncooperative user behaviour, do not provide a suitable testing platform for fairly representing the recognition accuracy attainable by our proposed fingerprint construct. It was thus necessary for us to construct our own database of fingerprints from cooperative users, the details of which are provided in Chapter 6. Chapters 7 and 8 evaluate the performance of our proposed fingerprint construct on this cooperative-user fingerprint database.

Recall that the performance evaluation for Experiment 1 was based on the *Single-Factor Authentication* method, for which a user is required to simply present their fingerprint to the recognition system and the system then searches through the query fingerprint for an *N*-node

Pattern that matches the reference *N*-node Pattern. For the *Two-Factor Authentication* method, where a user is required to remember their reference *N*-node Pattern and then input it along with their fingerprint during authentication, we would expect the FRR to be slightly higher and the FAR to be significantly lower than for *Single-Factor Authentication*. This is because a successful authentication now depends not only on a matching *N*-node Pattern *existing* in the query fingerprint, but on the user *identifying* that matching Pattern. Since a genuine user may sometimes forget their reference *N*-node Pattern, we may expect the FRR to be slightly higher than in the scenario where the user is not required to remember their reference Pattern. The reason we say "slightly" higher is because we assume that a user authenticating themselves with their *N*-node Pattern on a regular basis is unlikely to easily forget this Pattern. However, a separate investigation must be conducted into the probability of a genuine user remembering their reference *N*-node Pattern, since this will depend on many factors, including how often the user logs into the associated application with that particular Pattern, the user's age, the complexity of the chosen Pattern, etc. Since the nature of this investigation would be largely dependent upon the nature of a particular application (e.g., expected types of users, frequency of use, etc.) and would need to be conducted over a long period of time, we leave this for future work directed at a more specific application. The reason that we would expect the FAR for *Two-Factor Authentication* to be "significantly" lower than the FAR for *Single-Factor Authentication* is because the large number of *N*-node Patterns possible from the minutiae in a single fingerprint would make it very difficult for an impostor to guess the matching one, assuming that a matching *N*-node Pattern even *exists* in the impostor's fingerprint. The FAR in the *Two-Factor Authentication* scenario is evaluated in Chapter 8 on our cooperative-user fingerprint database.

A potential limitation of our proposed fingerprint construct in terms of the recognition accuracy attainable in practice is its reliance on all *N* of the reference *N*-node Pattern's minutiae being present in every query sample of the user's fingerprint. It may be unreasonable to expect that this will be the case for every verification attempt; however, it is possible to improve the chances of a reference Pattern repeating in a query sample of the same fingerprint by asking the user to scan their finger multiple times during enrolment and then using a subset of only those minutiae that appear in *every* scan for constructing the reference Pattern. This statement is backed up by our investigation in Chapter 6, which clearly shows that combining multiple reference fingerprints during enrolment to filter out only the most reliable reference minutiae is an effective way of increasing the probability of a reference minutia repeating in future (query) samples of the same fingerprint. Since the Patterns we propose use only a small number of minutiae, it is reasonable to expect that, more often than

not, the necessary minutiae will be detected in every sample of the user's fingerprint. Chapter 7 evaluates the *true* FRR of our proposed fingerprint construct, which quantifies the probability of a reference *N*-node Pattern *not* being present in a query sample of the same fingerprint due to one or more of the Pattern's *N* constituent minutiae not having been captured in that query fingerprint.

A second limitation of our proposed fingerprint construct is that the Pattern's global attributes rely on the reliable detection of the core point in the underlying fingerprint image. If the core point cannot be detected, or is detected incorrectly, verification may fail. While we acknowledge that this limitation exists, the proposed fingerprint template protection scheme is intended for use in cooperative-user civilian authentication applications, in which case it is fair to assume that: (i) the users will be cooperative, so the core area should be captured each time, and (ii) a quality check will be performed on the acquired fingerprint images, so if a fingerprint image is of too poor a quality (in which case the core point may be detected incorrectly or not detected at all), the user would be asked to re-scan their finger. To ensure that a valid core point can be found in fingerprints from *all* pattern classes (even the Arch types), we recommend defining the core point as the ridge point with the highest curvature. This is a common definition adopted in many automated fingerprint recognition algorithms. As an alternative, Chapter 11 suggests a modified version of our proposed fingerprint construct, which does not rely on the core point at all.

5.3.2 Experiment 2: Recognition Accuracy Attainable by Full Minutiae Templates

An important aspect of any fingerprint template protection approach is that it does not significantly degrade the performance of the underlying fingerprint recognition system. Therefore, the aim of this experiment was to evaluate the performance of a standard fingerprint recognition algorithm, which utilises full, unprotected minutiae templates, so that we may gain some insight into how our proposed fingerprint template protection scheme affects that performance. Section 5.3.2.1 outlines the methodology adopted in this experiment, and Section 5.3.2.2 discusses the results.

5.3.2.1 Methodology

It must be noted that a number of different fingerprint recognition algorithms exist in practice, and the selection of a particular algorithm depends on the requirements of the underlying application. For this experiment, we chose to implement a standard fingerprint recognition algorithm, which generally works as follows:

1. Extract the minutiae and core point from each fingerprint.
2. Align each set of minutiae according to the corresponding core point. This is done by expressing the minutiae (x, y) coordinates and orientations, θ, relative to the (x, y, θ) attributes of the core point.
3. Compare two fingerprints via a point-pattern matching approach, which attempts to pair up as many minutiae in the reference fingerprint with a corresponding minutia in the query fingerprint. Two minutiae are considered to correspond if they are located within a certain Euclidean distance of each other and their orientation difference is within a particular angle threshold. Two minutiae templates are considered to match if there are at least 12 minutiae correspondences between them, a number that has traditionally been recognised as the minimum requirement for a convincing fingerprint match (e.g., see [14]).

The standard fingerprint recognition algorithm was implemented in MATLAB according to the three steps above, and the same database as that described in Section 5.3.1.1 was employed in the evaluation of its recognition accuracy. Minutiae and core points were extracted in the same way as for Experiment 1. The Euclidean distance and angle difference thresholds used for minutiae matching were set to be the same as τ_l and $\tau_{\alpha\beta}$, respectively, because these are the closest corresponding attributes.

5.3.2.2 Results and Discussion for Experiment 2

The average FRR for the implemented standard fingerprint recognition algorithm was 0% and the average FAR was 0.4%. A comparison of these results to the FRRs and FARs in Table 5.6 suggests that the proposed fingerprint construct slightly affects the recognition accuracy obtainable using full, unprotected minutiae templates (depending on the Pattern size). This is expected, since the proposed fingerprint construct uses only a small portion of the minutiae available in a full minutiae template. Nevertheless, the preliminary experiment on the performance of our proposed fingerprint construct suggests that it is still suitable for many civilian authentication applications, especially considering that the recognition accuracy *can* be improved by optimising the matching thresholds. Furthermore, the increased fingerprint template security offered by our approach, which is rigorously examined in Chapters 10 to 12, makes the trade-off in the slightly reduced recognition accuracy worthwhile.

5.3.3 Experiment 3: Verification Speed

The aim of this experiment was to estimate the amount of time it takes to verify a single person using the proposed fingerprint construct when *Single-Factor Authentication* is employed, and to ascertain whether the size of the Patterns has a significant effect on the verification speed. The motivation for this experiment was to get an idea of how convenient our proposed fingerprint construct would be in a practical scenario, in terms of approximately how long a genuine user may expect to wait to be verified during *Single-Factor Authentication*. Section 5.3.3.1 outlines the methodology adopted in this experiment, and Section 5.3.3.2 discusses the results.

5.3.3.1 Methodology

For every reference Pattern constructed in Experiment 1, MATLAB's "tic toc" function was used to measure how long it takes to find *at least one* match for that Pattern in each of the genuine fingerprint samples. The resulting times were averaged across the total number of same-size Patterns and the total number of reference fingerprint samples (five) to obtain the average time taken to verify each N-node Pattern (where $N = 3$, 4, and 5, in turn).

5.3.3.2 Results and Discussion for Experiment 3

The average time taken to find at least one match for a single 3-node, 4-node, and 5-node Pattern is reported in Table 5.7. Note that all times were rounded to 1 decimal place, because any higher accuracy would not be perceived by a human user being authenticated in a practical scenario.

Table 5.7: The average time taken to find at least one match for a single 3-node, 4-node, and 5-node Pattern, obtained using MATLAB's "tic toc" function.

	Average Time to Find a Single Pattern Match
$N = 3$	0.1 sec
$N = 4$	0.1 sec
$N = 5$	0.1 sec

The results in Table 5.7 indicate that the proposed method is sufficiently fast for authentication purposes in practice. Furthermore, it appears that increasing the number of nodes from 3 to 5 has a negligible effect on the amount of time needed to find at least one Pattern match in the genuine user's query fingerprint.

Note that, for *Two-Factor Authentication*, the total amount of time taken to verify a person would depend on how quick the user is at recalling their Pattern and how fast they are

able to 'draw' it on their query fingerprint. After the user inputs their query Pattern, the amount of time taken to establish whether or not this Pattern matches the reference Pattern should be negligible in terms of the perceived verification speed; for example, in our MATLAB implementation, it took about 0.000005 seconds to compare a single query Pattern to a single reference Pattern, for all *N*. This should be perceived as instantaneous in practice.

5.4 SUMMARY

This chapter proposed a new fingerprint construct and evaluated its potential to serve as a fingerprint template protection scheme.

The proposed construct consists of a single Pattern generated using a small subset of minutiae from the fingerprint's entire minutiae template. A Pattern consisting of *N* minutiae is referred to as an *N*-node Pattern. An *N*-node Pattern consists of a set of *local features*, which describe the Pattern's *shape*, and a set of *global features*, which denote the Pattern's location and orientation in the fingerprint relative to the fingerprint's core. The proposed fingerprint construct lends itself to two different authentication methods: *Two-Factor Authentication*, in which a user is required to present both their fingerprint and their reference *N*-node Pattern during authentication, and *Single-Factor Authentication*, in which a user is required to present only their fingerprint for authentication purposes.

Considering the suitability of our proposed fingerprint construct to serve as a fingerprint template protection scheme, the sparsity of an *N*-node Pattern was intuitively attributed to its ability to satisfy the non-invertibility, cancellability, and diversity characteristics of an ideal fingerprint template protection scheme. A preliminary investigation into the performance of our proposed fingerprint construct produced encouraging results, suggesting that, despite the sparsity of an *N*-node Pattern, it is effectively able to discriminate between genuine users and impostors. Two main limitations of the proposed fingerprint construct were noted: its reliance on all *N* of a reference *N*-node Pattern's minutiae being present in every query sample of the same fingerprint, and its reliance on the reliable detection of a fingerprint's core point. It was mentioned, however, that both limitations can generally be dealt with in practice, particularly considering the fact that our construct is intended for deployment in cooperative-user civilian fingerprint recognition applications.

Further benefits of our new fingerprint construct include the fact that it is invariant to translation and rotation of the underlying fingerprint, so matching can be done directly on the feature vectors without any pre-alignment against the query Pattern. Finally, the highly compact nature of the proposed construct means that its storage requirements are minimal,

especially when compared to the amount of memory needed to store a person's entire minutiae template.

Overall, the analysis conducted in this chapter shines a very positive light on the proposed fingerprint construct, suggesting that it has potential to be used as an effective fingerprint template protection scheme in practice. The remainder of this thesis is thus dedicated to a rigorous examination of this promising new fingerprint template protection scheme.

Chapter 6

Consistency of Cooperative Users in Scanning their Fingerprints

Chapter 5 proposed a new fingerprint construct, which entails the representation of a fingerprint by a single Pattern constructed using a small subset of minutiae from the entire minutiae template. The fact that a successful Pattern match relies on all *N* of an *N*-node Pattern's constituent minutiae being present in the query fingerprint during every authentication attempt was identified as a drawback of this fingerprint construct, due to the possibility of missing minutiae in practice. However, it was also stated that, since our proposed fingerprint construct is intended for deployment in cooperative-user civilian fingerprint recognition applications, it is reasonable to expect that the probability of one or more of a Pattern's *N* constituent minutiae missing in a query sample of the same fingerprint will be small. This is because a cooperative user may be assumed to be consistent in the way in which they place their finger on the fingerprint scanner, and this, combined with a fingerprint quality checking module, should minimise the likelihood of missing minutiae. In order to validate this assumption, a dedicated investigation into user consistency in a cooperative-user scenario was conducted. This chapter presents the methodology and results pertaining to that investigation, as well as suggesting useful applications of the findings for general fingerprint recognition applications in practice.

Note: *The material presented in this chapter has been published in [22] and [23].*

6.1 INTRODUCTION

Recall, from Chapter 1, that fingerprint matching is usually based on small ridge discontinuities called minutiae [212]. The most common minutiae types are the bifurcation and the termination (see Figure 1.2). Our new fingerprint construct, proposed in Chapter 5, is also based on these fingerprint features.

A problem commonly encountered in automated fingerprint matching is that of missing minutiae. A minutia may be considered "missing" if it is present in the reference fingerprint (i.e., the fingerprint acquired during enrolment) but its corresponding minutia cannot be found in the query fingerprint (i.e., the fingerprint presented during authentication), when both fingerprints come from the same finger. There are four main reasons why a reference minutia may be missing from the query fingerprint:

1. The part of the fingerprint in which that particular minutia exists has not been captured in the query fingerprint, so the minutia is literally not present in the query fingerprint.
2. The minutia is physically present in the query fingerprint, but the quality of this fingerprint is poorer than that of the reference fingerprint, so the minutia cannot be noticed.
3. The minutia is present in the query fingerprint and the fingerprint is of sufficiently good quality for the minutia to be noticed by a human expert, but the automated feature extractor fails to detect it.
4. The minutia is present in the query fingerprint and it has been detected by the feature extractor, but the matcher does not consider this minutia to match its corresponding reference minutia (even though the two minutiae *do* match).

The likelihood of minutiae missing due to reasons 2 to 4 can be reduced by incorporating a fingerprint quality checker during fingerprint capture in civilian fingerprint recognition applications and by improving the robustness of the feature extractor and matcher. The probability of minutiae missing due to reason 1, however, is more difficult to control, since it mainly depends on how consistent the owner of the fingerprint is in presenting that fingerprint for image capture. For example, when a person does *not* want to be recognised by a fingerprint recognition system, they may be expected to be uncooperative in presenting their finger to the fingerprint scanner. In this case, it becomes likely that the acquired fingerprint image will be partial, such that many minutiae that are present in the person's reference fingerprint will be missing from the query fingerprint. On the other hand, if a person *wants* to be recognised, it is reasonable to expect that they would be consistent in placing their finger on the fingerprint scanner, such that the probability of a minutia missing will be small. Since our proposed fingerprint construct is intended for deployment in the latter scenario, we were interested in evaluating the consistency with which a cooperative user in a civilian fingerprint recognition application may be expected to place their finger on the fingerprint scanner. This, therefore, was the aim of the investigation presented in this chapter. Note the following two points regarding this objective:

- The focus on user consistency alone was due to the fact that the effectiveness of a fingerprint recognition system primarily relies on the fingerprint information that is received at the scanner; therefore, it is reasonable to conclude that user consistency is the main factor influencing minutiae persistence across multiple authentication attempts. Furthermore, the plethora of fingerprint quality checking algorithms, minutiae extraction algorithms, and minutiae matching algorithms makes it difficult to select a single "best" algorithm on which to base our findings.
- Although user consistency may naturally be expected to be high in a cooperative-user scenario, to the best of our knowledge there does not yet exist any empirical data to validate this assumption. Our investigation fills that void by empirically quantifying user consistency in such a scenario.

The remainder of this chapter begins by justifying the construction of a new fingerprint database for this investigation, and the process of collecting fingerprint images for this database is described in detail. The database is then analysed to provide insight into the consistency with which a cooperative user of a fingerprint recognition system may be expected to place their finger on the provided fingerprint scanner, in terms of the translation and rotation of the finger on the scanner surface and the probability of capturing the same minutiae during each scan of the same finger. Following this analysis, we draw several conclusions on what the findings of this investigation mean for our proposed fingerprint construct from Chapter 5.

6.2 FINGERPRINT DATABASE CONSTRUCTION

Commonly used public fingerprint databases, such as those provided for the Fingerprint Verification Competitions (FVC) [213], have generally been constructed by asking the participants to deliberately exaggerate the inconsistency with which they place their finger on the provided fingerprint scanner, e.g., [214]. Figure 6.1 shows three samples of the same fingerprint from the FVC2002 DB1_A database: the first image was acquired when the user's finger was placed on the scanner in a cooperative manner, and the second and third images are deliberately rotated and translated samples of the same fingerprint, respectively.

The nature of these databases makes them suitable for testing fingerprint recognition algorithms designed for deployment in uncooperative user scenarios, e.g., forensics, where the latent prints are usually partial and of poor quality; border security, where a criminal may attempt to avoid being recognised as someone on a "wanted" list; etc. However, they are not

representative of fingerprint samples that would be acquired from cooperative users in civilian fingerprint authentication applications. In such applications, it is in the users' best interests to be recognised, so it is fair to assume that they would be fairly consistent in the way in which they present their fingers to the fingerprint scanner.

Figure 6.1: Three samples of the same fingerprint from FVC2002 DB1_A.

The aim of this investigation was to quantify the expected consistency of cooperative users in civilian fingerprint recognition applications. This consistency was measured in terms of the translation and rotation of a person's finger on the fingerprint scanner surface across multiple authentication attempts, and the percentage of reference minutiae that are present in a query sample of the same person's reference fingerprint. At first, the FVC2006 public fingerprint database [215], which was collected by asking the participants to place their fingers on the scanner naturally, appeared suitable for our purposes. However, the construction of this database did not involve a quality check on the acquired fingerprint images. In our investigation, a quality check was important for two reasons. Firstly, since we were interested in evaluating minutiae persistence based on user consistency *alone*, we had to eliminate the fingerprint quality factor from the database. This means that fingerprint images acquired from the same finger had to be of approximately the same quality. Secondly, our investigation targets civilian fingerprint recognition applications, which usually perform a quality check on the captured fingerprint images [11]. This helps to improve the chances of a correct authentication decision by ensuring that the acquired fingerprint images are all of a sufficiently high quality for subsequent processing. For this reason, using fingerprint images of very variable quality was irrelevant to our investigation. Hence, the FVC2006 database was an unsuitable testing platform for our purposes and it was necessary to construct our own fingerprint database. Sections 6.2.1 to 6.2.3 describe our database collection procedure in detail.

6.2.1 Scanner Specifications

The images in our fingerprint database were acquired using the Futronic FS88 optical fingerprint scanner [210]. Futronic provides a simple user interface, which shows a live video of the user's fingerprint when it is placed on the scanner's surface. Scanning is quick and

easy, producing an 8-bit grey level fingerprint image with a resolution of 320 × 480 pixels, 500dpi.

A crucial property of electronic fingerprint scanners, which sets them apart, is their underlying sensor technology. Since optical sensors are a popular choice in fingerprint scanner design [216] and since these types of scanners generally exhibit similar user interfaces, the FS88 scanner may be considered to be "typical". This means that the results of our investigation are not limited to this particular scanner.

6.2.2 Participant Selection

Our fingerprint database was constructed using fingerprints provided by volunteers. The fact that participation was voluntary was the first step in ensuring that the database would represent cooperative users. The participants consisted of adults of both genders, from diverse ethnic backgrounds and of various ages in the range [18, 60] (though the majority were young adults). In total, 100 participants were used in this study.

Note that, since the construction of a fingerprint database involves the collection of personal information from human subjects, it was necessary to obtain approval for this investigation from the University of Auckland Ethics Committee. The investigation was approved under the condition that the acquired fingerprint database would not be shared, distributed, or used for purposes outside of those pertaining to this or a similar investigation. For this reason, our cooperative-user fingerprint database is not publicly available; however, we believe that the results obtained from the associated investigation will be a useful contribution to the research community. Further details on the practical applications of our results are discussed in Section 6.3.

6.2.3 Methodology

The participants were invited to play the part of cooperative users in a fingerprint-based computer login application. They were asked to sit down at a typical computer station with the scanner positioned on the desk approximately where the computer mouse would be. Each user was free to move the scanner around and position it in whichever way was most comfortable for them (as long as it stayed flat on the desk). Users were asked to choose a finger that they would use to authenticate themselves in a fingerprint-based computer login application. The only guidance that the users received regarding the proper placement of their finger on the scanner was that the line of the first joint from the fingertip should

approximately lie on the line just below the glass platen on the fingerprint scanner, such that the maximum fingerprint area is captured (see Figure 6.2).

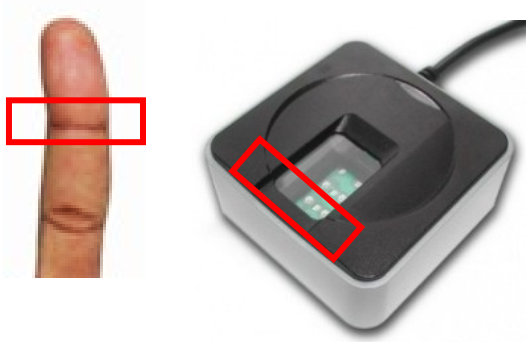


Figure 6.2: Guide on the proper placement of a finger on the Futronic FS88 scanner: the horizontal lines inside the red rectangles should approximately align.

The participants were then asked to find a comfortable position on the scanner, which they feel they could naturally repeat for future scans. Each participant's chosen fingerprint was scanned 8 times. Note that, in order to ensure that a fingerprint image was of sufficiently good quality for subsequent processing and that the quality across multiple samples of the same person's fingerprint was approximately consistent, the quality of the fingerprints was visually examined by the investigator. Users with dry skin were asked to rub their fingers on the side of their noise or onto their forehead to apply some grease to the finger, and users with excessively moist or greasy fingers were asked to dab their finger onto a piece of clothing. A fingerprint image was deemed to be of sufficiently good quality when the difference between the ridges and valleys was clear.

Note that fingerprint databases are often constructed by acquiring multiple samples of the same person's fingerprint over several days. The purpose of this is to simulate natural variability between the samples; e.g., on some days a person's finger may be drier than on other days. However, since our investigation required elimination of the quality factor, and because we assume that a cooperative-user civilian fingerprint authentication application would have an inbuilt quality checker, simulating this natural variability was unnecessary. So, we elected to collect each of a participant's 8 fingerprint samples on the same day. To simulate multiple authentication attempts, after each scan the participant was asked to remove their finger from the scanner while their previous fingerprint image was saved by a human operator. The images were saved manually to deliberately introduce some delay in between the scans and to 'distract' the participant, thereby mimicking different authentication attempts. Once the scanning started, the human operator did not guide the user in the placement of their finger on the scanner.

The participants were observed to be careful in the way in which they placed their fingers on the scanner. They also became very aware of what a good quality fingerprint image should

look like after the first quality check, and most controlled this quality on their own for subsequent scans, without prompting by the operator. These observations suggest that:

- Users are both capable and willing to be cooperative in a scenario in which they *want* to be recognised.
- Investing only a few seconds during enrolment to show people what a good quality fingerprint image should look like will help the users control this quality on their own during authentication.

6.3 ANALYSIS OF USER CONSISTENCY IN PLACING FINGER ON SCANNER

The consistency with which the participants placed their chosen finger on the scanner was analysed in terms of three factors: translation, rotation, and captured fingerprint minutiae. These factors are further described and analysed in Sections 6.3.1 to 6.3.3.

6.3.1 Translation

Translation refers to the horizontal and vertical offsets between multiple samples of the same fingerprint. A horizontal translation occurs when the user moves their finger to the left or right on the scanner surface, and a vertical translation occurs when the user moves their finger up or down. The more consistent a user is in placing their finger on the scanner, the smaller these translations will be.

To measure the translation between each person's 8 fingerprint samples, a reference point inside each fingerprint was first chosen. A reference point is a feature that is present in all 8 of a person's fingerprint samples. The most commonly used reference point in practice is the *core point*, which has traditionally been defined as the centre of the north-most loop-type pattern in a fingerprint image, or for fingerprints that do not contain loops the core usually corresponds to the point of maximum ridge line curvature [5]. We thus decided to use the core point as the common reference point between all 8 samples of each person's fingerprint.

The (x, y) location (corresponding to the (column, row) pixel indices in the fingerprint image) of the core point in every fingerprint was extracted using VeriFinger 6.7 [217]. To ensure that we were working with ground-truth data, each fingerprint was manually inspected to confirm that its core location was correctly determined. If the core in a particular fingerprint sample was detected in the wrong location, but it was correct in other samples of the same fingerprint, then those other samples were used as a guide in manually identifying the location of the core point in the former fingerprint. If the core was not detected in any

samples of the same fingerprint, which was often the case for Arch type fingerprints, then the point of highest curvature was selected as the core point. The horizontal and vertical translations between *every pair* of a person's 8 fingerprint samples were then calculated using Equations (6.1) and (6.2), respectively:

$$HT_{ij} = |x_i - x_j| \tag{6.1}$$

$$VT_{ij} = |y_i - y_j| \tag{6.2}$$

In Equations (6.1) and (6.2), HT_{ij} and VT_{ij} denote the horizontal and vertical translations (in pixels), respectively, between fingerprint samples i and j from a single person. Further, (x_i, y_i) and (x_j, y_j) represent the x- and y-coordinates of the core point in the same two sample images (i and j, respectively). The absolute value brackets in Equations (6.1) and (6.2) suggest that we are only interested in the *quantities* of the translations, rather than their specific directions. By "specific directions", we mean directions within the larger class of horizontal and vertical translations, i.e., left or right for horizontal translations, and up or down for vertical translations. The reason that we are not interested in these more specific directions is simply because they are arbitrary depending on which of a pair of sample images is chosen to be i and which is chosen to be j in Equations (6.1) and (6.2).

Equation (6.1) and Equation (6.2) were applied to our cooperative-user fingerprint database to calculate the horizontal and vertical translation, respectively, between each pair of fingerprint samples originating from the same finger. Figure 6.3 compares the box and whisker plots corresponding to the horizontal and vertical translation distributions resulting from applying Equations (6.1) and (6.2) to all 100 people in our cooperative-user fingerprint database.

In Figure 6.3, the median *horizontal* translation is 13 pixels, with an interquartile range of 17 pixels (upper quartile of 23 – lower quartile of 6) and a range of 48 pixels (upper whisker of 48 – lower whisker of 0). Similarly, Figure 6.3 indicates that the median *vertical* translation is 17 pixels, with an interquartile range of 23 pixels (upper quartile of 30 – lower quartile of 7) and a range of 64 pixels (upper whisker of 64 – lower whisker of 0).

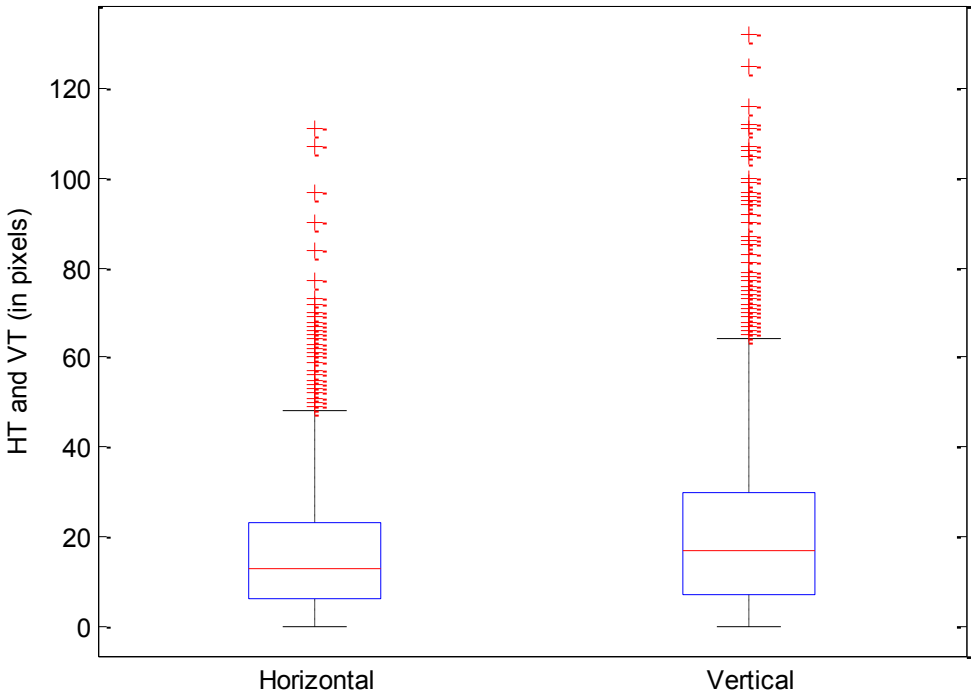


Figure 6.3: Box and whisker plots comparing the horizontal and vertical translation distributions.

The fact that the median horizontal translation is slightly smaller than the median vertical translation makes sense, because the height of the scanning surface of the Futronic FS88 scanner (which was used to acquire the fingerprint images for our database) is 1.5 times its width (this would have been a conscious design decision to approximately replicate the shape of a finger). This means that the user has more freedom to move their finger up and down than they do to move it left and right. Consequently, we would typically expect vertical translations to be larger than horizontal translations. Similarly, the fact that both the interquartile range and range of the horizontal translation distribution were found to be larger than the corresponding statistics pertaining to the vertical translation distribution suggests that there is more variability in the vertical translations between different samples of the same fingerprint than there is in the horizontal translations. So, not only are vertical translations more likely to be larger (larger median), but there is also likely to be more variation in the actual values of those vertical translations due to the greater degree of freedom in the vertical placement of the finger on the scanner. Since, for reasons outlined in Section 6.2.1, the Futronic FS88 may be considered a "typical" fingerprint scanner, we may conclude that the observations from Figure 6.3 extend beyond the Futronic FS88 scanner.

The results in Figure 6.3 provide a fair estimation of the amount of horizontal and vertical translation that we may expect, in pixels, between multiple images of the same fingerprint, when the fingerprints are captured from cooperative users. To gain a better appreciation of the significance of these translation amounts, Table 6.1 shows the pixel values of the median,

interquartile range and range of the horizontal and vertical translation distributions from Figure 6.3 in millimetres. Note that the fingerprint images in our database all measure 320 pixels in width and 480 pixels in height, and the scanning surface of the Futronic FS88 fingerprint scanner measures 16.26mm in width and 24.38mm in height. Taking these dimensions into account, Equation (6.3) was used to convert a horizontal translation from pixels to millimetres, and Equation (6.4) was used to convert a vertical translation from pixels to millimetres. Table 6.1 summarizes the results.

$$HT^{mm} = \left(HT^{pix} \div 320\right) \times 16.26 \tag{6.3}$$

$$VT^{mm} = \left(VT^{pix} \div 480\right) \times 24.38 \tag{6.4}$$

Table 6.1: Median, interquartile range and range of horizontal and vertical translation distributions in pixels and millimetres.

	Horizontal Translation		Vertical Translation	
	Pixels	**Millimetres**	**Pixels**	**Millimetres**
Median	13	0.66	17	0.86
Interquartile Range	17	0.86	23	1.17
Range	48	2.44	64	3.25

From Table 6.1, we can see that the median horizontal translation is 0.66mm, with an interquartile range of 0.86mm and a range of 2.44mm. The median vertical translation was found to be 0.86mm, with an interquartile range of 1.17mm and a range of 3.25mm. These observations may provide the reader with a better appreciation of just how consistent a cooperative user of a fingerprint recognition application may be expected to be, when the scanner's surface is designed in a similar manner to that of the Futronic FS88.

The results of this investigation could come in useful when developing fingerprint recognition applications in which user cooperation would be expected (e.g., in civilian fingerprint recognition applications, such as computer login, for which it would be in the users' best interests to be recognised). The most obvious use for these results would be in testing the suitability of the Futronic FS88 scanner, or a scanner with a similar user interface, for a particular application. Since the results indicate a high level of user consistency in terms of finger translation on the scanner surface, we may conclude that this type of scanner would be suitable for an application in which user consistency is important. It is our hope that this investigation will inspire developers of fingerprint recognition systems to conduct similar experiments when evaluating the suitability of a particular scanner for their applications, as

well as encouraging designers of fingerprint scanners to consider how these results may influence scanner surface design.

Another use for the results of this investigation would be in the development of fingerprint alignment algorithms. For example, our results in Figure 6.3 and Table 6.1 provide developers with a realistic approximation of the amount of horizontal and vertical translation that is likely to occur among cooperative users (when the Futronic FS88 or a similar scanner is employed). The results indicate that a suitable alignment algorithm must be capable of resolving translations within quite a small range, which immediately suggests that a coarse alignment algorithm may be unsuitable for this purpose. If translational offsets between two different samples of the same fingerprint are to be corrected using a common reference point, the results of our investigation could be applied in speeding up the search for a common reference point between the two fingerprints. For example, our results indicate that the locations of the core point in two different samples of the same fingerprint should not differ by more than ± 48 pixels in the horizontal direction and ± 64 pixels in the vertical direction. So, once the core point is found in the first fingerprint sample, we may use the results of our investigation to approximate the likely location of the core point in the second fingerprint sample. The core detection algorithm can then be applied to the selected area first.

6.3.2 Rotation

Rotation refers to the difference in orientation between multiple samples of the same fingerprint. The more consistent a user is in placing their finger on the scanner, the more similar the orientations of their fingerprint samples will be, and thus the smaller the rotation.

To calculate the rotation between each person's 8 fingerprint samples, it was first necessary to establish the ground truth orientation of each fingerprint. Since the orientation of a fingerprint is commonly represented by the orientation of the core point, we decided to adopt this method for calculating the ground-truth orientation of each fingerprint in our database. As for the translation calculations in Section 6.3.1, the core point from each fingerprint was extracted using VeriFinger 6.7 [217]; however, this time we were only interested in the core *angle*, rather than its *location*. In order to ensure that we were working with ground-truth fingerprint orientations, the angle of each core point was manually inspected and corrected if necessary. The rotation between *every pair* of a person's 8 fingerprint samples was then calculated using Equation (6.5), where θ_i and θ_j denote the orientations of fingerprints *i* and *j*, respectively, from the same person:

$$\phi_{ij} = \min(|\theta_i - \theta_j|, 360° - |\theta_i - \theta_j|) \tag{6.5}$$

Figure 6.4 depicts the resulting rotation distribution in terms of a box and whisker plot.

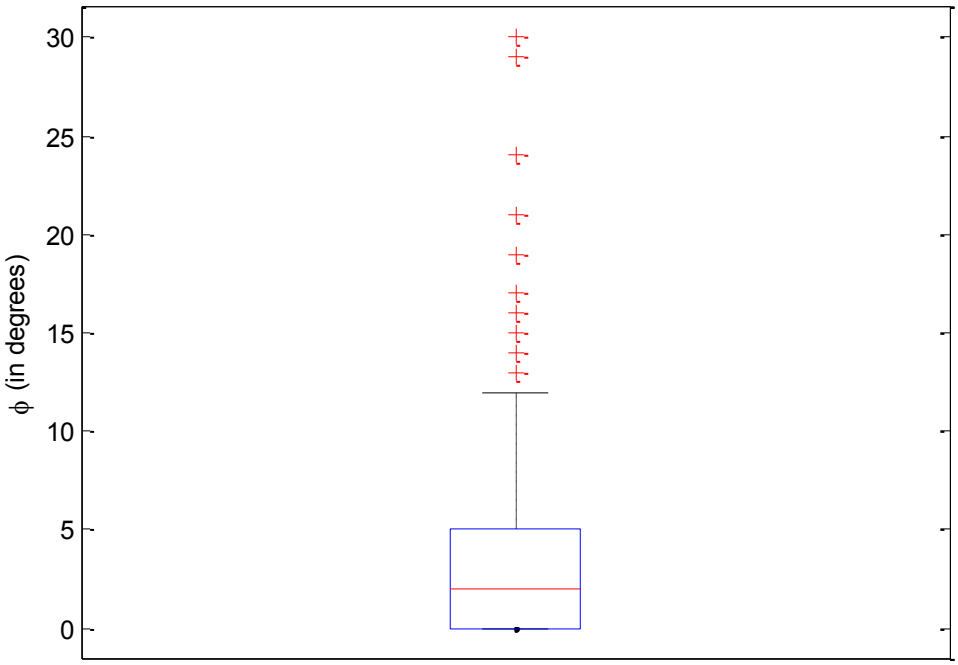


Figure 6.4: Box and whisker plot of the rotation distribution.

From Figure 6.4, we may conclude that cooperative users of a fingerprint recognition application may be expected to be very consistent in placing their finger onto the scanner, such that the median rotation between multiple samples of the same fingerprint should be 2°, with an interquartile range of 5° (upper quartile of 5° – lower quartile of 0°) and a range of 12° (upper whisker of 12° – lower whisker of 0°). Note that this consistency would also be influenced by the scanner's design. The scanning surface of the Futronic FS88, which was used for our fingerprint database collection, has been designed to approximately replicate the shape of a finger, so there is not much room for rotating the finger when it is placed on the scanning surface. We may thus assume that the results from this experiment extend to all other scanners with a similar design for the scanner surface as that of the Futronic FS88.

As for the investigation on translation in Section 6.3.1, our results from the investigation on rotation would be useful in the development of fingerprint recognition systems. For example, our results in Figure 6.4 indicate that, in order to correct rotational differences between two different samples of the same fingerprint acquired from cooperative users, the adopted alignment algorithm must be capable of resolving small rotations. This suggests that a coarse alignment algorithm may be unsuitable. Furthermore, our results indicate that it may be unnecessary to check for rotations of more than $\pm 12°$, which would be helpful in speeding up alignment methods that exhaustively check every possible rotation until the correct one is

found (e.g., [218]). Finally, as suggested in our investigation on translation in Section 6.3.1, the results of our evaluation on rotation would also be suitable for assessing the suitability of the Futronic FS88 scanner (and other scanners with a similar user interface) for a particular application, and it is hoped that both the results of this investigation and its methodology will come in useful for developers of fingerprint scanners.

6.3.3 Captured Fingerprint Minutiae

The area of the fingerprint that is captured during each scan is important for reliable fingerprint recognition. Depending on how a finger is placed on the scanner, different portions of the same fingerprint may be captured during multiple scans. The more consistent a user is in placing their finger on the scanner, the higher the probability of the same fingerprint area being captured every time. Since the ultimate point of acquiring a fingerprint image is to use it for recognition purposes, and since the most common fingerprint features used in recognition are a fingerprint's *minutiae*, the captured fingerprint area was analysed in terms of the minutiae that were present in each fingerprint image. Since the bifurcation and the termination are generally the only minutiae types considered in fingerprint recognition, and indeed in our new fingerprint construct proposed in Chapter 5, our investigation on minutiae persistence in this section considers only these two minutiae types.

Our new fingerprint database was analysed to gain insight into the expected persistence (repeatability) of reference minutiae in a cooperative-user civilian fingerprint recognition application, when that persistence depends on user consistency *alone*. This persistence was quantified in terms of the percentage of reference minutia that are physically present in a query sample of the same (reference) fingerprint.

To ensure that we were evaluating the *baseline* minutiae repeatability, based on user consistency *alone*, it was necessary to use *ground truth* minutiae information, free from the errors of automatic fingerprint feature extractors and matchers. For this reason, the minutiae from each fingerprint were extracted manually and correspondences between the minutiae in all 8 samples of each fingerprint were also established manually. All 8 samples of a person's fingerprint were scrutinised simultaneously to find matching minutiae. Once all the minutiae were thought to have been identified and matched, a final, careful check of all 8 samples was made to ensure that no minutiae were missed out. Note that minutiae identification and matching in good quality fingerprint images is fairly simple for an informed human, as people are naturally apt at pattern recognition. Since a quality check was performed during image acquisition (see Section 6.2.3), the images were of sufficiently good quality to make the process of identifying minutiae reasonably straightforward; it just took a lot of patience to

ensure that they were all found! Therefore, we may conclude that, if any human error crept into this process, it was insignificant compared to the total number of minutiae extracted for the entire database.

Reference minutiae persistence was analysed in two different scenarios: one in which the reference minutiae are extracted from a *single* reference fingerprint, and one in which *multiple* reference fingerprints are combined to filter out the reliable minutiae. Sections 6.3.3.1 and 6.3.3.2, respectively, detail the analysis in each of these scenarios.

6.3.3.1 Scenario 1: Single Reference Fingerprint

In this scenario, the reference minutiae were extracted from only one reference fingerprint and all the reference minutiae were considered reliable. For each person, all possible pairs of images from their 8 fingerprint samples were established (same as for the translation analysis in Section 6.3.1 and the rotation analysis in Section 6.3.2). In each pair, one of the fingerprints was chosen to be the reference fingerprint and the other was the query (test) fingerprint. To ensure fairness, each fingerprint in each pair had a turn at being the reference. The corresponding minutiae between the reference and test fingerprints were then established. In order to avoid faulty or missing minutiae correspondences, which may be the result of using automated minutiae matchers, minutiae correspondences were determined manually. For each reference-query fingerprint pair, the percentage of reference minutiae that were paired up with a minutia in the corresponding query fingerprint was then calculated, and this was repeated for all 100 people in our fingerprint database. Figure 6.5 depicts the resulting distribution corresponding to the percentage of reference minutiae persisting in a query sample of the same fingerprint, in terms of a box and whisker plot.

The box in the box and whisker plot from Figure 6.5 suggests that, 50% of the time, we may expect cooperative users of a fingerprint recognition application to be consistent enough in placing their finger on the fingerprint scanner to ensure that between 91.7% and 98.6% of the minutiae present in their reference fingerprint (acquired during enrolment) will be captured in their query fingerprint (acquired during authentication), with the median being 96.1%. The whiskers indicate that this percentage would be expected to lie within the range [81.4%, 100%]. Note that this range encompasses 96.8% of the entire distribution, which was used to generate the box and whisker plot in Figure 6.5; therefore, we may conclude that, 96.8% of the time, we may expect between 81.4% and 100% of the same minutiae to be captured during every authentication attempt.

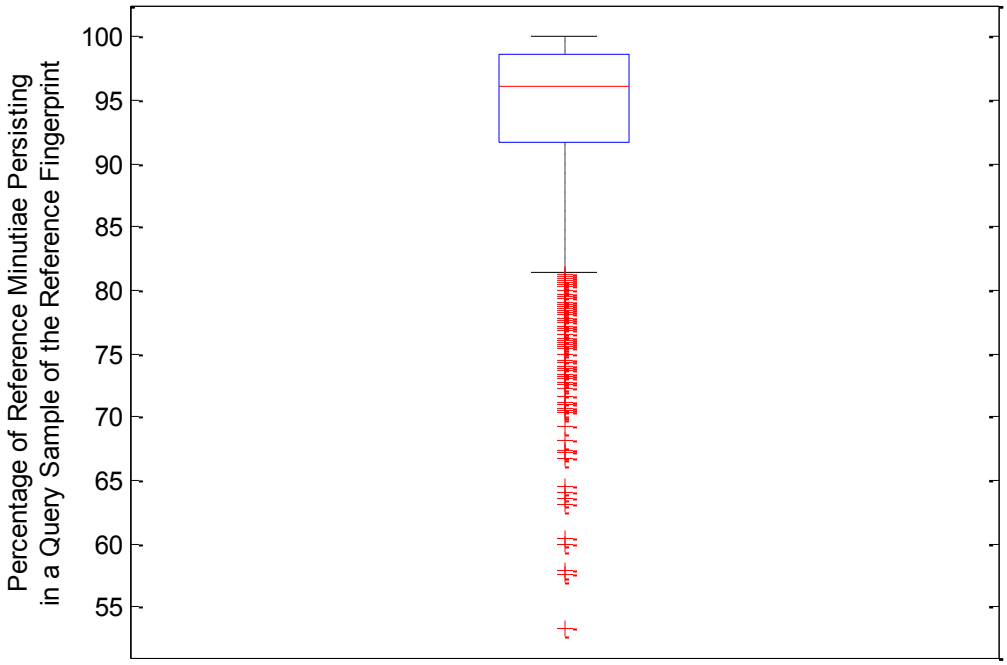


Figure 6.5: Box and whisker plot of the distribution corresponding to the percentage of reference minutiae persisting in a query sample of the reference fingerprint, when the minutiae are extracted and matched manually.

The results presented in this section would be useful for the development of automated fingerprint recognition algorithms intended for deployment in cooperative-user scenarios. For instance, knowing the percentage of reference minutiae that may be expected to occur in a query sample of the same fingerprint would help in an estimation of the likelihood that a false non-match is the user's fault (i.e., caused by user inconsistency in capturing the same minutiae). Such an analysis would be useful for honing in on the most problematic modules in a fingerprint recognition system. For example, consider the box and whisker plot in Figure 6.6, which was generated by extracting and matching the minutiae from each fingerprint in our database *automatically* (using VeriFinger 6.7 [217]) instead of *manually*. Table 6.2 compares the box and whisker plot quantities from Figure 6.5 to their corresponding statistics from Figure 6.6.

From Table 6.2, we can see that, when the minutiae are extracted and matched *automatically*, a reference minutia is less likely to appear in a query sample of the same fingerprint than when the minutiae are extracted and matched *manually*. This is because minutiae persistence in the scenario in which *manual* minutiae extraction and matching are adopted is influenced by user consistency alone; however, when the minutiae are extracted and matched *automatically*, then minutiae persistence is influenced by user consistency *and* potential minutiae extraction errors *and* potential matching errors. This hints at an interesting way in which the results of our investigation on minutiae persistence may be applied in the development and testing of automated fingerprint recognition algorithms. For example,

considering the *median* results in Table 6.2, we can see that the median percentage of reference minutiae occurring in a query sample of the same fingerprint when the minutiae are extracted and matched automatically is 9.3% lower than the median percentage obtained when the minutiae are extracted and matched manually. We may thus conclude that 9.3% of the reference minutiae are falsely not identified as being present in a query sample of the same fingerprint due to errors in the adopted automatic minutiae extractor and matcher alone. This tells us that, although minutiae persistence is most significantly influenced by user consistency, there are some errors in the automated minutiae extractor and matcher. Analysis of this sort would be useful for zoning in on the most problematic modules in a fingerprint recognition system, which would enable the development of more effective solutions.

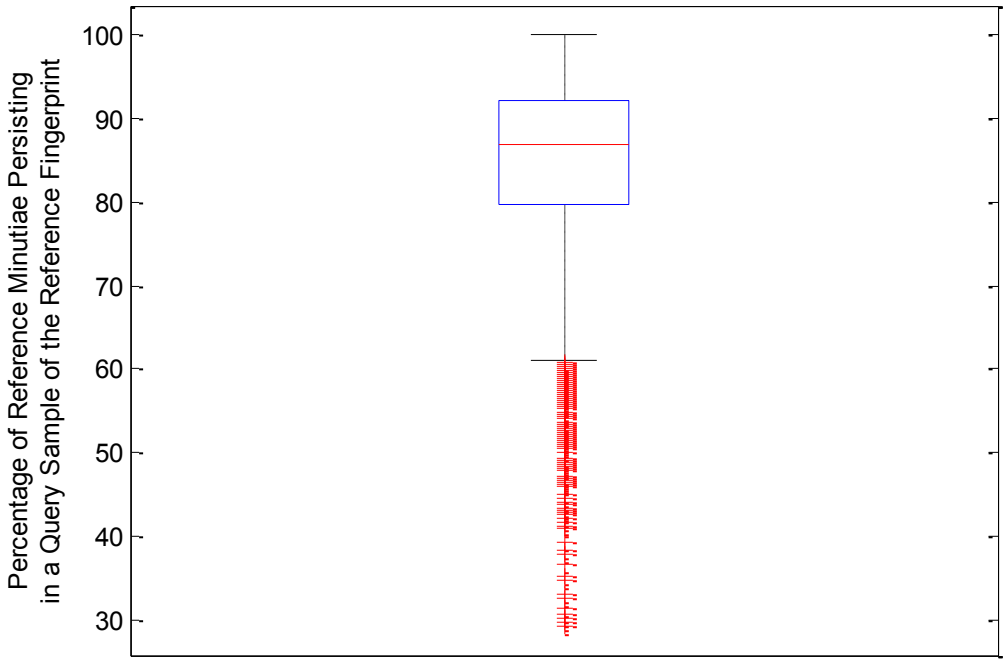


Figure 6.6: Box and whisker plot of the distribution corresponding to the percentage of reference minutiae persisting in a query sample of the reference fingerprint, when the minutiae are extracted and matched automatically.

Table 6.2: Comparison of box and whisker plot quantities for minutiae persistence when the minutiae are extracted and matched manually versus automatically.

	MANUAL Minutiae Extraction and Matching	AUTOMATIC Minutiae Extraction and Matching
Lower Whisker	81.4%	60.9%
Lower Quartile	91.7%	79.6%
Median	96.1%	86.8%
Upper Quartile	98.6%	92.2%
Upper Whisker	100%	100%

Our results on minutiae repeatability would also be useful for gauging the suitability of the Futronic FS88 scanner (and other scanners with a similar user interface) for a particular

application. Furthermore, these results may prove beneficial when applied towards the development of user-friendly fingerprint scanners, since the results indicate the ease with which the Futronic FS88 scanner (and others like it) allows its users to be consistent in capturing the same minutiae across multiple scans of the same fingerprint. A similar investigation conducted on different scanner designs would enable the respective developers to draw sensible conclusions regarding the usability of their product.

While the results in Figure 6.5 are very encouraging in terms of supporting the expected consistency of cooperative users, the percentage of reference minutiae repeating in a test sample of the same fingerprint may be further improved. For example, note that the reference minutiae that were missing from a test fingerprint were always those minutiae that were close to the edges of the reference fingerprint. This suggests that edge minutiae should not be relied upon to be present during every scan. Thus, a simple, yet effective, way of increasing the proportion of reference minutiae that are present in another scan of the same fingerprint would be to combine multiple samples of the same fingerprint during enrolment to filter out only the most "reliable" minutiae. For example, we could ask the user to scan their finger 3 times during enrolment, extract the minutiae from each of those 3 reference fingerprints and ignore any minutiae that do not appear in *all* 3 reference fingerprints. The remaining minutiae would be considered the most likely to appear in another sample of the same fingerprint, and thus only those minutiae should constitute the reference minutiae set. We would expect that, the more reference fingerprints that are employed for reference minutiae filtering during enrolment, the greater our confidence that a reference minutia will be present in a query sample of the same fingerprint. This improvement strategy is investigated in Section 6.3.3.2.

6.3.3.2 Scenario 2: Multiple Reference Fingerprints

In this scenario, instead of using only a single reference fingerprint at a time, multiple reference fingerprints were combined. The idea was to filter out only the most reliable minutiae to use as the reference minutiae. If n reference fingerprints are combined, then the most reliable minutiae are those minutiae that appear in *all* n reference fingerprints.

Logically, we would expect that using more reference fingerprints would improve the chances of a reference minutia repeating in a query sample of the same fingerprint. This is because our confidence in a reference minutia repeating in another sample of the same fingerprint grows with every reference fingerprint sample it appears in. To verify this expected trend, the number of reference fingerprints was varied from 1 to 7 for each person. If we let n denote the number of reference fingerprints used, then the percentage of reference minutia repeating a query sample of the same fingerprint was calculated for each n. Every

possible combination of n out of each person's 8 fingerprint samples was used in turn as the reference sample set, and the remaining $n - 1$ of the same person's samples served as the query fingerprints for that particular person. For each reference sample set, the reference minutiae were those minutiae that were found to be present in all n reference fingerprints (where the minutiae correspondences were established manually). The percentage of reference minutiae that were present in each of the $n - 1$ query fingerprints was then calculated (where the minutiae correspondences were also established manually). This process was repeated for each of the 100 people in our cooperative-user fingerprint database. Figure 6.7 compares the resulting distributions corresponding to the percentage of reference minutiae persisting in a query fingerprint for each n, in terms of box and whisker plots.

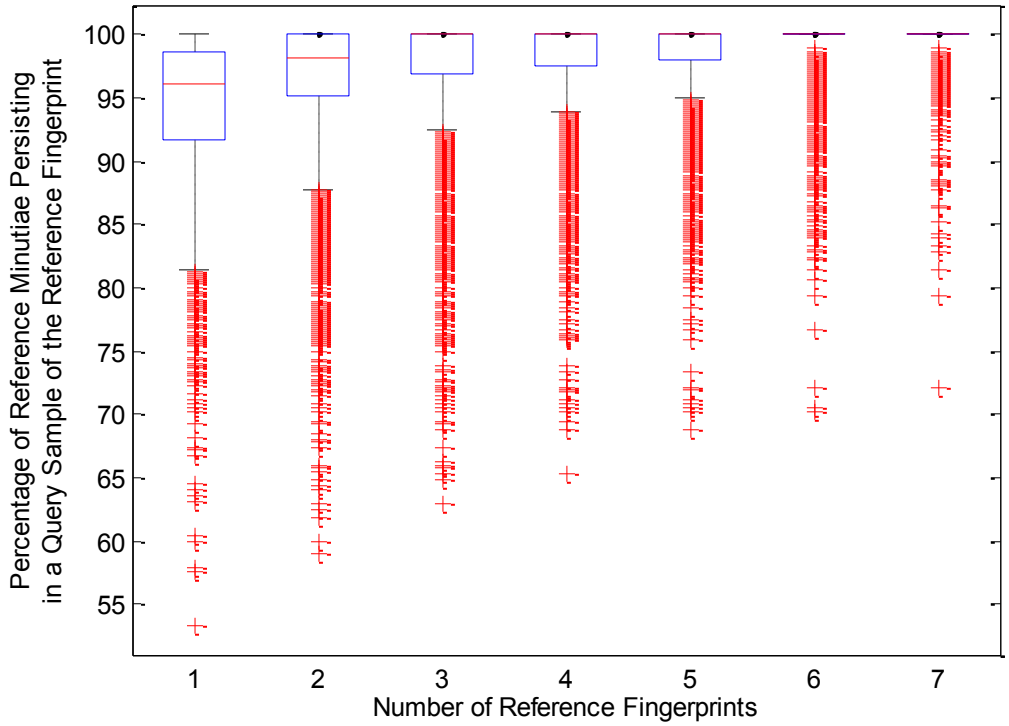


Figure 6.7: Box and whisker plots comparing the percentage of reference minutiae persisting in a query sample of the same fingerprint as the number of reference fingerprints increases.

The results in Figure 6.7 are extremely encouraging, because they suggest that it is possible to improve the probability of a reference minutia repeating in a query sample of the reference fingerprint simply by using more reference fingerprints to filter out only the most reliable reference minutiae. However, we must also consider the effect that this improvement strategy has on the total number of reference minutiae remaining for fingerprint matching purposes. Since using more reference fingerprints effectively gets rid of more (unreliable) minutiae, it makes sense to conclude that this filtering operation will result in fewer reference minutiae remaining. To show what happens to the total number of reference minutiae as the number of

reference fingerprints increases, Figure 6.8 compares the median numbers of reference minutiae, which were used to generate Figure 6.7, as *n* increases from 1 to 7.

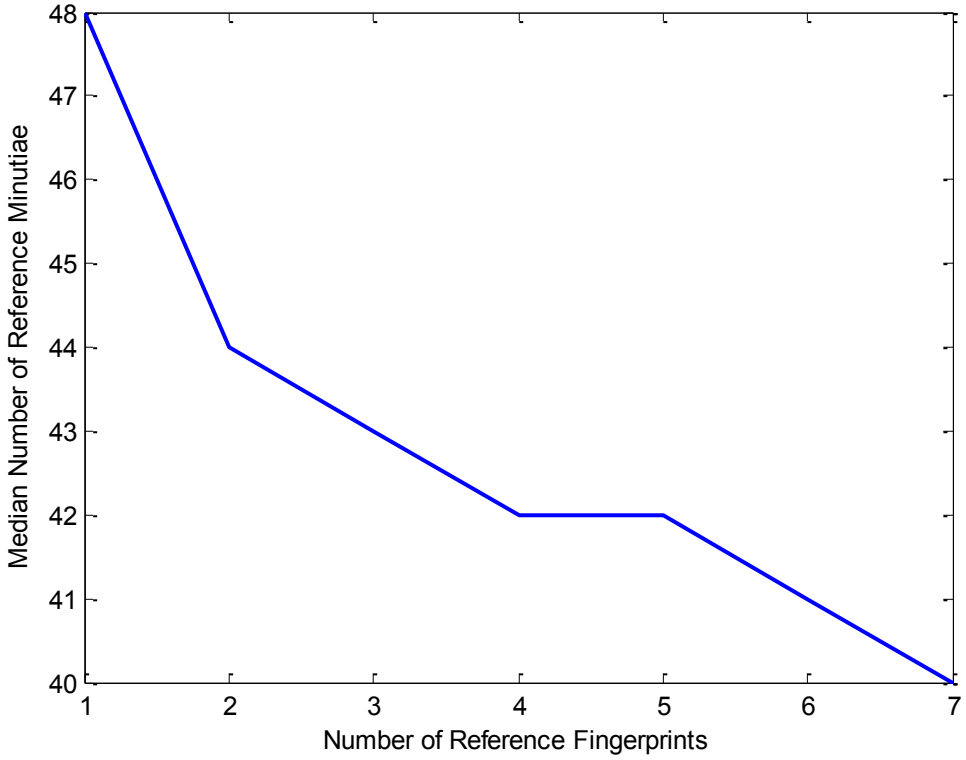


Figure 6.8: Plot showing the trend in the median number of reference minutiae remaining for fingerprint recognition purposes as the number of reference fingerprints increases.

The trend in Figure 6.8 indicates that, as the number of reference fingerprints increases, the number of reference minutiae decreases. This is because the idea behind using multiple reference fingerprints is to filter out only the most reliable reference minutiae. The most reliable reference minutiae are those minutiae that are present in *all* the reference fingerprints. So, the more reference fingerprints that are used, the less probable it becomes that a minutia will be present in *all* these fingerprints. Consequently, increasing the number of reference fingerprints has the effect of removing a larger number of (unreliable) reference minutiae.

In Figure 6.8, the median number of reference minutiae decreases from 48 when 1 reference fingerprint is used to 40 when 7 reference fingerprints are used. This is not a significant difference, which may be attributed to the fact that the participants in our database construction were very consistent in placing their fingers onto the fingerprint scanner. The more consistent a user is in placing their finger onto a scanner, the more similar multiple samples of their same fingerprint will be. Consequently, most of the minutiae should be the same across all of their fingerprint samples. This means that combining multiple samples of the reference fingerprint to filter out only the most reliable minutiae should not result in the loss of many minutiae, as is shown in Figure 6.8. Note that, traditionally, 12 matching minutiae have been considered sufficient evidence for a positive fingerprint match (e.g., see

[14]), and the preliminary investigation on the recognition accuracy our new fingerprint construct showed that an *N*-node Pattern consisting of 3, 4, or 5 minutiae is capable of discriminating between genuine users and impostors. This means that even the minimum of 40 reference minutiae in Figure 6.8 would provide ample opportunity for reliable fingerprint recognition; therefore, we may conclude that using a larger number of reference fingerprints (i.e., closer to 7) would ensure a higher probability of a reference minutia repeating in another sample of the same fingerprint than would the user of a smaller number of reference fingerprints (i.e., closer to 1), whilst maintaining satisfactory recognition accuracy.

6.4 IMOPRTANCE OF INVESTIGATION FOR THE PROPOSED FINGERPRINT CONSTRUCT

The findings of our investigation on user consistency provide meaningful insight into the practicality of our new fingerprint construct, which was proposed in Chapter 5. In particular, our results indicate that, in a cooperative-user scenario, which is the target application scenario of our new fingerprint construct, users may be expected to be consistent enough to ensure that a median of over 96% of the reference minutiae are captured in a query sample of the same fingerprint. Furthermore, it was shown that using 3 or more reference fingerprints during enrolment to filter out only the most reliable set of reference minutiae may be expected to result in a median of 100% of the reference minutiae being present in a query sample of the same fingerprint, with the interquartile range and range decreasing as the number of reference fingerprints increases. These findings indicate that, as far as user consistency is concerned, our proposed fingerprint construct should infrequently suffer from missing minutiae in practice. This point is especially important considering that our *N*-node Patterns use only 3, 4, or 5 minutiae, which means that the chances of one or more of those minutiae not being captured in the query fingerprint should be very small. A dedicated investigation into the probability of one or more of a reference *N*-node Pattern's *N* minutiae missing in a query sample of the reference fingerprint was conducted, and this forms the subject of Chapter 7.

6.5 SUMMARY

This chapter discussed an investigation into the consistency with which users in a cooperative-user civilian fingerprint recognition application may be expected to place their fingers on the fingerprint scanner during fingerprint image acquisition.

Due to the unavailability of suitable public fingerprint databases, it was necessary for us to construct our own cooperative-user fingerprint database for this investigation. The resulting database consists of 8 different samples of the same fingerprint acquired from 100 volunteers. This database will be used in Chapters 7 and 8 to evaluate the performance of our new fingerprint construct, proposed in Chapter 5, in its target application scenario (i.e., cooperative-user civilian fingerprint recognition applications).

In this investigation, we analysed our cooperative-user fingerprint database to empirically quantify the consistency of the database participants in placing their fingers on the fingerprint scanner. This consistency was evaluated in terms of the translation and rotation of a person's finger on the scanner between different scans, as well as in terms of the percentage of minutiae that were consistently captured across multiple samples of the same fingerprint. The median horizontal and vertical translations were found to be 13 pixels (0.66mm) and 17 pixels (0.86mm), respectively, the median rotation was 2°, and a median of 96.1% of the minutiae in a reference fingerprint were found to have also been captured in the query fingerprint. Using multiple reference fingerprints during enrolment to filter out only the most reliable reference minutiae was shown to be an effective mechanism for improving minutiae persistence to a median of 100%. This improvement strategy was found to have an insignificant effect on the number of reference minutiae remaining for recognition purposes. A complementary study showed that automated minutiae extractors and matchers may decrease the probability of a reference minutia being identified in a query fingerprint, and this comparison indicated a potential use of the results of our investigation in the testing of automated fingerprint recognition algorithms. Other example applications of the results of our investigation on user consistency include fingerprint scanner design and testing, and the development or informed selection of fingerprint alignment algorithms.

The insights gained from our investigation on user consistency in a cooperative-user scenario can be used to gauge the practicality of our new fingerprint construct, which was proposed in Chapter 5. In particular, our results on minutiae persistence among multiple samples of the same person's fingerprint indicate that, as far as user consistency is concerned, our proposed fingerprint construct should infrequently suffer from the problem of missing minutiae in practice. This should be the case especially if multiple reference fingerprints are used during enrolment to establish only the most reliable reference minutiae to use in reference Pattern construction.

Chapter 7

True FRR of New Fingerprint Construct

In Chapter 5, it was mentioned that a possible drawback of our proposed fingerprint construct is its reliance on the *N* minutiae in a user's reference *N*-node Pattern being present in each subsequent query sample of the same fingerprint: if one or more of these *N* minutiae is missing from the query fingerprint, authentication will fail. Our investigation into minutiae persistence in Chapter 6 showed that the probability of a reference minutia missing from a query sample of the reference fingerprint may be expected to be very low in a cooperative-user scenario. This chapter investigates the likelihood of a failed authentication attempt as a result of one or more of a reference *N*-node Pattern's *N* constituent minutiae *physically missing* in the query fingerprint. Since our proposed fingerprint construct is intended for deployment in a cooperative-user scenario, this investigation is based on our cooperative-user fingerprint database from Chapter 6.

7.1 INTRODUCTION

In Chapter 6, it was mentioned that a reference minutia may be *missing* in the query fingerprint for a number of reasons, including user inconsistency in capturing the same fingerprint area, differences in the quality of the reference and query fingerprint samples, and minutiae extraction and matching errors. While advances in fingerprint scanning technology and image processing algorithms make it increasingly easier to deal with image quality issues, feature extraction and matching errors, these technological enhancements can only do so much towards ensuring that minutiae will not be missing across multiple samples of the same fingerprint. The largest responsibility lies with the user being consistent in scanning their finger, such that the same fingerprint area is captured during every authentication attempt. For example, if the user scans a different part of their fingerprint each time, then, regardless of how good the subsequent modules in the recognition system are, we cannot expect them to be able to 'make up' information that the user has failed to provide. For this reason, the

investigation in this chapter considers missing minutiae as a result of user inconsistency alone. This decision was also based on the fact there are many different algorithms available for each of the automated processes in a fingerprint recognition system, so it is impossible to universally evaluate the likelihood of a missing minutia without biasing the estimation towards specific minutiae extraction and matching algorithms.

This chapter thus presents a study of the *true* False Reject Rate (FRR) of our proposed fingerprint construct. A *true* False Reject is caused *only* by one or more of a reference *N*-node Pattern's *N* constituent minutiae *physically missing* from the query fingerprint, such that the reference Pattern is literally *not present* in the query fingerprint. In other words, this chapter evaluates the FRR of our proposed fingerprint construct as a result of user inconsistency alone and in the absence of automated minutiae extraction and Pattern matching errors, which means that Pattern matching thresholds (see Section 5.2.5) are not considered in this investigation. To enable the evaluation of the true FRR, all the minutiae in our cooperative-user fingerprint database (see Chapter 6) were thus identified *manually*, and minutiae correspondences between multiple samples of the same fingerprint (to determine whether or not a minutia is missing) were also established *manually*.

The remainder of this chapter begins with an evaluation of the *true* FRR as the number of reference fingerprints (which are used during enrolment to establish only the most reliable reference minutiae to use in the construction of a user's reference *N*-node Pattern) increases. We then present an investigation into how a genuine user's chances for a successful authentication can be further improved by allowing them more than one authentication attempt. The most favourable number of reference fingerprints and authentication attempts, which ensures that a genuine user has the highest chance of being successfully authenticated, are established.

7.2 TRUE FRR AS THE NUMBER OF REFERENCE FINGERPRINTS INCREASES

Chapter 6 discussed our investigation into empirically evaluating the persistence of minutiae across multiple samples of the same person's fingerprint, when that person is a cooperative user in a civilian fingerprint recognition application. The corresponding analysis showed that the probability of a minutia being present in a query sample of the reference fingerprint increases as more reference fingerprints are used during enrolment to establish the most reliable reference minutiae. In light of this analysis, we may expect the persistence of an *N*-node Pattern to follow the same trend, which means that we may expect the FRR of our

proposed fingerprint construct to decrease as the number of reference fingerprints used during enrolment increases. This section presents an empirical evaluation of the *true* FRR for 3-, 4-, and 5-node Patterns extracted from our cooperative-user fingerprint database, when Pattern persistence depends on user consistency alone.

To evaluate the *true* FRR of our proposed fingerprint construct in this scenario, the following experiment was conducted:

1. For each person in our cooperative user fingerprint database, 7 reference minutiae sets were established by varying the number of reference fingerprints from 1 to 7. For n reference fingerprints, we used the first n of each person's 8 fingerprint samples as the reference fingerprints. Then, only those minutiae which appear in all of the first n of a person's fingerprint samples became part of the corresponding reference minutiae set.
2. From each of a person's 7 reference minutiae sets, 100 N-minutiae combinations were randomly selected[20]. Each N-minutiae combination was then searched for in the same person's 8th fingerprint sample, which served as that person's query fingerprint. We did not use a larger number of query fingerprint samples because only one fingerprint remains to be used as the query sample when 7 reference fingerprints are employed; so, to ensure fairness in the comparison, we used the same number of query samples regardless of how many reference fingerprints were used.
3. The number of reference N-minutiae combinations *not* found in the query fingerprint was counted, and this number corresponded to the number of False Rejects. Equation (7.1) was then used to calculate the *true* False Reject Rate for $N = \{3, 4, 5\}$ and $n = 1{:}7$ in turn.

$$FRR_n^N = \frac{FR_n^N}{10{,}000} \times 100\% \tag{7.1}$$

Note that, in Equation (7.1), FR_n^N denotes the total number of False Rejects obtained for all people when n reference fingerprints are used and when each Pattern consists of N minutiae. FR_n^N is divided by 10,000 because this corresponds to the total number of genuine authentication attempts (i.e., 100 people × 100 reference Patterns per person × 1 query fingerprint).

[20] These N-minutiae combinations represented N-node Patterns. The reason we used N-minutiae *combinations*, instead of N-minutiae *permutations*, is because any N-node Pattern generated from the *same* N-minutiae combination would give the same result in terms of the true FRR. For example, if a particular set of N minutiae is present in a query fingerprint, then *every* N-node Pattern possible from that N-minutiae combination would also be present; therefore, there is no point in using N-minutiae permutations for this experiment.

4. Steps 2 and 3 were repeated 10 times, and the average FRR_n^N across all 10 trials was calculated for each N and n in turn.

Figure 7.1 is a plot depicting the resulting *true* FRR for each $N = \{3, 4, 5\}$ as the number of reference fingerprints increases from 1 to 7.

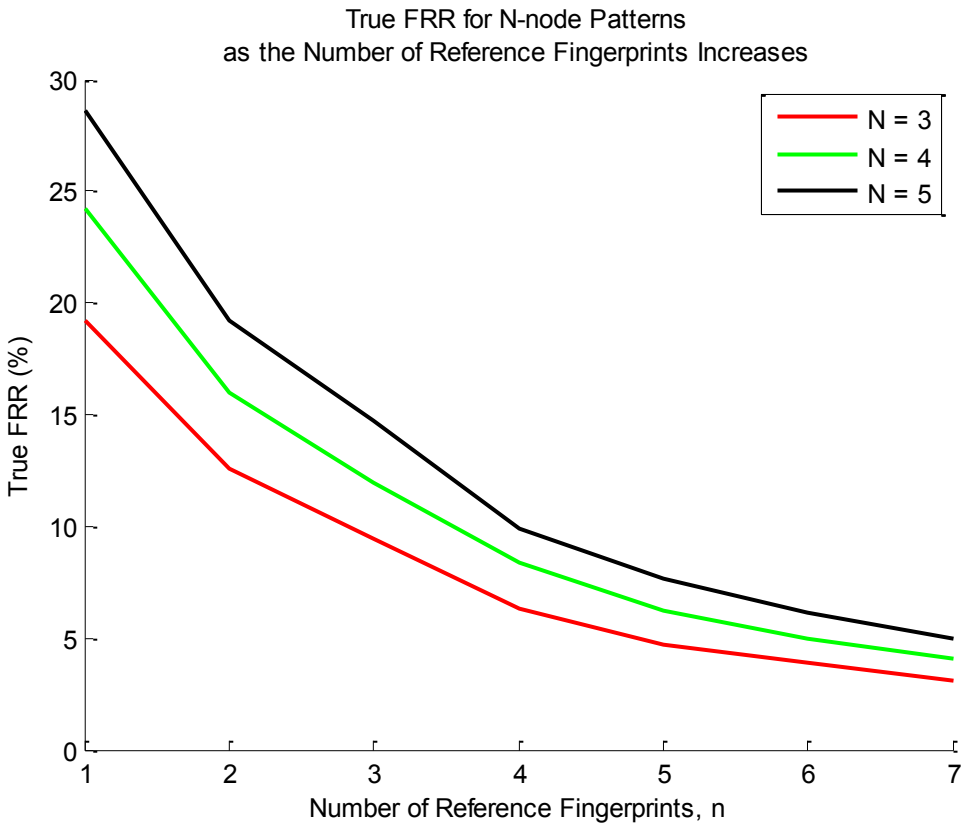


Figure 7.1: Plot showing the trend in the true FRR for *N*-node Patterns as the number of reference fingerprints increases from 1 to 7.

From Figure 7.1, it is clear that the true FRR for all Pattern sizes decreases with an increase in the number of reference fingerprints. This makes sense, because using a larger number of reference fingerprints has the effect of filtering out a larger number of unreliable reference minutiae, as shown in Chapter 6; consequently, it becomes increasingly likely that a subset of N of the resulting reference minutiae will be present in a query sample of the same fingerprint. The lowest true FRR in Figure 7.1 is thus observed at $n = 7$, at which point the true FRR for 3-node Patterns was found to be 3.07%, the true FRR for 4-node Patterns was found to be 4.03%, and the true FRR for 5-node Patterns was found to be 4.98%. The fact that the true FRR increases with an increase in the Pattern size is expected, because it is more likely that a smaller subset of N minutiae will be found in a query sample of the same fingerprint than a larger subset of N minutiae.

Note that the analysis of the *true* FRR in this section is essentially based on the assumption that the user of a fingerprint recognition system employing our proposed fingerprint construct would be given only one authentication attempt. So, the results obtained for the true FRR at $n = 7$, for example, indicate that, during a genuine user's *first* (and only) authentication attempt:

- There is approximately a 97% chance that their reference 3-node Pattern will physically exist in the query sample of their fingerprint.
- There is approximately a 96% chance that their reference 4-node Pattern will physically exist in the query sample of their fingerprint.
- There is approximately a 95% chance that their reference 5-node Pattern will physically exist in the query sample of their fingerprint.

While these figures may be considered acceptable in terms of providing a good level of convenience for the genuine user, the true FRR may be further improved by allowing the user to have multiple authentication attempts in practice. Section 7.3 investigates the true FRR in this more realistic scenario.

7.3 EVALULATION OF THE TRUE FRR WHEN THE USER IS ALLOWED MULTIPLE AUTHENTICATION ATTEMPTS

The analysis in Section 7.2 showed that the true FRR of our proposed fingerprint can be dramatically improved by combining multiple reference fingerprints to establish only the most reliable reference minutiae to use in the construction of a user's reference *N*-node Pattern. However, the analysis was based on the assumption that a genuine user would be given only *one* chance to authenticate themselves, which would not be the case in a practical implementation of our proposed fingerprint construct. As with any authentication mechanism targeted at a civilian application, user convenience would be a priority, which means that a user of our proposed fingerprint construct would, in practice, be given multiple chances for a successful authentication. This is a prudent implementation step, which would help deal with user inconsistency in placing their finger on the scanner (relevant for both *Single-Factor Authentication* and *Two-Factor Authentication*) or with the user entering the wrong *N*-node Pattern by mistake (relevant only for *Two-Factor Authentication*).

This section evaluates the true FRR of our proposed fingerprint construct when a user is allowed multiple authentication attempts. Recall that the *true* FRR refers to the probability of rejecting a genuine user as a result of one or more of the *N* minutiae in their reference *N*-node Pattern *physically missing* from the query fingerprint sample. Therefore, at this stage, we are

not interested in the number of attempts that a person would need to remember what reference *N*-node Pattern they enrolled with (when the *Two-Factor Authentication* method is adopted). Rather, we are simply interested in how many times a user must present their finger to the fingerprint scanner to ensure that the *N* minutiae making up their reference Pattern are physically captured in the query fingerprint sample. Consequently, in this investigation, each *authentication attempt* refers to the presentation of a *different* sample of the query fingerprint to the recognition system.

Instead of arbitrarily assigning a certain maximum allowed number of authentication attempts, we decided to establish this number empirically. In particular, we were interested in determining the *most favourable* balance between the number of reference fingerprints and the maximum number of authentication attempts, such that a genuine user would have the greatest possible chance of being successfully authenticated. Note that we could only determine this most favourable balance for our particular fingerprint database, so we cannot guarantee that this will be the optimum for *any* fingerprint database. However, the same method can be applied to any fingerprint database to establish its particular optimum.

Since our fingerprint database has no more than 8 fingerprint samples per person, we had to work within those constraints to establish the most favourable balance between the number of reference fingerprints and the maximum number of allowed authentication attempts. For example, if we chose to use n of a person's 8 fingerprint samples as their reference fingerprints, this left $8 - n$ of their fingerprint samples to be used as the query fingerprints, which directly corresponded to the maximum number of allowed authentication attempts.

To conduct this investigation, we essentially repeated the experiment from Section 7.2, except that, this time, each reference *N*-minutiae combination was searched for in m query fingerprints. If n reference fingerprints were used to establish the reference minutiae set, then the maximum number of authentication attempts (and thus the number of query fingerprints in which each reference *N*-node Pattern was searched for), $m = 8 - n$. This was repeated for all combinations of n and m as n varied from 1 to 7, and the true FRR in each scenario was computed. Note that the FRR was calculated using Equation (7.1), except that, this time, an *N*-minutiae combination was only considered *missing* if it could not be found in *any* of the m query fingerprints. Tables 7.1 to 7.3 depict the resulting true FRR for $N = 3$, $N = 4$, and $N = 5$, respectively.

From Tables 7.1, 7.2, and 7.3, it is clear that, regardless of whether a user enrols into the recognition system using a 3-node Pattern, a 4-node Pattern, or a 5-node Pattern, they have the best chance of being successfully authenticated if 5 samples of their fingerprint are used during enrolment to establish the reference minutiae set and the user is given a maximum of 3

authentication attempts. We shall, therefore, henceforth refer to this scenario as the *Most Favourable Genuine User Authentication Scenario* (MFGUAS).

Table 7.1: True FRR for 3-node Patterns when different numbers of reference fingerprints (*n*) are used and when the user is allowed different numbers of authentication attempts (*m*).

	True FRR for 3-node Patterns (%)						
	***n* = 1**	***n* = 2**	***n* = 3**	***n* = 4**	***n* = 5**	***n* = 6**	***n* = 7**
***m* = 1**	19.24	12.58	9.43	6.32	4.73	3.88	3.07
***m* = 2**	11.80	6.02	3.71	1.89	1.03	0.83	
***m* = 3**	9.41	4.95	2.51	1.04	0.16		
***m* = 4**	8.70	4.43	2.18	0.86			
***m* = 5**	7.14	3.23	1.41				
***m* = 6**	5.39	1.94					
***m* = 7**	3.46						

Table 7.2: True FRR for 4-node Patterns when different numbers of reference fingerprints (*n*) are used and when the user is allowed different numbers of authentication attempts (*m*).

	True FRR for 4-node Patterns (%)						
	***n* = 1**	***n* = 2**	***n* = 3**	***n* = 4**	***n* = 5**	***n* = 6**	***n* = 7**
***m* = 1**	24.19	15.96	11.99	8.41	6.25	4.98	4.03
***m* = 2**	15.18	7.99	4.80	2.44	1.28	1.08	
***m* = 3**	12.21	6.27	3.14	1.36	0.20		
***m* = 4**	11.11	5.73	2.76	1.14			
***m* = 5**	9.19	4.36	1.76				
***m* = 6**	6.74	2.58					
***m* = 7**	4.55						

Table 7.3: True FRR for 5-node Patterns when different numbers of reference fingerprints (*n*) are used and when the user is allowed different numbers of authentication attempts (*m*).

	True FRR for 5-node Patterns (%)						
	***n* = 1**	***n* = 2**	***n* = 3**	***n* = 4**	***n* = 5**	***n* = 6**	***n* = 7**
***m* = 1**	28.58	19.19	14.70	9.94	7.64	6.17	4.98
***m* = 2**	18.15	9.66	5.82	3.12	1.62	1.39	
***m* = 3**	14.54	7.63	3.93	1.67	0.24		
***m* = 4**	13.27	7.15	3.45	1.44			
***m* = 5**	11.13	5.17	2.15				
***m* = 6**	8.39	3.15					
***m* = 7**	5.72						

Our results in Tables 7.1, 7.2, and 7.3, respectively, indicate that, for our cooperative user fingerprint database, the true FRR in the *Most Favourable Genuine User Authentication Scenario* (MFGUAS) is 0.16% for 3-node Patterns, 0.20% for 4-node Patterns, and 0.24% for 5-node Patterns. These figures are extremely encouraging, suggesting that, in the MFGUAS, we may expect that:

- A user's reference 3-node Pattern will be physically present in their query fingerprint during at least one of their three allowed authentication attempts 99.84% of the time.
- A user's reference 4-node Pattern will be physically present in their query fingerprint during at least one of their three allowed authentication attempts 99.80% of the time.
- A user's reference 5-node Pattern will be physically present in their query fingerprint during at least one of their three allowed authentication attempts 99.76% of the time.

Note that the fact that the true FRR increases as the Pattern size increases is expected, since the likelihood of finding a smaller subset of minutiae in every sample of the same fingerprint is greater than the likelihood of finding a larger subset of minutiae.

7.4 SUMMARY

This chapter investigated the true FRR achievable by our proposed fingerprint construct in its target application scenario (i.e., a cooperative-user civilian fingerprint recognition application). A *true* False Reject occurs when a reference *N*-node Pattern cannot be found in a query sample of the same fingerprint only because one or more of the Pattern's *N* constituent minutiae are *physically missing* from the query fingerprint sample.

It was demonstrated that the true FRR of our proposed fingerprint construct can be dramatically improved by using multiple reference fingerprints during enrolment to establish only the most reliable reference minutiae set for the construction of a person's reference *N*-node Pattern. The more reference fingerprints that are used, the greater the probability of a genuine user being successfully authenticated and thus the lower the true FRR.

It was also shown that the true FRR of our proposed fingerprint construct can be further improved by allowing a user to have multiple authentication attempts. In particular, we found that the lowest true FRR is achievable when 5 samples of a user's reference fingerprint are employed during enrolment to establish their reference minutiae set and the user is given a maximum of 3 authentication attempts. This scenario was referred to as the *Most Favourable Genuine User Authentication Scenario* (MFGUAS). The true FRR in the MFGUAS was found to be 0.16% for 3-node Patterns, 0.20% for 4-node Patterns, and 0.24% for 5-node

Patterns. These findings indicate that, as far as user consistency is concerned, our proposed fingerprint construct should seldom be affected by missing minutiae in practice, especially if it operates in the established *Most Favourable Genuine User Authentication Scenario*.

Chapter 8

Performance of Proposed Fingerprint Construct on Cooperative-User Fingerprint Database

The fingerprint construct proposed in Chapter 5 has been designed with the aim of deploying it in civilian fingerprint recognition applications, in which it is assumed that the users would *want* to be recognised and would thus be cooperative. Preliminary experimentation on this fingerprint construct in Chapter 5 was promising, suggesting that the method would be suitable for deployment in this target scenario. Subsequent analysis in Chapter 7 showed that, for cooperative users of a fingerprint recognition system, the true FRR of the proposed fingerprint construct is dramatically improved when multiple reference fingerprints are employed during enrolment to establish the most reliable reference minutiae set. In this chapter, we investigate the effect that this FRR improvement strategy has on the resulting FAR. Analysis on the true FRR of our proposed fingerprint construct in Chapter 7 also showed that, for our cooperative-user fingerprint database, a genuine user has the best chance of a successful authentication when five samples of their fingerprint are used during enrolment to establish the reference minutiae set and the user is given a maximum of three authentication attempts. This was referred to as the *Most Favourable Genuine User Authentication Scenario* (MFGUAS). This chapter investigates the trade-offs between the FAR and FRR in a practical implementation of our proposed fingerprint construct on our cooperative-user fingerprint database in the MFGUAS. Finally, a modification to our fingerprint construct is proposed in order to further improve Pattern uniqueness.

8.1 INTRODUCTION

The experiments discussed in this chapter are an extension of the preliminary investigation on the recognition accuracy of our proposed fingerprint construct in Chapter 5. The experiments in this chapter differ from those in Chapter 5 in several ways.

Firstly, the experiments were conducted on a new database – our cooperative-user fingerprint database from Chapter 6 – which was specifically constructed to reflect the types of fingerprint images that would be acquired from cooperative users in a civilian fingerprint recognition application. Since the proposed fingerprint construct is targeted at this sort of application, the cooperative-user fingerprint database makes a suitable testing platform for evaluating the method's attainable recognition accuracy in such a scenario.

Secondly, the experiments in Chapter 5 were based on the assumption that the minutiae used in the construction of a person's reference *N*-node Pattern would *always* be present in a query sample of the same fingerprint. This assumption was reasonable in the context of our preliminary investigation, the focus of which was to evaluate the performance of our fingerprint construct in terms of the *uniqueness* of an *N*-node Pattern; so, it was important to eliminate the issue of missing minutiae. In the experiments described in the current chapter, however, this assumption has been relaxed. In other words, it is no longer assumed that the minutiae used for reference Pattern construction will *always* be present in a query sample of the same fingerprint. This assumption was relaxed in order to investigate the recognition accuracy of our proposed fingerprint construct in the face of the possibility of missing minutiae in practice.

Thirdly, the threshold selection algorithm employed in the experiments in Chapter 5 was based on the assumption that minutiae matching would be 'perfect', i.e., as good as if it were conducted by a human expert. This assumption was made for the same reason as the assumption of no missing minutiae. In the extended investigation, which is described in the current chapter, the assumption of 'perfect' minutiae matching has been relaxed. So, instead of matching the minutiae manually, as in Chapter 5, the minutiae were matched automatically (which makes it likely that some matching errors were introduced in the process). The reason that this assumption was relaxed is similar to the reason that the no missing minutiae assumption was relaxed: we wanted to evaluate the performance of our proposed fingerprint construct in a *practical* context, in which minutiae extraction and matching errors may exist and are thus likely to have an adverse effect on the resulting recognition accuracy.

The final difference between the experiments discussed in this chapter and those from Chapter 5 concerns the construction of reference *N*-node Patterns. In Chapter 5, *any N* reference minutiae could be used in the construction of a reference *N*-node Pattern. The experiments discussed in the current chapter, however, enforced a minimum separation threshold between a reference Pattern's *N* constituent minutiae, such that no two minutiae in the same Pattern are too close to each other. This was done in order to better contend with minutiae extraction errors in terms of limiting their effect on Pattern matching. More

specifically, consider two minutiae (in the same fingerprint) that are very close to each other. In a different sample of the same fingerprint, if one of those two minutiae is detected in slightly the wrong location, it is possible that the two minutiae would be flipped around. Since the order of the minutiae is important in Pattern construction (see Section 5.2.4), changing the order of two constituent minutiae would essentially result in a *different* Pattern being created, which would cause problems in the order-sensitive Pattern matching (see Section 5.2.5). For this reason, in our extended investigation into the recognition accuracy of our proposed fingerprint construct, we added the additional constraint that only those minutiae that are separated by a minimum distance are allowed to be used in the construction of the same *N*-node Pattern.

The remainder of this chapter is structured as follows. First, the general experimental set-up is explained. In particular, we detail the procedure used to extract the minutiae and core points from each fingerprint in our database, followed by a description of how the reference *N*-node Patterns were constructed. We then discuss the methodology and results of the experiment investigating the effect on the FAR of our proposed fingerprint construct when the number of reference fingerprints used to establish the reference minutiae set increases – this shall henceforth be referred to as Experiment 1. This is followed by an explanation of the details pertaining to the methodology and results of the experiment conducted to evaluate the FAR and FRR of our proposed fingerprint construct in the *Most Favourable Genuine User Authentication Scenario* – this experiment shall henceforth be referred to as Experiment 2. Finally, we evaluate the recognition accuracy when our proposed fingerprint construct is modified to improve Pattern uniqueness. The corresponding experiment shall henceforth be referred to as Experiment 3.

8.2 GENERAL EXPERIMENTAL SET-UP

This section provides details on the general set-up of the experiments discussed in this chapter. In particular, Section 8.2.1 discusses the automated extraction of the minutiae and core points from the fingerprints in our cooperative-user fingerprint database, and Section 8.2.2 explains the process behind constructing reference *N*-node Patterns from multiple reference fingerprints.

8.2.1 Extraction of the Minutiae and Core Points from the Fingerprints

Our proposed fingerprint construct (see Chapter 5) involves the generation of an *N*-node Pattern using a subset of *N* minutiae from a fingerprint's minutiae template. The Pattern is

derived from the relationships between the locations and orientations of the N minutiae, and the Pattern is localised in relation to the location and orientation of the core point. Therefore, the first step in Pattern generation requires the extraction of the minutiae and core point(s) from the underlying fingerprint.

The minutiae and core point(s) were extracted from each of the 800 fingerprints in our database using the VeriFinger 6.7 Software Development Kit [217]. For each feature, its location (i.e., (x, y) coordinates, corresponding to (column, row) indices in the underlying fingerprint image) and orientation (i.e., θ, relative to the horizontal and lying in the range [0°, 360°)) were extracted. Since a core was not detected in 13% (i.e., 104 out of 800) of the fingerprints in our database (mostly the Arch type fingerprints), we had to establish those core points manually[21]. This was done as follows:

- If a core was not detected at all in any of the 8 samples of a fingerprint, then it was established manually as the point of highest ridge curvature. The highest curvature point is commonly considered as the core point in automated core detection, since it ensures that a suitable core is found in *every* type of fingerprint (even the Arch classes).
- If a core was detected in the wrong place in some samples of the same person's fingerprint, but it was correct in their other samples, then the location and orientation of the core in the correct samples were used as a guide towards manually establishing the core point in the incorrect samples.

Note that, in the scenario where more than one core point was detected (which was usually the case with Whorl type fingerprints), only one of the core points was actually used. The upper core point was generally chosen for this purpose, unless the lower core point was more consistently detected in the correct location across multiple samples of the same fingerprint.

8.2.2 Construction of the Reference *N*-node Patterns

The next point to consider in the experimental set-up is the construction of the reference N-node Patterns, which would be stored in the recognition system's database. The reference Patterns came from the reference minutiae set of each person in our database. For Experiment 1, each person's reference minutiae set consisted of *only* those minutiae that were present in *all of the first n* of that person's fingerprint, where n was varied from 1 to 7. Consequently,

[21] Otherwise, since the core point is a necessary feature of the Pattern method, we would have had to exclude 104 fingerprints from our experiments, which would have decreased the size of our database from 800 to 696 fingerprints. An alternative could have been to consider a failed core extraction as a failed authentication attempt; however, this would have portrayed an unfair image of our proposed fingerprint construct, since we cannot assume that *every* core extraction algorithm would produce the same errors.

only the 8th fingerprint sample for each person remained to be used as the query fingerprint in Experiment 1. For Experiments 2 and 3, the *first* 5 of each person's 8 fingerprint samples were used as their *reference* fingerprints, and the *last* 3 of their fingerprint samples were used as the *query* fingerprints; so, each person's reference minutiae set consisted of *only* those minutiae that were present in *all of the first 5* of that person's fingerprint. This is based on our analysis in Chapter 7, which showed that, for our cooperative-user fingerprint database, a genuine user of our proposed fingerprint construct would have the best chance of being successfully authenticated if 5 samples of their reference fingerprint were used during enrolment to establish their reference minutiae set and the user was given a maximum of 3 authentication attempts.

Note that identifying minutiae that are present across multiple samples of a reference fingerprint is fairly straightforward for an informed human, since they can simply look at all the fingerprint images simultaneously and find the corresponding minutiae across them. However, this was more difficult to achieve with an automated minutiae matcher. As for the minutiae and core extraction, we employed the VeriFinger SDK [217] to match the minutiae across multiple samples of the same fingerprint and thus establish minutiae correspondences. Since this software could only compare two fingerprints at a time, however, it was necessary to develop a separate algorithm for establishing minutiae correspondences across n fingerprint samples, where $1 < n \leq 7$. The minutiae correspondences across the first n of each person's fingerprint samples were thus established as follows.

Each of the n fingerprints was used as the reference fingerprint in turn, while the remaining $n - 1$ of the n samples of the same person's fingerprint were used as the test fingerprints. The reference fingerprint was matched against each test fingerprint in turn, and a list of mated minutiae indices was obtained for each comparison. So, for example, when $n = 5$, there were $5 \times 4 = 20$ comparisons in total, thus 20 lists of mated minutiae indices were produced for each person in the database. Figure 8.1(a) illustrates the process by which the first 4 lists of mated minutiae indices were obtained, and Figure 8.1(b) depicts the process used to obtain the next 4 lists of mated minutiae indices. This process was repeated until each fingerprint had a turn being the reference, at the end of which we had 20 lists of mated minutiae indices.

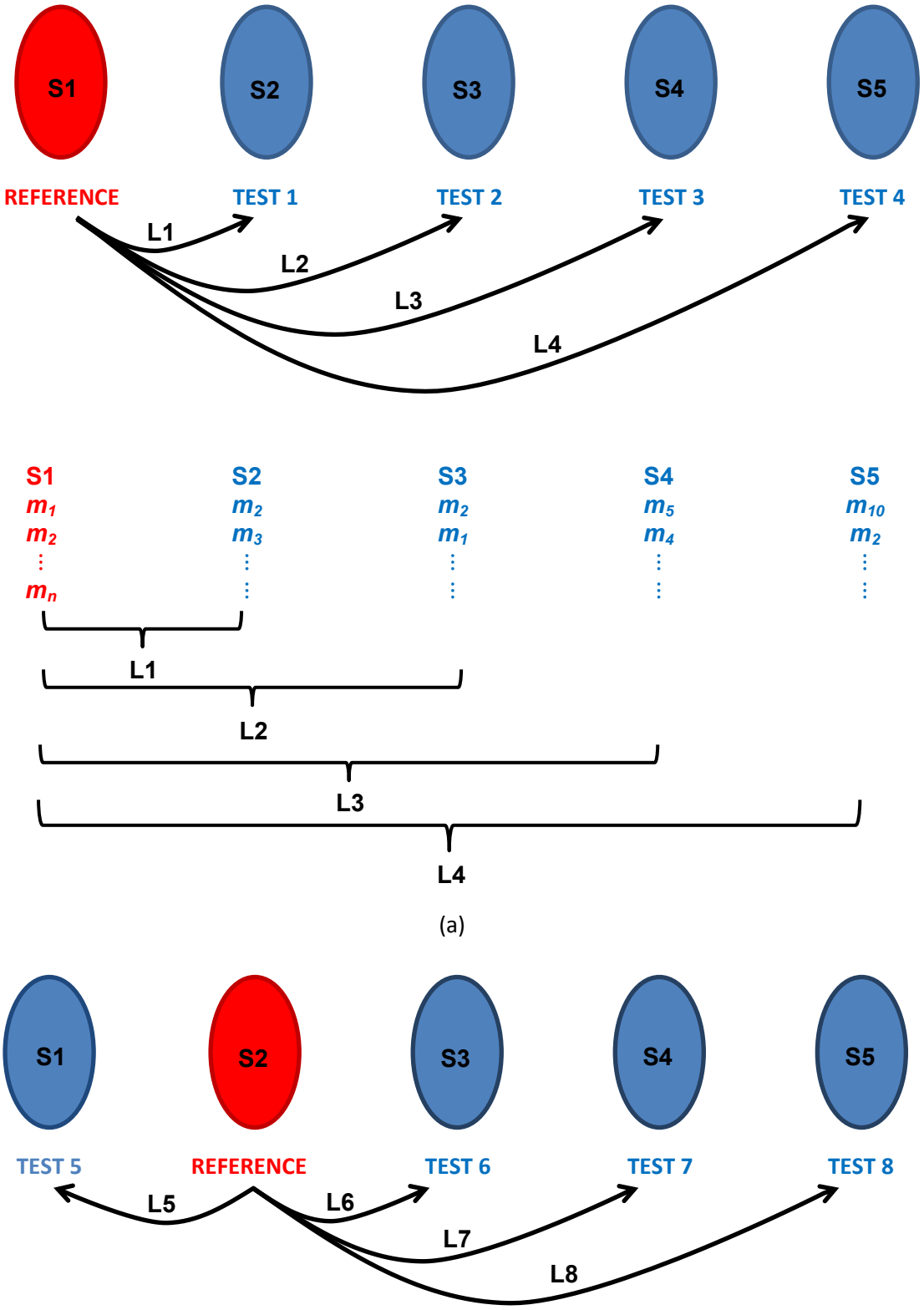


Figure 8.1: Illustration of the method used to obtain (a) the first 4 lists of mated minutiae indices (L1 to L4), and (b) the next 4 lists of mated minutiae indices (L5 to L8), where S*i* refers to fingerprint sample *i*. This process continued until we had 20 lists of mated minutiae indices (i.e., until we reached L20).

Continuing with our example for n = 5, the information from the 20 lists of mated minutiae indices was then consolidated to determine the matching minutiae across all 5 fingerprint samples. Each minutia was processed in turn to find its matching minutiae, starting from the first minutia in the first fingerprint sample and finishing with the last minutia in the fifth fingerprint sample. For example, to determine the matching minutiae for the first minutia, m_1, in the first fingerprint sample, we would proceed as follows. We would first look at L1 to L4 to establish what minutiae in S2, S3, S4 and S5 are matched to m_1 from S1 when S1 is used as the reference fingerprint. Let us say we obtain the following information:

S1	S2	S3	S4	S5
m_1	m_2	m_2	m_5	m_{10}
m_2	m_3	m_1	m_4	m_2
⋮	⋮	⋮	⋮	⋮
m_n	⋮	⋮	⋮	⋮

This tells us that, when the first fingerprint sample, S1, is used as the reference fingerprint, minutia number 1 from the first fingerprint sample, S1, matches minutia number 2 from S2 and S3, minutia number 5 from S4 and minutia number 10 from S5. This information would then be combined into a row vector as follows:

1 2 2 5 10

We would then proceed to look at the mated minutiae indices in L5 to L8, to determine what minutiae in S1, S3, S4 and S5 are matched to m_1 in S1 when S2 is used as the reference fingerprint. Let us say we obtain the following information:

S1	S2	S3	S4	S5
m_4	m_1	m_5	m_1	m_3
m_1	m_2	m_3	m_5	m_{10}
⋮	⋮	⋮	⋮	⋮
⋮	m_k	⋮	⋮	⋮

This tells us that, when the second fingerprint sample, S2, is used as the reference fingerprint, minutia number 1 from the first fingerprint sample, S1, matches minutia number 2 from S2, minutia 3 from S3, minutia 5 from S4 and minutia 10 from S5. We would then combine this information into another row vector and place the new vector below the previous row vector, as follows:

1	2	2	5	10
1	2	3	5	10

Looking at the two rows above, we can see that there is a discrepancy in the information between them; in particular, row 1 tells us that minutiae 1, 2, **2**, 5 and 10 correspond between this person's first 5 fingerprint samples, while row 2 tells us that minutiae 1, 2, **3**, 5 and 10 correspond. This problem would occur as a result of falsely pairing up two non-matching minutiae, which sometimes happens when using automated minutiae matchers. Incorrect matching is often a consequence of error propagation from the minutiae extraction stage; for example, if a minutia is detected in the wrong place in the reference fingerprint, then it will likely be matched to the wrong minutia in the query fingerprint. Note that, while the VeriFinger SDK [217] must be commended on its minutiae extraction and matching algorithms, unfortunately some errors have been discovered in both processes. For example, often a large number of spurious minutiae were detected in a fingerprint, and sometimes fairly obvious minutiae were not detected. Furthermore, sometimes seemingly random minutiae were matched. These errors naturally had a negative impact on the recognition accuracy of our Pattern method, starting with the establishment of minutiae correspondences across each person's first n fingerprint samples.

To help deal with the discrepancies in minutiae matching, we continued building the aforementioned matrix of row vectors until we had n row vectors per minutia (one row vector per each of the n reference fingerprints). Continuing with the example for $n = 5$ from above, let us say that the 5 vectors corresponding to m_1 from S1 look like this:

1	2	2	5	10
1	2	3	5	10
1	2	3	5	10
1	2	3	5	10
1	2	2	5	10

We then use majority voting on the n rows to establish the *most likely* set of minutiae correspondences across all n samples of each person's fingerprint. Using majority voting on the matrix of 5 rows above, we can see that the following row is the most common:

1	2	3	5	10

So, we conclude that, *most likely*, minutia number 1 from person *i*'s first fingerprint sample matches minutia number 2 from their second fingerprint sample, minutia number 3 from their third sample, minutia number 5 from their fourth sample, and minutia number 10 from their fifth sample.

The procedure described above was applied to establish the entire set of minutiae correspondences across all first *n* fingerprint samples of each person in our database. In the end, the resulting row vectors were all placed into a matrix, and only the *unique* rows of that matrix were kept. Each row of the resulting matrix corresponded to a different minutia, and reading across the *n* columns told us the matching minutiae of that particular minutia across all *n* reference fingerprints. Therefore, only those minutiae that have a matching minutia in each of the *n* fingerprint samples were kept to form the *reference* minutiae set; so, there were effectively *n* corresponding reference minutiae sets for each person in our database.

Once a person's reference minutiae sets were established, *n* "versions" of a reference *N*-node Pattern were constructed by using the corresponding sets of *N* minutiae from the *n* reference minutiae sets. Construction of an *N*-node Pattern was done in the same way as for the experiments in Chapter 5, except for one additional practical constraint. In the preliminary experiments in Chapter 5, *all* the minutiae in the reference minutiae set were considered suitable candidates for Pattern construction. However, in the experiments in this chapter, a minimum minutiae separation threshold was enforced to ensure that a Pattern's constituent minutiae were not too close. If two minutiae are too close together, then a slight inaccuracy in the detection of either of the minutiae in a different sample of the same fingerprint could essentially 'flip' the two minutiae around. This would cause problems in Pattern matching, since the order of the minutiae in the query Pattern would be different to the order of the same minutiae in the reference Pattern. For this reason, it was decided that only those minutiae that were sufficiently far apart could be used in the construction of the same Pattern. To determine the minimum separation threshold, the Euclidean distance between each pair of minutiae in every reference minutiae set was calculated. The first percentile[22] of the resulting Euclidean distances distribution was used to determine the minimum separation threshold. The first percentile was found to be between 18 and 19 for each reference minutiae set (i.e., corresponding to each *n*), so the minimum minutiae separation threshold was set to 19. This implies that only those minutiae for which the Euclidean distance between them was at least 19 could be used in the construction of the *same* Pattern.

[22] We used the first percentile because this would ensure that about 99% of the minutiae pairs were above the minimum distance requirement; consequently, only a small number of minutiae pairs would actually be considered 'invalid'.

After creating the *n* "versions" of a reference *N*-node Pattern from each of a person's *n* reference minutiae sets, the *n* "versions" were averaged by averaging the corresponding attributes in the Pattern feature vectors. In the end, *one* overall reference *N*-node Pattern was obtained, and this Pattern was stored in the database for matching purposes. For example, let us denote the *n* "versions" of a particular *N*-node Pattern (where each "version" comes from one of the *n* reference minutiae sets) as follows:

$$v^1 = [l_{12}^1, \alpha_{12}^1, \beta_{12}^1, l_{23}^1, \alpha_{23}^1, \beta_{23}^1, \dots, l_{N1}^1, \alpha_{N1}^1, \beta_{N1}^1, x^1, y^1, \omega^1]$$
$$v^2 = [l_{12}^2, \alpha_{12}^2, \beta_{12}^2, l_{23}^2, \alpha_{23}^2, \beta_{23}^2, \dots, l_{N1}^2, \alpha_{N1}^2, \beta_{N1}^2, x^2, y^2, \omega^2]$$
$$\vdots$$
$$v^n = [l_{12}^n, \alpha_{12}^n, \beta_{12}^n, l_{23}^n, \alpha_{23}^n, \beta_{23}^n, \dots, l_{N1}^n, \alpha_{N1}^n, \beta_{N1}^n, x^n, y^n, \omega^n]$$

The attributes of the *reference* *N*-node Pattern feature vector, v^R, are then computed as follows:

1. The *l* attributes:

$$l_{12}^R = \frac{1}{n}\sum_{i=1}^{n} l_{12}^i \tag{8.1}$$

$$l_{23}^R = \frac{1}{n}\sum_{i=1}^{n} l_{23}^i \tag{8.2}$$

$$\vdots$$

$$l_{N1}^R = \frac{1}{n}\sum_{i=1}^{n} l_{N1}^i \tag{8.3}$$

2. The α attributes:

$$\alpha_{12}^R = \tan^{-1}\left(\frac{\frac{1}{n}\sum_{i=1}^{n}\sin(\alpha_{12}^i)}{\frac{1}{n}\sum_{i=1}^{n}\cos(\alpha_{12}^i)}\right) \tag{8.4}$$

$$\alpha_{23}^R = \tan^{-1}\left(\frac{\frac{1}{n}\sum_{i=1}^{n}\sin(\alpha_{23}^i)}{\frac{1}{n}\sum_{i=1}^{n}\cos(\alpha_{23}^i)}\right) \tag{8.5}$$

$$\vdots$$

$$\alpha_{N1}^R = \tan^{-1}\left(\frac{\frac{1}{n}\sum_{i=1}^{n}\sin(\alpha_{N1}^i)}{\frac{1}{n}\sum_{i=1}^{n}\cos(\alpha_{N1}^i)}\right) \tag{8.6}$$

3. The β attributes:

$$\beta_{12}^{R} = \tan^{-1}\left(\frac{\frac{1}{n}\sum_{i=1}^{n}\sin(\beta_{12}^{i})}{\frac{1}{n}\sum_{i=1}^{n}\cos(\beta_{12}^{i})}\right) \tag{8.7}$$

$$\beta_{23}^{R} = \tan^{-1}\left(\frac{\frac{1}{n}\sum_{i=1}^{n}\sin(\beta_{23}^{i})}{\frac{1}{n}\sum_{i=1}^{n}\cos(\beta_{23}^{i})}\right) \tag{8.8}$$

$$\vdots$$

$$\beta_{N1}^{R} = \tan^{-1}\left(\frac{\frac{1}{n}\sum_{i=1}^{n}\sin(\beta_{N1}^{i})}{\frac{1}{n}\sum_{i=1}^{n}\cos(\beta_{N1}^{i})}\right) \tag{8.9}$$

4. The x, y, and ω attributes:

$$x^{R} = \frac{1}{n}\sum_{i=1}^{n} x^{i} \tag{8.10}$$

$$y^{R} = \frac{1}{n}\sum_{i=1}^{n} y^{i} \tag{8.11}$$

$$\omega^{R} = \tan^{-1}\left(\frac{\frac{1}{n}\sum_{i=1}^{n}\sin(\omega^{i})}{\frac{1}{n}\sum_{i=1}^{n}\cos(\omega^{i})}\right) \tag{8.12}$$

In the end, the averaged attributes from these n Pattern feature vectors are combined to create the final *reference* N-node Pattern feature vector, which takes the following form:

$$v^{R} = [l_{12}^{R}, \alpha_{12}^{R}, \beta_{12}^{R}, l_{23}^{R}, \alpha_{23}^{R}, \beta_{23}^{R}, \dots, l_{N1}^{R}, \alpha_{N1}^{R}, \beta_{N1}^{R}, x^{R}, y^{R}, \omega^{R}]$$

8.3 EXPERIMENT 1: EFFECT OF MULTIPLE REFERENCE FINGERPRINTS ON FAR

In Section 7.2, we showed that increasing the number of reference fingerprints used during enrolment to establish the most reliable reference minutiae set considerably improves the true FRR of our proposed fingerprint construct. This section discusses the methodology and results of Experiment 1, which was used to determine the effect that increasing the number of reference fingerprints has on the FAR of our proposed fingerprint construct, whilst also empirically validating the effectiveness of this scheme for improving the FRR. Note that

Single-Factor Authentication was adopted for this experiment for the same reason as in the preliminary experiments[23] in Chapter 5. Since the aim of this experiment was to empirically evaluate and compare the effect that increasing the number of reference fingerprints has on the FAR and FRR, however, we may reasonably deduce that the trend obtained for *Single-Factor Authentication* also applies to *Two-Factor Authentication*.

Section 8.3.1 explains how the matching thresholds were established for this experiment. Section 8.3.2 discusses the Pattern matching procedure. Section 8.3.3 presents the resulting FAR and FRR as the number of reference fingerprints increases from 1 to 7.

8.3.1 Selection of the Matching Thresholds for Experiment 1

The matching thresholds refer to the maximum allowable differences between corresponding attributes of the reference and query Pattern feature vectors in order to consider the two vectors as matching (see Section 5.2.5 for an explanation of how Pattern matching is conducted). Since the aim of Experiment 1 was only to *compare* the FAR and FRR as the number of reference fingerprints, n, increases, the matching thresholds were not so important. This is because we were only interested in the FAR and FRR *trends*, as opposed to their actual *values*. Nevertheless, we wanted to use suitable matching thresholds, so they were determined in a methodical manner similar to the procedure adopted for the preliminary experiments in Chapter 5.

Firstly, for each person, we established which minutiae are present in all 8 of their fingerprint samples. Then, the resulting, filtered set of minutiae in each of the 8 fingerprint samples was used to create every possible 3-node Pattern. Since we knew which minutiae in one fingerprint sample corresponded to which minutiae in the other 7 fingerprint samples, this meant that we could easily establish the corresponding 3-node Patterns across all 8 of a person's fingerprint samples. Next, all the corresponding 3-node Patterns were compared to each other, and the differences between the corresponding Pattern attributes for each comparison for every person were used to plot a distribution. The 99th percentile of each distribution was then used as the matching threshold for that particular attribute. Table 8.1 summarises the resulting 99th percentiles and the corresponding matching thresholds[24].

[23] See Section 5.3.1.

[24] Note that the α and β attributes were considered together, since the nature of these attributes means that they are likely to require the same matching threshold. See Section 5.2.2 for a thorough description of the α and β Pattern attributes.

Table 8.1: The 99th percentiles of Pattern attribute differences, and the corresponding matching thresholds.

Pattern Attribute	99th Percentile of Differences	Matching Threshold
l	14.15	$\lceil 14.15 \rceil = 15$
α and β	12.76°	$\lceil 12.76° \rceil = 13°$
(x, y)	27.63	$\lceil 27.63 \rceil = 28$
ω	28.83°	$\lceil 28.83° \rceil = 29°$

The following points should be noted with regards to the threshold selection algorithm for Experiment 1:

- Only 3-node Patterns were considered, because the aim of Experiment 1 was simply to show the trend in the FAR compared to the trend in the FRR as the number of reference fingerprints, n, increases. So, we essentially could have used any Pattern size. However, the memory requirements and experimental run time involved in comparing such a large number of Patterns for threshold selection meant that this procedure could only be practically applied to 3-node Patterns at the present time.
- As for Experiment 1 in Chapter 5, here we used every *combination* of 3 minutiae for reference Pattern construction, instead of every *permutation*. This was due to two reasons. Firstly, since a permutation of a Pattern is obtained by rearranging the order of its constituent minutiae, this means that, if a particular Pattern is found in a query fingerprint, then all permutations of that Pattern will also be found, and vice-versa if the Pattern is *not* found. Secondly, using permutations instead of combinations results in an impractically long experimental run-time (see the analysis in Section 5.3.1.3). So, constructing reference Patterns via every *permutation* of the constituent minutiae was not necessary, since *combinations* were both sufficient and more efficient for our purposes.
- In practice, the query fingerprint (i.e., the 8th sample for each person in our database) would not be available for establishing the matching thresholds. However, the fact that we used all 8 of each person's fingerprint samples for determining the matching thresholds for Experiment 1 is not important, since the actual *numbers* obtained for the FAR and FRR in this experiment are not the focus. Instead, the focus is on the *trends* in the FAR and FRR. So, in fact we could have used basically *any* values for the matching thresholds; however, the procedure described in this section was used to determine a set of sensible values.
- Initially, as for the preliminary experiments in Chapter 5, we intended to use the *maximum* value of each Pattern attribute differences distribution to establish the corresponding matching threshold. However, the resulting maximums were unreasonably large, which is

due to errors in the automated feature extraction and minutiae matching processes. For this reason, we used the 99th percentile of each Pattern attribute differences distribution to determine the corresponding matching thresholds. We only used the one set of matching thresholds (from Table 8.1) for Experiment 1, because here we were not interested in showing the change in performance as the matching thresholds vary; rather, we only wanted to show the trend in the FAR and FRR as the number of reference fingerprints, *n*, increases.

8.3.2 Matching Procedure for Experiment 1

The final stage in Experiment 1 was the actual matching process, the purpose of which was to compare the trends in the FRR and FAR of our proposed fingerprint construct as the number of reference fingerprints, *n*, increases from 1 to 7. Since this analysis was based on the *Single-Factor Authentication* method, the matching stage for Experiment 1 was conducted as follows:

1. For each *n* in the range [1, 7], 100 different, corresponding 3-minutiae combinations were randomly selected from each of the *n* reference minutiae sets for each of the 100 people in our database. This was done only for $N = 3$, since, as stated in Section 8.3.1, a single Pattern size was sufficient for the purposes of Experiment 1. For each of the 100 minutiae sets for every *n*, a random ordering of the 3 minutiae was chosen and a reference 3-node Pattern was created using the reference Pattern construction process outlined in Section 8.2.2. In the end, there were 100 reference 3-node Patterns for every *n*, for each of the 100 people in our cooperative-user fingerprint database.
2. To calculate the FRR, each reference Pattern was searched for in the *same* person's *last* fingerprint sample to simulate an authentication attempt by a *genuine user*. If a reference Pattern could *not* find a match in the query fingerprint, then this was considered as a False Reject for that particular Pattern. Note that a *match* was found if there existed an *N*-node Pattern whose attributes were similar enough to the attributes of the reference *N*-node Pattern, where the similarity was determined according to the thresholds specified in Table 8.1. The total number of genuine comparisons for each *n* was thus 10,000 (i.e. 100 reference Patterns per person × 100 reference people × 1 genuine attempt). Therefore, the FRR for each *n* was calculated using Equation (8.13):

$$FRR = \frac{Total\ Number\ of\ False\ Rejects}{10{,}000} \times 100\% \qquad (8.13)$$

3. To calculate the FAR, each reference Pattern was searched for in every *other* person's *last* fingerprint sample to simulate an authentication attempt by an *impostor*. If a reference Pattern *found at least one* match in the query fingerprint, then this was considered as a False Accept for that particular Pattern. Note that a *match* was found if there existed an *N*-node Pattern whose attributes were similar enough to the attributes of the reference *N*-node Pattern, where the similarity was determined according to the thresholds specified in Table 8.1. The total number of impostor comparisons for each *n* was thus 990,000 (i.e., 100 reference Patterns per person × 100 reference people × 99 impostor attempts). Therefore, the FAR for each *n* was calculated using Equation (8.14):

$$FAR = \frac{Total\ Number\ of\ False\ Accepts}{990{,}000} \times 100\% \tag{8.14}$$

8.3.3 FAR and FRR for Experiment 1

Before presenting the FAR and FRR results from Experiment 1, it is important to note the following point. To ensure that 100 randomly selected reference 3-node Patterns was sufficient for every *n*, we ran the experiment 3 times. The idea was that, if the corresponding FAR and FRR results across the 3 trials are approximately the same, then 100 reference Patterns is sufficient. Table 8.2 shows the FAR and FRR results for each *n* for all 3 trials:

Table 8.2: FAR and FRR across 3 runs of Experiment 1 as the number of reference fingerprints (*n*) increases from 1 to 7.

Trial #	Results (%)	Number of Reference Fingerprints, *n*						
		***n* = 1**	***n* = 2**	***n* = 3**	***n* = 4**	***n* = 5**	***n* = 6**	***n* = 7**
1	FAR	1.29	1.42	1.47	1.52	1.58	1.62	1.63
	FRR	41.80	25.68	21.97	14.77	11.63	9.04	7.38
2	FAR	1.29	1.42	1.46	1.54	1.57	1.63	1.62
	FRR	41.91	25.43	21.20	15.11	12.20	8.61	7.30
3	FAR	1.29	1.42	1.45	1.57	1.57	1.63	1.62
	FRR	42.93	26.27	21.38	14.96	12.20	8.61	7.30

From Table 8.2, we can see that the corresponding FAR and FRR results for each *n* across the 3 trials are very close in value, and any differences appear insignificant compared to the magnitude of the entire number. Therefore, we may conclude that using 100 reference 3-node Patterns from each of the 100 people in our database for every *n* is sufficient to provide a reasonable estimate of the resulting FAR and FRR.

Table 8.3 depicts the *average* FAR and FRR values across the 3 trials for each n from the results in Table 8.2:

Table 8.3: Average FAR and FRR across the 3 trials from Table 8.2.

	Number of Reference Fingerprints, n						
	$n = 1$	$n = 2$	$n = 3$	$n = 4$	$n = 5$	$n = 6$	$n = 7$
Average FAR (%)	1.29	1.42	1.46	1.54	1.58	1.63	1.63
Average FRR (%)	42.21	25.79	21.52	14.95	12.01	8.75	7.33

From Table 8.3, the most important observation to make is that, as the number of reference fingerprints, n, increases from 1 to 7, the average FRR undergoes a dramatic decrease from 42.21% at $n = 1$ to 7.33% at $n = 7$. This empirically validates our analysis in Section 7.2, where we established that increasing the number of reference fingerprints can dramatically improve the FRR of our proposed fingerprint construct.

Since the trade-off between the FRR and FAR in a fingerprint recognition system is common knowledge, we would assume that the observed dramatic decrease in the FRR as the number of reference fingerprints increases would cause a correspondingly dramatic *increase* in the FAR. However, we note, from Table 8.3, that this is not the case for our proposed fingerprint construct. In particular, although the average FRR was found to decrease by nearly 6 times as n increased from 1 to 7, the average FAR increased by only 1.3 times (from 1.29% at $n = 1$ to 1.63% at $n = 7$). We may thus conclude that increasing the number of reference fingerprints is an effective strategy for reducing the FRR of our proposed fingerprint construct, whilst having an almost negligible effect on the corresponding FAR.

8.4 EXPERIMENT 2: FAR AND FRR IN MFGUAS

The results of Experiment 1 confirm that increasing the number of reference fingerprints is a good method of decreasing the FRR of our proposed fingerprint construct (whilst having a fairly insignificant effect on the FAR). In this section, we discuss the methodology and results pertaining to Experiment 2, which investigates the FAR and FRR trade-off in the second FRR improvement strategy from Chapter 7. In Section 7.3, it was shown that allowing a user to have multiple authentication attempts can reduce the FRR even further. In particular, it was established that a genuine user has the best chance of being successfully authenticated when 5 of their reference fingerprints are used during enrolment to establish their reference minutiae set and the user is then given a maximum of 3 authentication attempts. This was referred to as the *Most Favourable Genuine User Authentication Scenario*

(MFGUAS). This section discusses the methodology and results pertaining to Experiment 2, which investigates the FAR and FRR trade-off in a practical implementation of our proposed fingerprint construct in the MFGUAS. Section 8.4.1 explains how the matching thresholds were established for this experiment. Section 8.4.2 discusses the Pattern matching procedure and the resulting FAR and FRR when the *Single-Factor Authentication* method is used. Section 8.4.3 discusses the Pattern matching procedure and the resulting FAR and FRR when the *Two-Factor Authentication* method is used.

8.4.1 Selection of the Matching Thresholds for Experiment 2

The matching thresholds for Experiment 2 were established in a similar way as for Experiment 1, but with a few important differences. For both experiments, only 3-node Patterns were used to determine the matching thresholds. This is because the threshold selection algorithm involved comparing a large number of Patterns to each other, and the resulting memory requirements and experimental run-time involved in constructing and comparing such a large number of Patterns were too large to render such an experiment feasible for N larger than 3 at the present time. Also, for both experiments, in constructing every possible 3-node Pattern, only *combinations* of 3 minutiae (instead of *permutations*) were considered, for the reasons outlined in Section 8.3.1.

The main difference between the threshold selection algorithms for Experiment 1 and Experiment 2 was that, while the algorithm for Experiment 1 considered corresponding 3-node Patterns across *all 8* of a person's fingerprint samples, the threshold selection algorithm for Experiment 2 only considered corresponding Patterns across *the first 5* of each person's fingerprint samples. This is because the remaining 3 fingerprint samples from each person were to be used as the query fingerprints. Since the actual FAR and FRR values in Experiment 2 matter (as opposed to in Experiment 1, for which only the *trend* was important), and since Experiment 2 considered a *practical* implementation of our fingerprint construct, it was important to keep the training (reference) and testing (query) fingerprints separate. Table 8.4 summarises the maximum, 99^{th} percentile, and 99.5^{th} percentile of the resulting Pattern attribute differences distributions. These were the only statistics that we were interested in, because we wanted to use them to set matching thresholds such that as many *genuine* query Patterns as possible would match their corresponding reference Patterns. This is because our proposed fingerprint construct is targeted at everyday civilian fingerprint recognition applications, so it is important that the FRR is kept low.

Table 8.4: The maximum, 99th percentile, and 99.5th percentile of Pattern attribute differences.

	l Differences	α and β Differences (combined)	(x, y) Differences (Euclidean distance)	ω Differences
Maximum	190.21	152.08°	69.35	155.47°
99th Percentile	15.34	14.00°	21.36	15.78°
99.5th Percentile	19.59	17.22°	25.71	22.13°

From Table 8.4, the first thing we must note is that most of the maximums appear too large, which is especially evident for the angle attributes (i.e., α, β, and ω). For example, a difference of 152.08° between two *corresponding* α or β attributes is unreasonable, since such a large difference should only be possible for two α or β attributes that do *not* correspond. These overly large maximums thus suggest that there were errors in the minutiae extraction and/or core extraction process, or that there were errors in the minutiae matching algorithm used to establish the corresponding minutiae during reference Pattern construction, or that there were errors in both of these modules. This is the main problem with using *automated* feature extractors and matchers, which is why the improvement of automated fingerprint recognition algorithms is an on-going development.

Since the maximum values in Table 8.4 are unreasonably large, we decided not to use them in the establishment of the matching thresholds, since they would be likely to result in an impractically high FAR. For this reason, the matching thresholds were based on the 99th and 99.5th percentiles from Table 8.4, and these thresholds are summarised in Table 8.5:

Table 8.5: Attribute-specific thresholds based on the 99th and 99.5th percentiles from Table 8.4.

Attribute-Specific Matching Threshold	Value based on 99th Percentile	Value based on 99.5th Percentile
τ_l	$\lceil 15.34 \rceil = 16$	$\lceil 19.59 \rceil = 20$
$\tau_{\alpha\beta}$	$\lceil 14.00° \rceil = 14°$	$\lceil 17.22° \rceil = 18°$
τ_{loc}	$\lceil 21.36 \rceil = 22$	$\lceil 25.71 \rceil = 26$
τ_ω	$\lceil 15.78° \rceil = 16°$	$\lceil 22.13° \rceil = 23°$

Each set of thresholds from Table 8.5 was used in turn to empirically evaluate the recognition accuracy of our proposed fingerprint construct in the *Most Favourable Genuine User Authentication Scenario* when the *Single-Factor Authentication* method is adopted and when the *Two-Factor Authentication* method is adopted (see Section 8.4.2 and Section 8.4.3, respectively). The reason that we decided to use two sets of thresholds in this experiment, instead of just one, is because our threshold selection algorithm is not guaranteed to result in

the *optimal* recognition accuracy. This is because different fingerprint recognition applications would have different performance specifications, which means that it is impossible to define the word *optimal* universally. For example, in a high-security application (such as the access of a top-secret location), falsely granting access to an impostor would be more catastrophic than falsely rejecting a genuine user. Such an application would, therefore, use stricter matching thresholds, which would ensure a low FAR at the cost of a higher FRR. Alternatively, an application whose top priority is user convenience would be more lenient with the matching thresholds to ensure that the FRR is as low as possible, while still maintaining an acceptable level of security (i.e., impostor rejection). So, since threshold setting is application-specific, we do not propose an *optimal* threshold selection algorithm, since the definition of *optimal* would be application-dependent. In practice, the *optimal* thresholds would generally be determined by trying out a range of threshold values on a training database of fingerprints and calculating the FAR and FRR at each step. Then, those thresholds that result in the desired FAR and FRR would be selected for the particular application. In Experiment 2, however, we were simply interested in comparing the FRR and FRR resulting from adopting the *Most Favourable Genuine User Authentication Scenario* and we used the two sets of thresholds in Table 8.5 as an illustration of how changing the matching thresholds would be expected to affect the FRR and FAR of our proposed fingerprint construct.

The final stage in Experiment 2 was the actual matching process, the purpose of which was to calculate the FRR and FAR of our proposed fingerprint construct in the MFGUAS. Section 8.4.2 and Section 8.4.3 describe the matching procedure and experimental results for *Single-Factor Authentication* and *Two-Factor Authentication*, respectively.

8.4.2 Experiment 2a: Single-Factor Authentication

The matching procedure for Experiment 2a (i.e., Experiment 2 with the *Single-Factor Authentication* method) was conducted in essentially the same way as for Experiment 1. However, there are a few important differences, so, for the reader's convenience, below we outline the overall matching procedure as applied to Experiment 2a:

1. In Experiment 1, it was experimentally shown that 100 randomly selected reference Patterns from each person is sufficient for reliably estimating the FAR and FRR of our proposed fingerprint construct; therefore, we used the same number of reference Patterns in Experiment 2a. So, the first step in the matching procedure was to randomly select 100 different, corresponding N-minutiae combinations from each of the 5 reference minutiae sets for each of the 100 people in our database. This was done for each value of $N = \{3, 4,$

5} in turn, such that the resulting minutiae sets consisted of N minutiae. For each of the 100 minutiae sets for every N, a random ordering of the N minutiae was chosen and a reference N-node Pattern was created using the reference Pattern construction process outlined in Section 8.2.2. In the end, there were 100 reference N-node Patterns for each value of N = {3, 4, 5}, for each of the 100 people in our cooperative-user fingerprint database.

2. To calculate the FRR, each reference Pattern was searched for in the *same* person's *last 3* fingerprint samples to simulate 3 authentication attempts by a *genuine user*. If a reference Pattern could *not* find a match in *any* of the 3 query fingerprints, then this was considered as a False Reject for that particular Pattern. Note that a *match* was found if there existed an N-node Pattern whose attributes were similar enough to the attributes of the reference N-node Pattern, where the similarity was determined according to the thresholds specified in Table 8.5. The total number of genuine comparisons for each N = {3, 4, 5} was 10,000, as in Experiment 1, and the FRR for each N = {3, 4, 5} was thus calculated using Equation (8.13).
3. To calculate the FAR, each reference Pattern was searched for in every *other* person's last 3 fingerprint samples to simulate 3 authentication attempts by an *impostor*. If a reference Pattern *found at least one* match in *any* of the 3 query fingerprints, then this was considered as a False Accept for that particular Pattern. Note that a *match* was found if there existed an N-node Pattern whose attributes were similar enough to the attributes of the reference N-node Pattern, where the similarity was determined according to the thresholds specified in Table 8.5. The total number of impostor comparisons for each N = {3, 4, 5} was 990,000, as in Experiment 1, and the FAR for each N = {3, 4, 5} was thus calculated using Equation (8.14).

The above 3 steps were repeated for the two sets of matching thresholds from Table 8.5. Table 8.6 shows the FAR and FRR for *Single-Factor Authentication* when the first set of thresholds from Table 8.5 was used (i.e., $\tau_l = 16$, $\tau_{\alpha\beta} = 14°$, $\tau_{loc} = 22$, $\tau_\omega = 16°$), and Table 8.7 shows the FAR and FRR resulting from using the second set of thresholds in Table 8.5 (i.e., $\tau_l = 20$, $\tau_{\alpha\beta} = 18°$, $\tau_{loc} = 26$, $\tau_\omega = 23°$).

Table 8.6: FAR and FRR for Single-Factor Authentication, when $\tau_l = 16$, $\tau_{\alpha\beta} = 14°$, $\tau_{loc} = 22$, $\tau_\omega = 16°$.

	$N = 3$	$N = 4$	$N = 5$
FAR (%)	2.16	0.41	0.08
FRR (%)	2.41	3.09	3.96

Table 8.7: FAR and FRR for Single-Factor Authentication, when $\tau_l = 20$, $\tau_{\alpha\beta} = 18°$, $\tau_{loc} = 26$, $\tau_\omega = 23°$.

	$N = 3$	$N = 4$	$N = 5$
FAR (%)	8.78	2.91	1.03
FRR (%)	1.16	1.70	2.58

From Table 8.6 and Table 8.7, we may draw the following conclusions regarding the performance of our proposed fingerprint construct in the *Most Favourable Genuine User Authentication Scenario* under *Single-Factor Authentication*:

- Increasing the matching thresholds decreases the FRR but increases the FAR. This is because larger thresholds make matching more lenient, so it becomes easier both for a genuine user to be successfully authenticated and for an impostor to be falsely accepted as a genuine user. In practice, our proposed fingerprint construct should be tested at a range of thresholds, and the thresholds that result in the desired balance between the FAR and the FRR for a particular application should be selected.
- Increasing the Pattern size, *N*, decreases the FAR but increases the FRR of our proposed fingerprint construct. This is in line with the findings from the preliminary experiments in Chapter 5, indicating that increasing the Pattern size makes a Pattern more *unique*.
- The decrease in the FAR as *N* increases is more dramatic than the corresponding increase in the FRR. For example, our results in Table 8.6 show that increasing *N* by 1 decreases the FAR by approximately 5 times, while the FRR increase is only about 1.3 times. In Table 8.7, an increase in *N* by 1 results in a decrease in FAR by about 3 times, while the FRR increases by only about 1.5 times. This indicates that increasing *N* by a single minutia can have quite a dramatic influence on the Pattern *uniqueness* and thus the resulting FAR, whilst having a less pronounced effect on the FRR.
- For an application where user convenience is the most important factor, and thus a low FRR is essential, we would recommend using 3-node Patterns. In our experiments, the lowest FRR of 1.16% was achieved with 3-node Patterns when the second set of matching thresholds was used (i.e., $\tau_l = 20$, $\tau_{\alpha\beta} = 18°$, $\tau_{loc} = 26$, $\tau_\omega = 23°$) (see Table 8.7). This FRR would ensure that user convenience is kept high, while the FAR of 8.78% would be suitable for a low-security application, where impostor attacks are unlikely. Note that the FRR can be further decreased by increasing the matching thresholds.
- For an application where security is critical, and thus a low FAR is of paramount importance, we would recommend using 5-node Patterns. In our experiments, the lowest FAR of 0.08% was achieved with 5-node Patterns when the first set of matching thresholds was used (i.e., $\tau_l = 16$, $\tau_{\alpha\beta} = 14°$, $\tau_{loc} = 22$, $\tau_\omega = 16°$) (see Table 8.6). This FAR

would ensure that impostor access is kept very low, and the FRR of 3.96% is reasonable considering the trade-off we must make for an extremely secure application. Once again, note that the FAR versus FRR trade-off can be further optimised by varying the matching thresholds in practice.

Overall, Experiment 2a produced highly encouraging results for the recognition accuracy of our proposed fingerprint construct in the *Most Favourable Genuine User Authentication Scenario* under *Single-Factor Authentication*, thus further supporting the suitability of the construct for deployment in cooperative-user civilian fingerprint recognition applications. It was shown that the proposed fingerprint construct can be tuned to the performance requirements of different applications by varying the Pattern size and determining suitable matching thresholds. In particular, we showed that the use of 3-node Patterns together with larger matching thresholds would be most suitable for low-security applications, in which user convenience is the top priority and the FRR must therefore be kept as low as possible. Alternatively, it was demonstrated that the use of 5-node Patterns together with lower matching thresholds would be more suited for high-security applications, in which preventing impostor access is of paramount importance and it is thus vital to keep the FAR as low as possible. Note the following two points:

- The results presented in this section are based on only one commercial fingerprint feature extractor and matcher [217], which, although fairly robust, does make mistakes in accurately extracting the minutiae and core points and in finding matching minutiae between two samples of the same fingerprint during the establishment of reliable reference minutiae at the time of Pattern enrolment. Using a more robust algorithm would improve the results further.
- The results presented in this section are not based on *optimal* matching thresholds, as stated in Section 8.4.1, because what is *optimal* depends on the requirements of a particular application (i.e., the desired balance between the FAR and FRR). Furthermore, since Pattern matching relies on 4 matching thresholds, a robust method of determining the optimal mix of those 4 thresholds must be developed. This will depend on which Pattern attributes can be more/less reliably computed, which will in turn depend on the quality of the adopted feature extractor[25], the user-friendliness of the fingerprint scanner[26],

[25] For example, if the adopted core extraction algorithm is not very reliable, then it might be necessary to use more lenient matching thresholds for the Pattern attributes that are influenced by the core point, i.e., the *global* attributes.

[26] For example, the scanner might have a small scanning surface, so that it is difficult to capture a whole fingerprint.

etc. Consequently, at this stage it is difficult to propose a meaningful general threshold selection algorithm, since this will be influenced by a number of factors that are specific to the nature of a particular application and its adopted fingerprint processing algorithms. For this reason, we leave the development of a suitable threshold selection algorithm as part of future work to be conducted in the context of a specific application.

8.4.3 Experiment 2b: Two-Factor Authentication

Experiment 2b (i.e., Experiment 2 with the *Two-Factor Authentication* method) was actually conducted simultaneously with Experiment 2a, since the majority of the experimental procedure used in Experiment 2b was the same as that used in Experiment 2a. In particular, Steps 1 and 2, which are outlined in Section 8.4.2 for Experiment 2a, were the same for Experiment 2b. Note that the FRR for *Two-Factor Authentication* was calculated in the same way as the FRR for *Single-Factor Authentication* in order to simulate the scenario where a genuine user always presents the correct reference *N*-node Pattern[27] along with their fingerprint during authentication. In this case, a False Reject would only result in the scenario where the reference *N*-node Pattern *does not exist* in the user's query fingerprint, which boils down to the definition of a False Reject under *Single-Factor Authentication*.

The difference between Experiment 2a and Experiment 2b thus lies in the way in which the FAR was calculated. In Experiment 2a, which corresponds to *Single-Factor Authentication*, a False Accept was considered to occur if a genuine user's reference *N*-node Pattern was simply found to *exist* in at least one of 3 different samples of the same impostor's fingerprint (see Step 3 in Section 8.4.2). The *Single-Factor Authentication* method may thus be considered as the *Two-Factor Authentication* method in the scenario in which an impostor *knows* a genuine user's reference *N*-node Pattern; consequently, authentication now only relies upon that Pattern *existing* in the impostor's own fingerprint. In practice, however, an impostor should *not* know a genuine user's reference *N*-node Pattern, and it was the aim of Experiment 2b to evaluate the FAR in this more realistic implementation of *Two-Factor Authentication*. The FAR for Experiment 2b (i.e., Step 3) was thus calculated as follows:

3. Each of every person's 100 reference *N*-node Patterns was searched for in every *other* person's last 3 fingerprint samples to simulate 3 authentication attempts by an *impostor*. Let *M* denote the number of *N*-node Patterns in a *single* impostor fingerprint that match a

[27] As stated in Section 5.3.1.5, an investigation into how easy it is for a user to remember their reference *N*-node Pattern is better left as a subject for future work. This is because the ability of a user to remember their reference *N*-node Pattern would be influenced by many factors, such as the frequency with which the user uses their Pattern to log into a particular application, the age of the user, the complexity of their chosen Pattern, etc. For this reason, such an investigation would be most beneficial in the context of a particular application.

genuine user's reference *N*-node Pattern. Further, let *P* denote the total number of *N*-node Patterns possible from the entire set of *T* minutiae available in the impostor's fingerprint. Then the probability of the impostor successfully presenting a matching *N*-node Pattern from their own fingerprint was calculated using Equation (8.15):

$$P_{success} = \frac{M}{P} = \frac{M}{\left(\frac{T!}{(T-N)!}\right)} \tag{8.15}$$

The *largest* of the three $P_{success}$ values calculated from the impostor's 3 fingerprint samples was used towards calculating the FAR, in order to ensure that the impostor, like the genuine user, was allowed to present a maximum of 3 different samples of their fingerprint for 3 authentication attempts. Let $P_{success}^{MAX}$ denote the largest of the impostor's three $P_{success}$ values. The FAR for *Two-Factor Authentication* was then calculated using Equation (8.16), in which it is assumed that the impostor presents their *own* reference *N*-node Pattern along with their *own* fingerprint at every authentication attempt[28]:

$$FAR_{2F} = \frac{\sum_{i=1}^{990,000}(P_{success}^{MAX})_i}{990,000} \times 100\% \tag{8.16}$$

Note that, in Equation (8.16), the total number of impostor attempts is 990,000, which is the same as the total number of impostor attempts used to calculate the FAR for *Single-Factor Authentication* (see Equation (8.14)).

Table 8.8 compares the FAR obtained for *Two-Factor Authentication* to the FAR obtained for *Single-Factor Authentication* in Experiment 2a (note that the FRR is the same in both cases) when the first set of thresholds from Table 8.5 was used (i.e., $\tau_l = 16$, $\tau_{\alpha\beta} = 14°$, $\tau_{loc} = 22$, $\tau_\omega = 16°$), and Table 8.9 compares the resulting FARs when the second set of matching thresholds from Table 8.5 was used (i.e., $\tau_l = 20$, $\tau_{\alpha\beta} = 18°$, $\tau_{loc} = 26$, $\tau_\omega = 23°$).

[28] In practice, an impostor may attempt to *guess* a genuine user's reference *N*-node Pattern by trying out multiple *N*-node Patterns from their own fingerprint. Let *G* denote the total number of allowed guesses. Then the probability of an impostor succeeding in this endeavour may be estimated as $G \times P_{success}$. Provided that *G* is much smaller than the total number of *N*-node Patterns possible from the impostor's fingerprint, which should be the case in practice, the impostor's probability of success may be approximated by $P_{success}$.

Table 8.8: Comparison of the FARs for Single-Factor Authentication and Two-Factor Authentication, when $\tau_l = 16$, $\tau_{\alpha\beta} = 14°$, $\tau_{loc} = 22$, $\tau_\omega = 16°$.

	$N = 3$	$N = 4$	$N = 5$
Single-Factor Authentication FAR (%)	2.16	0.41	0.08
Two-Factor Authentication FAR (%)	3.37×10^{-5}	1.53×10^{-7}	7.90×10^{-10}

Table 8.9: Comparison of the FARs for Single-Factor Authentication and Two-Factor Authentication, when $\tau_l = 20$, $\tau_{\alpha\beta} = 18°$, $\tau_{loc} = 26$, $\tau_\omega = 23°$.

	$N = 3$	$N = 4$	$N = 5$
Single-Factor Authentication FAR (%)	8.78	2.91	1.03
Two-Factor Authentication FAR (%)	1.90×10^{-4}	1.57×10^{-6}	1.38×10^{-8}

From Tables 8.8 and 8.9, it is evident that the FAR for *Two-Factor Authentication* is significantly lower than the FAR for *Single-Factor Authentication*, for all Pattern sizes at both sets of matching thresholds. This is because the FAR in *Single-Factor Authentication* depends only upon a genuine user's reference *N*-node Pattern *existing* in an impostor's fingerprint, while the FAR in *Two-Factor Authentication* additionally depends upon the impostor identifying the matching *N*-node Pattern. Due to the large number of *N*-node Patterns available in a fingerprint, the likelihood of an impostor guessing the matching Pattern would be expected to be very low; indeed, the FAR results presented in Table 8.8 and Table 8.9 confirm this expectation.

Overall, the results obtained from Experiment 2b strongly suggest that *Two-Factor Authentication* provides a considerably higher level of security than *Single-Factor Authentication*, in terms of making it more difficult for an impostor to falsely authenticate as a genuine user. In fact, the difficulty of guessing a matching *N*-node Pattern from the large number of *N*-node Patterns available in a single fingerprint suggests that, even if an impostor were to gain access to a genuine user's entire fingerprint, it would still be extremely difficult for them to use that fingerprint to authenticate as the genuine user. This is in contrast to *Single-Factor Authentication*, where an attacker that has access to a genuine user's fingerprint would be able to use that fingerprint directly to fool the recognition system into accepting them as the genuine user[29], since they do not need to know the user's reference *N*-node Pattern. For this reason, we recommend adopting *Two-Factor Authentication* in favour of *Single-Factor Authentication* in practice. While the requirement for a user to remember their

[29] This would only be possible if the attacker were able to use the stolen fingerprint to create a physical spoof, or if they were able to replay the digital fingerprint image in the channel between the sensor and feature extractor, etc. For more information on typical attack points in a fingerprint recognition system, see Section 2.2.

reference *N*-node Pattern may be of some inconvenience to the user, we believe that this trade-off is worthwhile for the considerable gain in security.

An important point to note is that the experiments presented in this section, and indeed in this chapter as a whole, were based on the assumption that a user of our proposed fingerprint construct would essentially be scanning their fingerprint 'blindly' during authentication. In other words, it is assumed that the user would not be able to see what the fingerprint scanner sees until their fingerprint image has been acquired. In this case, the user could not be sure whether or not the part of their fingerprint in which their reference Pattern is contained has been captured; therefore, multiple authentication attempts may be necessary. In practice, however, we would recommend incorporating a live scan of the user's finger on the scanner, which should not be difficult to do; for example, the Futronic FS88 scanner, which was used for the acquisition of our cooperative-user fingerprint database described in Chapter 6, has this function incorporated into the simple user interface that comes with the scanner. In this case, a user of our proposed fingerprint construct could move their finger around on the scanner until they can see that the minutiae used to generate their reference *N*-node Pattern exist in the fingerprint image appearing on screen. They can then simply indicate to the system that they are ready for image acquisition, and the acquired fingerprint image will contain the relevant minutiae. Provided that the adopted feature extractor is able to correctly extract those minutiae, only one authentication attempt should be necessary to ensure that the user's reference *N*-node Pattern exists in their query fingerprint.

8.5 FAR AND FRR WHEN PROPOSED FINGERPRINT CONSTRUCT IS MODIFIED TO IMPROVE PATTERN UNIQUENESS

The FAR and FRR results achieved in Experiments 2a and 2b indicate that, by appropriately tuning the matching thresholds and selecting the most appropriate Pattern size, our proposed fingerprint construct would be suitable for use in civilian fingerprint recognition applications. In particular, we have shown that, despite the small size of the *N*-node Patterns, our proposed fingerprint construct is capable of effectively discriminating between a genuine user and an impostor. This was especially evident for *Two-Factor Authentication*, which is why we recommend that this form of authentication, as opposed to *Single-Factor Authentication*, be adopted in practice.

While the probability of an impostor being falsely authenticated as a genuine user was found to be extremely small for *Two-Factor Authentication*, in practice this extra security would drop to the FAR achievable with *Single-Factor Authentication* in the event that an

impostor *knows* a genuine user's reference Pattern and then attempts to use it with their own fingerprint to access the recognition system. For this reason, even if *Two-Factor Authentication* is adopted in practice, it is important to select the matching thresholds such that the FAR for *Single-Factor Authentication* is sufficiently low[30] to prevent the false acceptance of impostors that may have acquired the enrolled (reference) Pattern of a genuine user. As an additional layer of security, we now propose a modification to our fingerprint construct, which improves the *uniqueness* of *N*-node Patterns and thus further decreases the FAR for both *Single-Factor Authentication* and *Two-Factor Authentication*.

In Section 5.2.2, it was shown that, due to the use of Equations (5.3) and (5.4), the α and β attributes of an *N*-node Pattern are restricted to the range [0°, 180°) instead of being allowed to lie within the [0°, 360°) range. The ensuing discussion went on to prove that this restriction does not incur an information loss in the resulting α and β attributes, apart from in the case where the connection line entering a minutia is collinear with the connection line exiting the same minutia. Since the latter scenario was deemed unlikely to occur frequently in practice, it was concluded that this restriction would be expected to have a negligible effect on Pattern *uniqueness* and thus the recognition accuracy of our proposed fingerprint construct.

In Section 5.2.3, it was shown that, due to the use of Equation (5.13), which restricts θ_l to the range [0°, 180°), the y attribute, corresponding to the vertical displacement of an *N*-node Pattern from the fingerprint core, can only take on positive values. Consequently, it was concluded that this would incur some information loss in the resulting *N*-node Pattern, since there would be ambiguity as to the true y value. A similar conclusion was drawn for the ω attribute, whose restriction to the range [0°, 180°) arises as a result of using Equation (5.17). However, since these ambiguities in the true y and ω attributes would, in practice, only become problematic for Pattern *uniqueness* if there happened to exist two or more *N*-node Patterns that were the exact mirror images of each other in terms of their location and orientation, and since such an occurrence was deemed unlikely, it was reasoned that this information loss would not have a significant effect on the recognition accuracy of our proposed fingerprint construct.

In this section, we empirically verify the assumptions made in Sections 5.2.2 and 5.2.3, regarding the effect that limiting a Pattern's angle attributes to the range [0°, 180°) would have on the resulting recognition accuracy. To conduct this investigation for the *Single-Factor Authentication* scenario and the *Two-Factor Authentication* scenario, we essentially repeated Experiments 2a and 2b, respectively, except that Equations (5.3), (5.4), (5.13), and

[30] This level would depend on the security requirements of a particular application.

(5.17), which were used for Pattern formation in Experiments 2a and 2b, were now replaced with Equations (8.17), (8.18), (8.19), and (8.20), respectively:

$$\alpha_{ij} = \begin{Bmatrix} \varphi_{ij} - \theta_i & , \varphi_{ij} \geq \theta_i \\ \varphi_{ij} - \theta_i + 360° & , \varphi_{ij} < \theta_i \end{Bmatrix}, \quad 0° \leq \alpha_{ij} < 360° \tag{8.17}$$

$$\beta_{ij} = \begin{Bmatrix} \varphi_{ij} - \theta_j & , \varphi_{ij} \geq \theta_j \\ \varphi_{ij} - \theta_j + 360° & , \varphi_{ij} < \theta_j \end{Bmatrix}, \quad 0° \leq \beta_{ij} < 360° \tag{8.18}$$

$$\gamma = \begin{Bmatrix} \theta_l - \theta_c & , \theta_l \geq \theta_c \\ \theta_l - \theta_c + 360° & , \theta_l < \theta_c \end{Bmatrix}, \quad 0° \leq \gamma < 360° \tag{8.19}$$

$$\omega = \begin{Bmatrix} \varphi_{12} - \theta_c & , \varphi_{12} \geq \theta_c \\ \varphi_{12} - \theta_c + 360° & , \varphi_{12} < \theta_c \end{Bmatrix}, \quad 0° \leq \omega < 360° \tag{8.20}$$

The modified experiments shall be referred to as Experiments 3a and 3b, which correspond to Experiments 2a and 2b, respectively.

We shall refer to the original fingerprint construct, which was proposed in Chapter 5, as **FC$_{180°}$**, since the angle attributes in this construct are limited to the range [0°, 180°). Similarly, we shall refer to the modified fingerprint construct, investigated in Experiments 3a and 3b, as **FC$_{360°}$**, since all the angles in this construct lie in the range [0, 360°).

Table 8.10 compares the FAR and FRR results obtained for **FC$_{360°}$** in Experiment 3a (i.e., *Single-Factor Authentication*) when the first set of matching thresholds from Table 8.5 was used (i.e., $\tau_l = 16$, $\tau_{\alpha\beta} = 14°$, $\tau_{loc} = 22$, $\tau_\omega = 16°$) with the FAR and FRR results obtained for **FC$_{180°}$** at the same thresholds from Table 8.6.

Table 8.10: Comparison of FAR and FRR for FC$_{360°}$ versus FC$_{180°}$ under Single-Factor Authentication, when $\tau_l = 16$, $\tau_{\alpha\beta} = 14°$, $\tau_{loc} = 22$, $\tau_\omega = 16°$.

	N = 3		*N* = 4		*N* = 5	
	FC$_{180°}$	**FC$_{360°}$**	**FC$_{180°}$**	**FC$_{360°}$**	**FC$_{180°}$**	**FC$_{360°}$**
FAR (%)	2.16	0.86	0.41	0.16	0.08	0.03
FRR (%)	2.41	2.61	3.09	3.46	3.96	4.10

Table 8.11 compares the FAR and FRR results obtained for **FC$_{360°}$** in Experiment 3a (i.e., *Single-Factor Authentication*) when the second set of matching thresholds from Table 8.5 was used (i.e., $\tau_l = 20$, $\tau_{\alpha\beta} = 18°$, $\tau_{loc} = 26$, $\tau_\omega = 23°$) with the FAR and FRR results obtained for **FC$_{180°}$** at the same thresholds from Table 8.7.

Table 8.11: Comparison of FAR and FRR for $FC_{360°}$ versus $FC_{180°}$ under Single-Factor Authentication, when $\tau_l = 20$, $\tau_{\alpha\beta} = 18°$, $\tau_{loc} = 26$, $\tau_{\omega} = 23°$.

	$N = 3$		$N = 4$		$N = 5$	
	$FC_{180°}$	$FC_{360°}$	$FC_{180°}$	$FC_{360°}$	$FC_{180°}$	$FC_{360°}$
FAR (%)	8.78	3.42	2.91	1.08	1.03	0.36
FRR (%)	1.16	1.50	1.70	2.17	2.58	2.63

Table 8.12 compares the FAR and FRR results obtained for $FC_{360°}$ in Experiment 3b (i.e., *Two-Factor Authentication*) when the first set of matching thresholds from Table 8.5 was used (i.e., $\tau_l = 16$, $\tau_{\alpha\beta} = 14°$, $\tau_{loc} = 22$, $\tau_{\omega} = 16°$) with the FAR and FRR results obtained for $FC_{180°}$ at the same thresholds from Table 8.8.

Table 8.12: Comparison of FAR and FRR for $FC_{360°}$ versus $FC_{180°}$ under Two-Factor Authentication, when $\tau_l = 16$, $\tau_{\alpha\beta} = 14°$, $\tau_{loc} = 22$, $\tau_{\omega} = 16°$.

	$N = 3$		$N = 4$		$N = 5$	
	$FC_{180°}$	$FC_{360°}$	$FC_{180°}$	$FC_{360°}$	$FC_{180°}$	$FC_{360°}$
FAR (%)	3.37×10^{-5}	1.29×10^{-5}	1.53×10^{-7}	5.75×10^{-8}	7.90×10^{-10}	2.93×10^{-10}
FRR (%)	2.41	2.61	3.09	3.46	3.96	4.10

Table 8.13 compares the FAR and FRR results obtained for $FC_{360°}$ in Experiment 3b (i.e., *Two-Factor Authentication*) when the second set of matching thresholds from Table 8.5 was used (i.e., $\tau_l = 20$, $\tau_{\alpha\beta} = 18°$, $\tau_{loc} = 26$, $\tau_{\omega} = 23°$) with the FAR and FRR results obtained for $FC_{180°}$ at the same thresholds from Table 8.9.

Table 8.13: Comparison of FAR and FRR for $FC_{360°}$ versus $FC_{180°}$ under Two-Factor Authentication, when $\tau_l = 20$, $\tau_{\alpha\beta} = 18°$, $\tau_{loc} = 26$, $\tau_{\omega} = 23°$.

	$N = 3$		$N = 4$		$N = 5$	
	$FC_{180°}$	$FC_{360°}$	$FC_{180°}$	$FC_{360°}$	$FC_{180°}$	$FC_{360°}$
FAR (%)	1.90×10^{-4}	6.57×10^{-5}	1.57×10^{-6}	5.25×10^{-7}	1.38×10^{-8}	4.48×10^{-9}
FRR (%)	1.16	1.50	1.70	2.17	2.58	2.63

In Tables 8.10 to 8.13, using $FC_{360°}$ has resulted in an FAR that is *at least* 2.5 times *smaller* than the FAR for the same Pattern size, N, when $FC_{180°}$ was used, with the FRR increasing *at most* by only about 1.3 times. Note that the differences in the FARs and FRRs between $FC_{360°}$ and $FC_{180°}$ are generally larger at the second (larger) set of matching

thresholds (i.e., in Tables 8.11 and 8.13) than the first (smaller) set of matching thresholds (i.e., in Tables 8.10 and 8.12).

These results indicate that, despite the theoretical claims on the negligible loss in recognition accuracy put forth in Sections 5.2.2 and 5.2.3, restricting a Pattern's angle attributes to the [0°, 180°) range does appear to affect its recognition accuracy in practice. The fact that the FAR for $\mathbf{FC}_{360°}$ was found to be less than *half* of the FAR obtained for $\mathbf{FC}_{180°}$ appears to be consistent with the fact that restricting a Pattern's angle attributes to the [0°, 180°) range instead of allowing them to lie in the [0°, 360°) range results in *two* possibilities for each α, β, y, and ω. We may thus draw the following conclusions from Experiments 3a and 3b:

- The discussion in Section 5.2.2 demonstrates that, theoretically, the ambiguity in the α and β attributes can be resolved in practice, provided that the corresponding Pattern connection lines are not collinear. Although it appears reasonable to conclude that collinear Pattern connection lines would be infrequent in practice, perhaps they are more common than initially anticipated. In particular, due to the use of matching thresholds in practice, *almost* collinear connection lines would pass for *truly* collinear connection lines. This may, therefore, be a contributing factor towards the lower recognition accuracy of $\mathbf{FC}_{180°}$ compared to that of $\mathbf{FC}_{360°}$. A dedicated investigation into this collinearity is recommended as part of the future work on our proposed fingerprint construct.
- The discussion in Section 5.2.3 concluded that the ambiguity in the y and ω Pattern attributes cannot be resolved, but would only pose a problem for Pattern uniqueness if there happen to exist at least two N-node Patterns that are the exact mirror images of each other in terms of their vertical location coordinate and their orientation, respectively. While such an occurrence was deemed unlikely, it is possible that the use of matching thresholds in practice would make the matching less strict, such that Patterns that are *almost* mirror images of each other would be considered *true* mirror images. This may, therefore, contribute towards decreasing the recognition accuracy of our proposed fingerprint construct when the $\mathbf{FC}_{180°}$ version is used instead of the $\mathbf{FC}_{360°}$ version. It would be interesting to further investigate the frequency of occurrence of mirror image N-node Patterns at various matching thresholds as part of the future work in this research direction.

Overall, the results obtained from Experiments 3a and 3b suggest that the $\mathbf{FC}_{360°}$ version of our proposed fingerprint construct would be expected to produce Patterns that are more *unique* that those generated using the $\mathbf{FC}_{180°}$ version. Consequently, $\mathbf{FC}_{360°}$ may be expected

to provide better recognition accuracy than $\mathbf{FC}_{180°}$, resulting in a much lower FAR at the expense of a relatively insignificant increase in the FRR. We thus recommend adopting $\mathbf{FC}_{360°}$ in favour of $\mathbf{FC}_{180°}$ in practice.

Note that, although an $\mathbf{FC}_{180°}$ N-node Pattern would require a smaller amount of storage space than an $\mathbf{FC}_{360°}$ N-node Pattern of the same size, the difference is still insignificant compared to the amount of storage space saved by not storing an entire minutiae template. In Section 5.2.4, it was established that $4N + 5$ bytes are required to store an $\mathbf{FC}_{180°}$ N-node Pattern, compared to $6T$ required for a full minutiae template (where T denotes the total number of minutiae available in the entire minutiae template). To store an $\mathbf{FC}_{360°}$ N-node Pattern, we would require $6N + 6$ bytes, since we now need 2 bytes, instead of 1, to store each angle attribute. Table 8.14 compares the number of bytes needed to store an $\mathbf{FC}_{360°}$ N-node Pattern feature vector to the number of bytes required to store an $\mathbf{FC}_{180°}$ N-node Pattern from Table 5.1.

Table 8.14: Comparison of the estimated amount of storage space required for an $FC_{180°}$ *N*-node Pattern versus an $FC_{360°}$ *N*-node Pattern.

	Number of Bytes Needed to Store $FC_{180°}$ Feature Vector	Number of Bytes Needed to Store $FC_{360°}$ Feature Vector
$N = 3$	$(4 \times 3) + 5 = 17$	$(6 \times 3) + 6 = 24$
$N = 4$	$(4 \times 4) + 5 = 21$	$(6 \times 4) + 6 = 30$
$N = 5$	$(4 \times 5) + 5 = 25$	$(6 \times 5) + 6 = 36$

From Table 8.14, it is evident that $2N + 1$ extra bytes are required to store the feature vector of an $\mathbf{FC}_{360°}$ N-node Pattern that is the same size as an $\mathbf{FC}_{180°}$ N-node Pattern. However, considering that the analysis in Section 5.2.4 estimated that a minimum of 150 bytes and a maximum of 576 bytes would be required to store an entire minutiae template, we may conclude that this increase in the amount of storage space needed to store an $\mathbf{FC}_{360°}$ N-node Pattern feature vector is comparatively insignificant.

8.6 SUMMARY

This chapter presented experimental results on the recognition accuracy of our proposed fingerprint construct when applied to our cooperative-user fingerprint database.

We provided a detailed description of our experimental set-up, which may serve as an example of how our proposed fingerprint construct can be implemented in practice.

We investigated the effect that increasing the number of reference fingerprints, which are used during enrolment to establish the most reliable reference minutiae, has on the FRR and

FAR of our proposed fingerprint construct. The FRR was shown to decrease dramatically as the number of reference fingerprints increases, while the corresponding increase in the FAR was found to be comparatively marginal.

The FAR and FRR of our proposed fingerprint construct in the *Most Favourable Genuine User Authentication Scenario*, which was established in Chapter 7, were then investigated. It was shown that the recognition accuracy can be tuned to suit the requirements of different applications by selecting the most appropriate Pattern size along with a suitable set of matching thresholds. In particular, smaller Patterns, combined with larger matching thresholds, were found to be suitable for low-security applications, while larger Patterns, combined with smaller matching thresholds, were found to be more applicable to high-security applications. Unsurprisingly, *Two-Factor Authentication* was shown to provide much better recognition accuracy than *Single-Factor Authentication* in terms of its ability to discriminate between a genuine user and an impostor. It was thus recommended that *Two-Factor Authentication* be adopted in practice in favour of *Single-Factor Authentication*.

Finally, we proposed a modification to our fingerprint construct, which involves extending all of a Pattern's angle attributes from the [0°, 180°) range to the [0°, 360°) range. The modified fingerprint construct was thus named $\mathbf{FC}_{360°}$, and the original fingerprint construct, proposed in Chapter 5, was termed $\mathbf{FC}_{180°}$. Experimental results indicated that $\mathbf{FC}_{360°}$ effectively increases Pattern *uniqueness* (by at least 2.5 times), which in turn improves the recognition accuracy of the resulting *N*-node Patterns. We thus recommended adopting $\mathbf{FC}_{360°}$, instead of $\mathbf{FC}_{180°}$, in practice.

As a general note on the results presented in this chapter, it was mentioned that the associated experiments were conducted under the assumption that the user would not be able to see their fingerprint image prior to it being captured by the scanner. This would potentially necessitate multiple authentication attempts in order to capture the part of the fingerprint in which a user's reference Pattern is contained. In practice, however, we recommended incorporating a live scan of the user's finger on the scanner, such that the user can see what the scanner sees. This would ensure that only a single authentication attempt is required to capture the minutiae needed to construct the user's *N*-node Pattern (provided that a robust feature extractor is adopted).

Chapter 9

Performance Comparison

Chapter 8 provided encouraging evidence to support the deployment of our proposed fingerprint construct in cooperative-user scenarios in practice, in terms of the attainable recognition accuracy. A modified version of our fingerprint construct, termed $\mathbf{FC}_{360°}$, was proposed to improve Pattern uniqueness, and it was recommended that this version of the fingerprint construct be used in practice due to its superior recognition accuracy compared to the original version, termed $\mathbf{FC}_{180°}$. In this chapter, we present a comparison of the performance of $\mathbf{FC}_{360°}$ to the performance of other non-invertible fingerprint template protection schemes in the literature.

9.1 INTRODUCTION

Due to the relative infancy of the fingerprint template protection research field, there do not yet exist standardised methods of evaluating the proposed techniques. While the performance of a fingerprint template protection scheme is generally evaluated in a similar way to the performance of a standard fingerprint recognition algorithm (i.e., in terms of the FAR and FRR), there are inconsistencies in the fingerprint database used for testing, the adopted alignment and feature (e.g., minutiae, core) extraction methods, the matching algorithm, the thresholds and various parameters used for matching, etc. For this reason, unless one is directly carrying on the work of someone else and thus *precisely* mimics their experimental conditions, it is difficult to conduct a fair comparison between the performance of different fingerprint template protection algorithms. Nevertheless, it is common for such performance comparisons to be reported in the literature, even though it is generally accepted that the comparison will usually be unfair. For this reason, the aim of this chapter is to compare the performance of the improved version of our proposed fingerprint construct, $\mathbf{FC}_{360°}$, to the performance of other non-invertible fingerprint template protection schemes in the literature, and we make every effort to ensure that the comparison is as fair as possible. Note that, since

performance comparisons in the literature are generally made in terms of the EER[31], we adopt the same performance measure for our method in this chapter.

The remainder of this chapter begins by discussing the set-up of the experiment used to calculate the EER of $\mathbf{FC}_{360^\circ}$ in a way that would enable a fair comparison to other methods. The resulting EER is then compared to the reported EERs of other non-invertible fingerprint template protection schemes in the literature.

9.2 EXPERIMENTAL SET-UP

This section provides details on the set-up of the experiment used to compute the EER of the improved version of our proposed fingerprint construct, $\mathbf{FC}_{360^\circ}$, such that the resulting recognition accuracy is comparable to the performance of other non-invertible fingerprint template protection schemes in the literature. Section 9.2.1 describes the rationale behind the fingerprint database selected for this experiment, and the existing non-invertible fingerprint template protection schemes that were tested on the same database are listed. Section 9.2.2 explains the procedure used to extract the core and minutiae from each fingerprint in the database, and the list of non-invertible fingerprint template protection schemes from Section 9.2.1 is filtered to include only those techniques that employed the same fingerprint feature extractor. Section 9.2.3 then discusses the methodology adopted to calculate the EER of $\mathbf{FC}_{360^\circ}$ on the fingerprint database selected in Section 9.2.1, using the feature extraction procedure established in Section 9.2.2.

9.2.1 Selected Fingerprint Database

In order to make the performance comparison as fair as possible, it is best to employ the same fingerprint database for testing the recognition accuracy of the methods being compared. Since the cooperative-user fingerprint database used to evaluate the recognition accuracy of our proposed fingerprint construct in Chapter 8 is proprietary, we cannot claim that the results from Chapter 8 can be used in a fair comparison to the recognition accuracy of other non-invertible fingerprint template protection schemes. For this reason, we do not use our cooperative-user fingerprint database for the experiment conducted in this chapter.

Since there is no consensus on the use of a single fingerprint database in the literature, we decided to select the database that was employed in the testing of the majority of existing non-invertible fingerprint template protection schemes. Most of the non-invertible fingerprint

[31] Equal Error Rate: $EER = \frac{FAR+FRR}{2}$

template protection schemes reviewed in Chapter 4 used the FVC2002 DB2 public fingerprint database [219], and these include [111, 112, 133-137, 139, 140, 142, 144-146, 149-151]; therefore, we opted for the same database to evaluate the recognition accuracy of $\mathbf{FC}_{360°}$ for comparison purposes.

The FVC2002 DB2 database is split into two separate databases: DB2_A consists of 8 samples of the same fingerprint for each of 100 different people, and DB2_B contains 8 samples of the same fingerprint for each of 10 different people. The idea is to use DB2_B for parameter tuning and DB2_A for evaluating the recognition accuracy of a fingerprint recognition system using the parameters selected with the help of DB2_B.

All of the fingerprint images in FVC2002 DB2 were acquired using the FX2000 optical sensor by Biometrika [220]. The resolution of each image is 296 × 560 pixels, 500 dpi. An important point to note about the FVC2002 DB2 database is that it was constructed with the deliberate aim of testing the performance of fingerprint recognition algorithms designed to operate in *un*cooperative user scenarios. For this reason, the database generally consists of fingerprint samples that would be acquired from *un*cooperative users of a fingerprint recognition system in practice. For example, Figure 9.1 illustrates 8 different samples of the same fingerprint acquired from person 3 in DB2_A.

Figure 9.1: The 8 fingerprint samples from person 3 in FVC2002 DB2_A: Illustrating user inconsistency in placing their finger on the scanner.

While samples 1, 2, 6, 7, and 8 may be said to have been captured when the person placed their finger on the scanner in a fairly cooperative manner, sample 3 represents an exaggerated rotation of the finger on the scanner, and samples 4 and 5 represent exaggerated translations of the finger on the scanner surface. Our investigation into the consistency of cooperative users in placing their fingers on the fingerprint scanner (described in Chapter 6), showed that extreme rotations and translations of the finger on the scanner surface is very unlikely to be observed among cooperative users. Consequently, we may conclude that fingerprint samples 3, 4, and 5 in Figure 9.1 are not representative of the types of fingerprint images that would be acquired from cooperative users in practice. Since, as stated in Chapter 5, our proposed fingerprint construct is intended for deployment in a cooperative-user scenario, we may expect that the FVC2002 DB2 database will not fairly reflect the attainable recognition

accuracy of $\mathbf{FC}_{360°}$ in its target application scenario in practice; indeed, this is why our evaluation of the performance of our proposed fingerprint construct in Chapter 8 was based on our cooperative-user fingerprint database. Nevertheless, in this chapter we shall apply $\mathbf{FC}_{360°}$ to FVC2002 DB2 for the purpose of comparing its performance to the performance of other non-invertible fingerprint template protection schemes evaluated on the same database; however, we ask that the reader keep in mind that the resulting EER is expected to be worse than the EER attainable in practice when $\mathbf{FC}_{360°}$ is used in its intended application scenario.

Besides user inconsistency in placing their finger on the fingerprint scanner, another important point to note regarding FVC2002 DB2 is that the acquired fingerprint samples are of varying quality. For example, while the quality of the fingerprints in Figure 9.1 is generally pretty good (i.e., it is fairly easy to differentiate between the ridges and valleys in each fingerprint image), the quality of sample 8 may be said to be visibly worse than the quality of the other seven samples (i.e., many of the fingerprint ridges are broken). A more striking example of the varying quality between different samples of the same fingerprint is illustrated in Figure 9.2, which represents the 8 fingerprint samples from person 93 in FVC2002 DB2_A.

Figure 9.2: The 8 fingerprint samples from person 93 in FVC2002 DB2_A: Illustrating varying fingerprint quality.

Samples 1, 2, 3, 4, 5, and 8 in Figure 9.2 represent fingerprint images that would be acquired from a person with dry skin. While dry skin is a natural phenomenon and is thus likely to be encountered in any fingerprint recognition system, an important finding during the construction of our cooperative-user fingerprint database for the investigation in Chapter 6 was that simply asking a person to rub their finger on the side of their nose or onto the forehead is usually sufficient to deal with dry skin and thus significantly improve the quality of the acquired fingerprint image. Moreover, we found that providing a simple, visual explanation of what a good quality fingerprint image should look like was generally sufficient to encourage cooperative users to control this quality on their own as much as possible during fingerprint image acquisition (for example, by rubbing dry fingers on the side of their nose or onto their forehead to apply some grease to them, and dabbing sweaty fingers onto a piece of clothing to remove the excess moisture). These findings suggest that fingerprint images of

the type represented by samples 1, 2, 3, 4, 5, and 8 in Figure 9.2 should not be acquired in a cooperative-user scenario in practice. This is particularly true for automated civilian fingerprint recognition applications, which would generally have an in-built quality-checker to ensure that the acquired fingerprint images are of sufficiently good quality for further processing – if the quality of the acquired fingerprint image is poor, the user would be asked to re-scan the finger and/or the acquired fingerprint image would be submitted to an image enhancement algorithm. Since our proposed fingerprint construct is intended for deployment in a cooperative-user civilian fingerprint recognition application, it is fair to assume that $\mathbf{FC}_{360^\circ}$ would infrequently need to deal with fingerprint images of very bad and/or extremely variable quality[32]. This provides another reason (i.e., in conjunction with the user inconsistency illustrated in Figure 9.1) to conclude that the EER obtained for our proposed fingerprint construct on FVC2002 DB2 will be worse than what may be expected when $\mathbf{FC}_{360^\circ}$ is applied in its intended application scenario in practice.

9.2.2 Minutiae and Core Extraction Procedure

The recognition accuracy of any non-invertible fingerprint template protection scheme primarily relies upon the accuracy of the feature extractor used to extract the necessary fingerprint features. For example, if the employed feature extractor fails to detect a large number of minutiae in the underlying fingerprint, then, regardless of how good the actual fingerprint template protection mechanism is, its recognition accuracy will be affected by the errors inherent in the adopted feature extractor. For this reason, a comparison of the performance of different fingerprint template protection schemes cannot be considered fair unless the *same* feature extractor is employed.

Recall, from Chapter 5, that our proposed fingerprint construct generates an N-node Pattern based on a subset of N fingerprint minutiae and the fingerprint's core point. The first step in evaluating the recognition accuracy of $\mathbf{FC}_{360^\circ}$ on FVC2002 DB2 is thus the extraction of the minutiae and core point from each fingerprint in FVC2002 DB2. Since the majority of non-invertible fingerprint template protection schemes evaluated on the FVC2002 DB2 database adopted the feature extractor provided by Neurotechnology's VeriFinger SDK (e.g., [217]), we used the same software to extract the minutiae and core points for $\mathbf{FC}_{360^\circ}$. Note that, although different researchers adopt different versions of this software for their

[32] Note that there are people for whom the quality of their fingerprints is naturally bad; for example, people with chronically dry skin, wrinkly skin, or fingers with many cuts and bruises. For such people, special image processing algorithms would need to be applied to enhance the quality of their fingerprints as much as possible for further processing.

performance evaluations, in this chapter we shall nevertheless compare the performance of our proposed fingerprint construct to the reported performance of all non-invertible fingerprint template protection schemes that employ FVC2002 DB2 along with *any* version of the VeriFinger SDK, which includes [111, 112, 133-136, 139, 145, 149]. This is because, otherwise, we would not be able to compare to any of the previous techniques, since we adopt the latest VeriFinger version, version 6.7, to extract the minutiae and core points for our proposed fingerprint construct.

It must be noted that, while the VeriFinger software has commendable recognition accuracy, we found that its feature extractor is not error-free. Concerning the reliability of the minutiae extractor, we discovered that, sometimes, seemingly obvious minutiae are undetected, and, at other times, spurious minutiae (i.e., parts of the fingerprint that are not actually minutiae) are falsely detected. For example, Figure 9.3 shows the minutiae detected for fingerprint image 14_4 in FVC2002 DB2_A. Notice how there is a large number of spurious minutiae (marked by red dots) detected at the top of the image, in an area where the actual fingerprint is not even located. Furthermore, notice that there are three fairly obvious minutiae that have not been detected (circled in green).

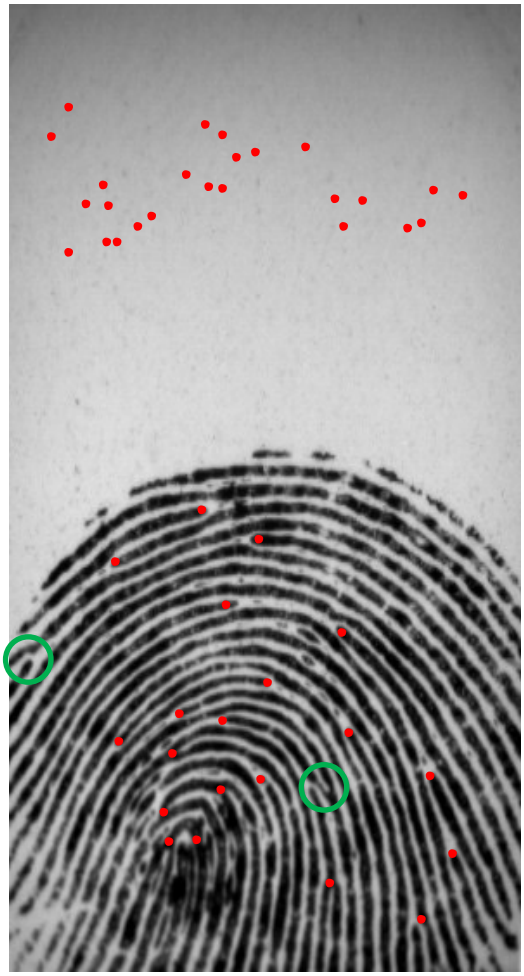


Figure 9.3: Minutiae detected (marked by red dots) and missed (circled in green) for image 14_4 in FVC2002 DB2_A, when VeriFinger 6.7 is used for feature extraction.

Figure 9.3 is possibly one of the worst examples of VeriFinger's minutiae extraction accuracy; otherwise, the minutiae extractor is fairly good. Assuming that the evaluations of [111, 112, 133-136, 139, 145, 149] did not manually correct any errors in the minutiae detection, then it is fair to assume that, in this respect, any errors in the minutiae extraction will have an effect on the recognition accuracy of all the non-invertible fingerprint template protection schemes being compared in this chapter. Thus, the comparison may be considered fair despite the errors.

Considering VeriFinger's core detection algorithm, a manual check revealed that the core point was often determined in inconsistent locations across different samples of the same person's fingerprint, and, sometimes, the core point was not detected at all. To decide on the fairest approach to this issue when evaluating the recognition accuracy of $\mathbf{FC}_{360°}$, we consulted the three other non-invertible fingerprint template protection schemes out of [111, 112, 133-136, 139, 145, 149] that used the core point in their proposed method: [112, 136, 145]. Of these methods, [145] used a different core extraction algorithm (i.e., not VeriFinger), which means that we must exclude [145] from the performance comparison in this chapter. The other two methods, [112, 136], manually corrected any faulty cores determined by VeriFinger; therefore, we adopted the same approach. The manual correction was done using the same procedure as that applied for correcting core points on fingerprints in our cooperative-user fingerprint database, which is described in Section 8.2.1.

9.2.3 Methodology to Calculate EER of $\mathbf{FC}_{360°}$

To calculate the EER of the $\mathbf{FC}_{360°}$ version of our proposed fingerprint construct on the FVC2002 DB2 database, it was first necessary to establish a set of sensible matching thresholds. Recall, from Section 5.2.5, that four thresholds are used to determine whether or not two N-node Patterns can be considered to match: τ_l (specifies the maximum allowed difference between the lengths of the Patterns' corresponding connection lines), $\tau_{\alpha\beta}$ (specifies the maximum allowed difference between the Patterns' corresponding α and β angle attributes), τ_{loc} (specifies the maximum allowed Euclidean distance between the Patterns' locations), and τ_{ω} (specifies the maximum allowed difference between the Patterns' orientations). Section 9.2.3.1 describes the threshold selection procedure adopted for FVC2002 DB2, and Section 9.2.3.2 outlines the methodology used to calculate the EER of $\mathbf{FC}_{360°}$ on FVC2002 DB2 using the thresholds established in Section 9.2.3.1.

9.2.3.1 Threshold Selection

FVC2002 DB2_B was used to determine the matching thresholds for calculating the EER of $\mathbf{FC}_{360°}$. The thresholds were determined in essentially the same way as for Experiment 1 on our cooperative-user fingerprint database in Section 8.3.1.

Firstly, for each person in DB2_B, we established which minutiae are present in all 8 of their fingerprint samples. Note that minutiae correspondences were established in the same way as for our cooperative-user fingerprint database, and this process is thoroughly explained in Section 8.2.2.

Then, the resulting, filtered set of minutiae in each of the 8 fingerprint samples was used to create every possible 3-node Pattern. Since we knew which minutiae in one fingerprint sample corresponded to which minutiae in the other 7 fingerprint samples, we could easily establish the corresponding 3-node Patterns across all 8 of a person's fingerprint samples.

Next, all the corresponding 3-node Patterns were compared to each other[33], and the differences between the corresponding Pattern attributes in each comparison for every person were used to plot a distribution. Table 9.1 summarises the maximum values and the 99th percentiles of the resulting distribution for each Pattern attribute.

Table 9.1: The maximum and 99th percentile of Pattern attribute differences.

	l Differences	α and β Differences (combined)	(x, y) Differences (Euclidean distance)	ω Differences
Maximum	444.24	179.56°	208.82	178.99°
99th Percentile	27.45	17.79°	37.57	46.19°

An important observation from Table 9.1 is that the maximum values of the Pattern attribute distributions are very large, which is especially evident for the angle attributes (i.e., α, β, and ω). For example, a difference of 179.56° between two *corresponding* α or β attributes is unreasonable, since such a large difference should only be possible for two α or β attributes that do *not* correspond; similarly for the orientation difference of 178.99°. These unreasonably large maximums suggest that there were errors in the minutiae extraction and/or core extraction process[34], or that there were errors in the minutiae matching algorithm used to establish the corresponding minutiae, or that there were errors in both of these modules.

[33] See Section 5.2.5 for the Pattern matching procedure.

[34] Since the cores were manually verified and corrected if necessary, it is unlikely that there were core detection errors capable of producing such large differences between *corresponding* Patterns. Nevertheless, it is possible that some small errors crept into the core extraction process.

These observations confirm our earlier reflections in Section 9.2.2 on VeriFinger's feature extraction errors.

Since the maximum values in Table 9.1 are unreasonably large, we decided not to use them in the establishment of the matching thresholds, since they would be likely to result in an impractically high FAR. For this reason, the matching thresholds were based on the 99th percentiles from Table 9.1, which, although also perhaps a little large (e.g., an orientation difference of 46.19°), nevertheless seem reasonable. The resulting matching thresholds are summarised in Table 9.2.

Table 9.2: Attribute-specific thresholds based on the 99th percentiles from Table 9.1.

Pattern Attribute	Matching Threshold
l	$\tau_l = \lceil 27.45 \rceil = 28$
α and β	$\tau_{\alpha\beta} = \lceil 17.79° \rceil = 18°$
(x, y)	$\tau_{loc} = \lceil 37.57 \rceil = 38$
ω	$\tau_{\omega} = \lceil 46.19° \rceil = 47°$

Note that, as in the evaluation of the performance of our proposed fingerprint construct on our cooperative-user fingerprint database in Chapter 8, the threshold selection algorithm adopted in this experiment is not guaranteed to result in the *optimal* recognition accuracy attainable by $\mathbf{FC}_{360°}$. This is because, as explained in Chapter 8, different fingerprint recognition applications would have different performance specifications, which means that it is impossible to define the word *optimal* universally. Furthermore, part of our future work includes the development of a robust threshold selection algorithm, which is able to determine the optimal balance between the four matching thresholds. Nevertheless, the matching thresholds in Table 9.2 would be expected to provide a reasonable indication of the EER of $\mathbf{FC}_{360°}$ on the FVC2002 DB2 database, and any optimisation of the matching thresholds would only improve the EER.

9.2.3.2 EER Computation

To calculate the EER of $\mathbf{FC}_{360°}$, the FVC2002 DB2_A fingerprint database was used. The following experimental procedure was adopted:

1. The first fingerprint sample of each person in DB2_A was used as that person's reference fingerprint, and their second sample was used as the query (test) fingerprint. This is in line with the experimental procedure adopted in [111, 133-135, 149].
2. Any minutiae that were physically captured in a person's reference fingerprint sample, but not physically captured in their query fingerprint sample, were excluded from that

person's reference minutiae set. This is because one of the benefits of our proposed fingerprint construct, when operating in the recommended *Two-Factor Authentication* scenario, is that the user *knows* where in their fingerprint their reference Pattern is located. Consequently, assuming that a live scan of the user's fingerprint is available at the time of authentication (as recommended in Section 8.4.3), the user could ensure that they capture the necessary portion of their fingerprint (i.e., the portion that contains their reference Pattern). Note that we did *not* correct cases where spurious minutiae were detected in a person's reference fingerprint, or cases where a true minutia was detected in the reference fingerprint but missed out in the query fingerprint. This is because, while the user can control how they place their finger on the scanner, they cannot control what the feature extractor does with their captured fingerprint.

3. From each person's reference minutiae set, 100 reference N-node Patterns for each value of $N = \{3, 4, 5\}$ were randomly generated, as per the procedure outlined in Section 8.2.2.
4. To calculate the FRR, each reference Pattern was searched for in the *same* person's query fingerprint sample to simulate an authentication attempt by a *genuine user*. If a reference Pattern could *not* find a match in the query fingerprint, this was considered as a False Reject for that particular Pattern. Note that a *match* was found if there existed an N-node Pattern whose attributes were similar enough to the attributes of the reference N-node Pattern, where the similarity was determined according to the thresholds specified in Table 9.2. The total number of genuine comparisons for each N was thus 10,000 (i.e. 100 reference Patterns per person × 100 reference people × 1 genuine attempt per person). Therefore, the FRR for each N was calculated using Equation (8.13).
5. To calculate the FAR under *Single-Factor Authentication* (which is equivalent to *Two-Factor Authentication* in the scenario where an impostor knows a genuine user's reference Pattern), each reference Pattern was searched for in every *other* person's query fingerprint sample to simulate an authentication attempt by an *impostor*. If a reference Pattern *found at least one* match in the query fingerprint, this was considered as a False Accept for that particular Pattern. Note that a *match* was found if there existed an N-node Pattern whose attributes were similar enough to the attributes of the reference N-node Pattern, where the similarity was determined according to the thresholds specified in Table 9.2. The total number of impostor comparisons for each N was thus 990,000 (i.e., 100 reference Patterns per person × 100 reference people × 99 impostor attempts per person). The FAR for each N was thus calculated using Equation (8.14).
6. To calculate the FAR under *Two-Factor Authentication*, we repeated Step 5, except this time we counted the *total number of matches* found in each impostor's fingerprint as

opposed to looking for *at least one* match. The FAR in this scenario was thus calculated using Equations (8.15) and (8.16), where, for this experiment, $P_{success}^{MAX} = P_{success}$ since each impostor now gets only *one* authentication attempt as opposed to the 3 authentication attempts allowed in Experiment 2b in Section 8.4.3.

7. The EER for each $N = \{3, 4, 5\}$ was then calculated using Equation (9.1):

$$EER = \frac{FAR + FRR}{2} \tag{9.1}$$

This experiment was run 3 times and the average EER across the 3 trials was recorded.

Note the following two points regarding the experimental procedure described above:

- Although it was recommended that *Two-Factor Authentication* be used in practice in favour of *Single-Factor Authentication*, Step 5 still calculated the FAR under *Single-Factor Authentication*. This is because *Single-Factor Authentication* pertains to *Two-Factor Authentication* in the worst-case scenario, in which it is assumed that an impostor *knows* a genuine user's reference *N*-node Pattern (in this case, the performance drops to the recognition accuracy attainable under *Single-Factor Authentication*). The reason this step was included in this experiment is because, as mentioned in Section 4.5.4, it is common for two-factor fingerprint template protection schemes to be compared to other methods in the scenario in which the user-specific external factor (in our case, the knowledge of the genuine user's reference Pattern) is stolen or known by an attacker.
- While our investigation in Chapter 7 determined that allowing a user to have multiple authentication attempts would improve the likelihood of a genuine user being accepted, other non-invertible fingerprint template protection schemes in the literature have essentially assumed a single authentication attempt per person[35]. For this reason, in this experiment it is assumed that each user is allowed only *one* authentication attempt, such that the resulting EER is comparable to other methods' EERs.

[35] This is not explicitly stated, but is implied by the way in which the experiments were run.

9.3 EXPERIMENTAL RESULTS AND COMPARISON TO OTHER TECHNIQUES

This section presents the EER obtained for $\mathbf{FC}_{360^\circ}$ at $N = \{3, 4, 5\}$, and the performance of our proposed fingerprint construct is compared to the reported performance of other non-invertible fingerprint template protection schemes in the literature.

Table 9.3 summarizes the EERs obtained for $\mathbf{FC}_{360^\circ}$ at $N = \{3, 4, 5\}$ in the *Normal Scenario* and the *Stolen-token Scenario*. For our method, the *Normal Scenario* refers to *Two-Factor Authentication* in the case where an attacker does not know a genuine user's reference *N*-node Pattern. We use the term "Normal Scenario" to comply with terminology commonly used in the literature to denote the scenario where the transform in a non-invertible fingerprint transform is unknown to an attacker. On the other hand, the *Stolen-token Scenario* refers to either *Single-Factor Authentication*, or *Two-Factor Authentication* in the case where an attacker knows a genuine user's reference *N*-node Pattern. We use the term "Stolen-token Scenario" to comply with terminology commonly used in the literature to denote the scenario where the transform in a non-invertible fingerprint transform is stolen by, or known to, an attacker.

Table 9.3: EER obtained for FC_{360° for each $N = \{3, 4, 5\}$ in the Normal and Stolen-token Scenarios.

	$N = 3$	$N = 4$	$N = 5$
Normal Scenario EER	1.93%	2.28%	2.58%
Stolen-token Scenario EER	4.00%	2.92%	2.80%

Table 9.4 compares the EERs of $\mathbf{FC}_{360^\circ}$ from Table 9.3 with the EERs reported in [111, 112, 133-136, 139, 149][36].

Prior to drawing any conclusions from the results in Table 9.4, note that:

- All the non-invertible fingerprint template protection schemes in Table 9.4 to which $\mathbf{FC}_{360^\circ}$ is compared use the full minutiae template in generating the protected fingerprint template. In comparison, $\mathbf{FC}_{360^\circ}$ uses only a small portion of the entire minutiae template.
- Even though all of the non-invertible fingerprint template protection schemes in Table 9.4 were tested on the same fingerprint database, the portion of the database used was not the same in all cases. In particular, [111, 133-135, 149] used the first two fingerprint samples from each person in FVC2002 DB2_A, where the first sample served as the reference

[36] These are the non-invertible fingerprint template protection schemes reviewed in Chapter 4, which employ FVC2002 DB2_A for testing and VeriFinger for feature extraction.

fingerprint and the second sample was the query (test) fingerprint. This is the approach adopted for evaluating the EER of **$FC_{360°}$** in this chapter also (see Section 9.2.3.2). Alternatively, the entire DB2_A fingerprint database was used in the calculation of the EER for the fingerprint template protection scheme proposed in [139]. Finally, [112, 136] employed the first seven of each person's fingerprint samples in DB2_A to establish the person's most reliable minutiae[37] and their 8th sample served as the query fingerprint. Due to this inconsistency, we shall refrain from drawing definitive conclusions on how the fingerprint template protection schemes in Table 9.4 rank based on their reported EERs. Nevertheless, some important observations may be drawn from these results.

Table 9.4: A comparison of the EERs obtained for $FC_{360°}$ at different Pattern sizes to the reported EERs of other non-invertible fingerprint template protection schemes in the literature, both in the Normal and the Stolen-token Scenarios.

Non-Invertible Fingerprint Template Protection Scheme	Reported EER
NORMAL SCENARIO:	
$FC_{360°}$ at $N = 3$	1.93%
$FC_{360°}$ at $N = 4$	2.28%
$FC_{360°}$ at $N = 5$	2.58%
[111]	4.04%
[136]	2.21%
[112]	5.52%
[135]	2.23%
STOLEN-TOKEN SCENARIO:	
$FC_{360°}$ at $N = 3$	4.00%
$FC_{360°}$ at $N = 4$	2.92%
$FC_{360°}$ at $N = 5$	2.80%
[133]	2.50%
[139]	6.94%
[134]	6.00%
[149]	2.30%

From Table 9.4, we may draw the following conclusions:

- The EER for **$FC_{360°}$** *increases* as N increases in the Normal Scenario but *decreases* with an increase in N in the Stolen-token Scenario. This is because the FAR in the Normal Scenario is extremely low[38], so the EER is mainly controlled by the FRR. Since the FRR

[37] This was done in a different way to that employed for our proposed fingerprint construct.

[38] See the results in Tables 8.12 and 8.13 for an example.

tends to increase with an increase in *N*, this causes a corresponding increase in the EER. In the Stolen-token Scenario, however, the FAR and FRR are closer in magnitude than the FAR in FRR in the Normal Scenario[39]. Since the FAR tends to decrease more rapidly than the FRR tends to increase with an increase in *N*, the FAR has more influence on the EER. Consequently, the EER tends to decrease with an increase in *N*. Notice that, both in the Normal and Stolen-token Scenarios, the EERs across the different *N*s get closer together as *N* increases. This is because the FAR gets smaller as *N* increases and the EER starts to be more influenced by the FRR. Since the FRR increase tends to be less significant than the FAR decrease[40], the EERs get closer together with an increase in *N*.

- The EER for $\mathbf{FC}_{360°}$ at all three Pattern sizes in the Normal Scenario is better than the EER reported in [111] and [112], and comparable to the EER reported in [136] and [135]. The fact that $\mathbf{FC}_{360°}$ appears to perform better than the method in [112] is especially encouraging considering that [112] employed the first 7 of each person's fingerprint samples to establish that person's most reliable reference minutiae, while reliable minutiae filtering was not adopted in the approach used to calculate the EER of $\mathbf{FC}_{360°}$ in this section[41].
- The EER for $\mathbf{FC}_{360°}$ at all three Pattern sizes in the Stolen-token Scenario is better than the EER of 6.94% reported in [139]; however, this is a bit difficult to comment on, since the method in [139] was evaluated on the whole database, while $\mathbf{FC}_{360°}$ was tested using the first two samples of each person's fingerprint in the database. Nevertheless, our results suggest that the two methods may be comparable in terms of their recognition accuracy (i.e., neither seems considerably better or worse than the other).
- The EER for $\mathbf{FC}_{360°}$ at all three Pattern sizes in the Stolen-token Scenario is better than the EER of 6.00% reported in [134]. This may be due, in part, to the fact that users of the new fingerprint construct are more aware of how the authentication process works, so they know what area of the fingerprint they must capture each time. Furthermore, $\mathbf{FC}_{360°}$ is based on only a single, sparse Pattern per user, whilst [134] (and, indeed, essentially all other non-invertible fingerprint template protection schemes in the literature), are based

[39] See the results in Tables 8.10, 8.11, 8.12, and 8.13 for an example of this trend.

[40] See the discussion following Table 8.7 in Section 8.4.2 (p.163) for evidence of this trend.

[41] In Section 9.2.3.2, it was noted that only the minutiae occurring in parts of the fingerprint that were physically captured in both the reference and query fingerprints were considered for each person's reference minutiae set. We did not, however, filter out reliable minutiae as we did for the performance evaluation on our cooperative-user fingerprint database in Chapter 8. Had we used the latter approach, a lot of the spurious minutiae detected by VeriFinger's feature extractor would have been eliminated, and we may expect that, in this scenario, the EER of our proposed fingerprint construct would have been better.

on the entire minutiae template. In that sense, provided that the right fingerprint area is captured, that a robust feature extractor is used, and that the matching thresholds are appropriately set, the new fingerprint construct may make it easier for a genuine user to be authenticated than methods with more stringent matching requirements. Additionally, as shown in the analysis in Chapters 5 and 8, despite the sparsity of the *N*-node Patterns generated by our proposed fingerprint construct, our method is effectively able to discriminate between a genuine user and an impostor. These two observations combined suggest that the EER of our proposed fingerprint construct may be expected to be favourable in practice.

- The EER for $\mathbf{FC}_{360^\circ}$ at all three Pattern sizes in the Stolen-token Scenario is better than the EER reported in [111] and [112] in the Normal Scenario. This is encouraging considering that the EER in [111, 112] was calculated in the scenario where an attacker does *not* have access to the genuine user's transform, while we assumed that the attacker *knows* a genuine user's reference Pattern. So, the EER calculated for $\mathbf{FC}_{360^\circ}$ in the worst-case scenario appears to be better than the EER calculated for [111, 112] in the normal-case scenario.
- In the Stolen-token Scenario, the EER for $\mathbf{FC}_{360^\circ}$ at $N = 3$ is worse than the EER reported for [133] and [149], but the performances of these methods become comparable as N is increased to 4 or 5.

Overall, the results presented in Table 9.4 provide evidence to suggest that, despite using only a small portion of the full minutiae template, $\mathbf{FC}_{360^\circ}$ is able to perform reasonably well in comparison to non-invertible fingerprint template protection schemes that employ the entire minutiae template. This is in spite of the fact that the EER was calculated using un-optimised matching thresholds, and that the experiment was conducted on a database that is unrepresentative of the types of fingerprint images with which our proposed fingerprint construct is likely to work in practice. Furthermore, the EER reported for our proposed fingerprint construct in this chapter is based on only one feature extractor (i.e., VeriFinger). As stated in Section 9.2.2, the VeriFinger software has been observed to make errors in its feature extraction; indeed, spurious minutiae or minutiae that were not detected despite being captured by the user would affect the recognition accuracy of our proposed fingerprint construct. Using a more robust feature extractor may thus be expected to significantly improve the EER attainable by $\mathbf{FC}_{360^\circ}$ in practice.

9.4 SUMMARY

This chapter compared the recognition accuracy of our improved fingerprint construct from Section 8.5, $\mathbf{FC}_{360^\circ}$, with the recognition accuracy reported for other non-invertible fingerprint template protection schemes in the literature.

In order to ensure that the performance (in terms of the EER) of $\mathbf{FC}_{360^\circ}$ is fairly comparable to the EERs reported for other non-invertible fingerprint template protection schemes, it was necessary to perform the experiment on the same fingerprint database, using the same fingerprint feature extractor. Since FVC2002 DB2 was found to be the most commonly adopted database in the evaluation of the non-invertible fingerprint template protection schemes reviewed in Chapter 4, we opted to use the same database in the calculation of the EER for $\mathbf{FC}_{360^\circ}$. Of those non-invertible fingerprint template protection schemes that were evaluated on FVC2002 DB2, the most commonly used fingerprint feature extractor was the commercial VeriFinger SDK; the same software was thus applied in the extraction of minutiae and core points from the fingerprints in FVC2002 DB2 for the purpose of evaluating the EER of $\mathbf{FC}_{360}$.

A comparison of the performance of $\mathbf{FC}_{360^\circ}$ to that of other non-invertible fingerprint template protection schemes, all of which use the full minutiae template, suggested that $\mathbf{FC}_{360^\circ}$ may be considered a significant competitor despite using only a small portion of the full minutiae template. This observation is especially encouraging considering that the EER for $\mathbf{FC}_{360^\circ}$ was evaluated using un-optimised matching thresholds and that the experiment was conducted on a database that does not reflect the types of fingerprint images with which our proposed fingerprint construct would be expected to work in practice.

Chapter 10

Non-invertibility of $FC_{360°}$

The analysis in Chapters 5 through to 9 provides encouraging evidence to support the idea that our new fingerprint construct complies satisfactorily with the *Performance* requirement for an ideal fingerprint template protection scheme. While Experiment 2 in Chapter 5 showed that the new fingerprint construct may slightly degrade the recognition accuracy attainable using full, unprotected minutiae templates, the performance evaluations in Chapters 5 to 9 nevertheless suggest that our proposed fingerprint construct would be capable of effectively discriminating between a genuine user and an impostor in a cooperative-user scenario in practice, particularly when the improved version, $\mathbf{FC_{360°}}$, is adopted. This chapter presents analysis to prove that $\mathbf{FC_{360°}}$, which was proposed in Chapter 8, also satisfies the *non-invertibility* characteristic of an ideal fingerprint template protection scheme.

10.1 INTRODUCTION

Section 4.5.1 discussed the methods that are commonly employed in evaluating the *non-invertibility* of a fingerprint template protection scheme in the literature. It was concluded that, although the *non-invertibility* analysis has not yet been standardised, there appear to be two main approaches. In the first approach, it is simply proven that, since the employed transform is a many-to-one mapping, the reverse process is *non-invertible*; however, there is no mention of *how* non-invertible the transform is. The second approach involves quantifying the *non-invertibility* in terms of the number of guesses required to recover the unprotected template from its protected version via brute force. A less common but, in our opinion, more intuitive, approach is to quantify *non-invertibility* by estimating the proportion of the original minutiae template that remains unrecoverable in the event that the protected template is compromised.

Perhaps the reason that the analysis of *non-invertibility* is not agreed upon in the literature is because the notion of *non-invertibility* is difficult to define. For example, are we interested in whether *any* of the original fingerprint template is recoverable from the protected template,

or are we more concerned with whether the *entire* original template can be determined from its protected counterpart? Since it seems rather naïve to suggest that *no* information about the original fingerprint template can be leaked by the protected template, we believe that it is more important to consider the difficulty of recovering the *entire* original template from its protected counterpart. This is because *any* fingerprint template protection scheme is bound to leak some information about the original template (regardless of whether or not this is immediately obvious); otherwise, the protected template would be completely unrelated to the underlying fingerprint, thereby questioning the point of using the fingerprint in the first place. Considering *non-invertibility* in terms of the difficulty of recovering the *entire* original template, however, allows us to gain some appreciation of essentially how much of a fingerprint's individuality (or uniqueness) is compromised if the protected template is compromised. Consequently, this provides insight into whether that particular fingerprint can continue to be used for recognition purposes, in terms of the remaining fingerprint *uniqueness*.

This chapter evaluates the *non-invertibility* of the improved version of our fingerprint construct, $\mathbf{FC}_{360^\circ}$. We begin by investigating the amount of information about the original minutiae template that is leaked by the feature vector of an $\mathbf{FC}_{360^\circ}$ *N*-node Pattern. The outcome of this investigation is then used to evaluate the *non-invertibility* of our proposed fingerprint construct via two methods. The first method considers the proportion of a fingerprint's minutiae template that remains unrecoverable despite the information leaked by an *N*-node Pattern's feature vector. The second method estimates the complexity of recovering a fingerprint's entire minutiae template from this information leak via a brute force approach.

10.2 INFORMATION LEAKED BY AN $\mathbf{FC}_{360^\circ}$ *N*-NODE PATTERN

This section investigates the amount of information about a fingerprint's minutiae template that is leaked by an $\mathbf{FC}_{360^\circ}$ *N*-node Pattern's feature vector. In particular, we are interested in determining whether it is possible to recover the Pattern's *N* constituent minutiae from the Pattern attributes stored in the feature vector. In Section 5.2.4, it was stated that the feature vector, *v*, of an *N*-node Pattern has the following form:

$$v = [l_{12}, \alpha_{12}, \beta_{12}, l_{23}, \alpha_{23}, \beta_{23}, \dots, l_{N1}, \alpha_{N1}, \beta_{N1}, x, y, \omega]$$

This feature vector would have been constructed from N minutiae, which shall be denoted by $m_1, m_2, \ldots, m_N$, and the corresponding fingerprint's core point, which shall be denoted by c. The attributes of these features that were used in the construction of the N-node Pattern (which is represented by v) are their locations, (x, y), expressed in terms of the (column, row) indices in the underlying fingerprint image, and their orientations, θ, which lie in the range $[0°, 360°)$ and increase in the clockwise direction from the horizontal in the underlying fingerprint image. Let us say that:

$$m_1 = (x_1, y_1, \theta_1)$$
$$m_2 = (x_2, y_2, \theta_2)$$
$$\vdots$$
$$m_N = (x_N, y_N, \theta_N)$$
$$c = (x_c, y_c, \theta_c)$$

In this section, we are interested in recovering $m_1, m_2, \ldots, m_N$ from v.

We begin by considering all the equations used to obtain v from $m_1, m_2, \ldots, m_N$, and c, after which we will establish whether it is possible to reverse the process to obtain $m_1, m_2, \ldots, m_N$. From Sections 5.2.2, 5.2.3, and 8.5, we know that the attributes of v for an $\mathbf{FC}_{360°}$ N-node Pattern are calculated using Equations (10.1) to (10.20):

$$l_{12} = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2} \tag{10.1}$$
$$l_{23} = \sqrt{(x_3 - x_2)^2 + (y_3 - y_2)^2} \tag{10.2}$$
$$\vdots$$
$$l_{N1} = \sqrt{(x_1 - x_N)^2 + (y_1 - y_N)^2} \tag{10.3}$$
$$\varphi_{12} = \tan^{-1}\left(\frac{y_2 - y_1}{x_2 - x_1}\right), \quad 0° \leq \varphi_{12} < 360° \tag{10.4}$$
$$\varphi_{23} = \tan^{-1}\left(\frac{y_3 - y_2}{x_3 - x_2}\right), \quad 0° \leq \varphi_{23} < 360° \tag{10.5}$$
$$\vdots$$
$$\varphi_{N1} = \tan^{-1}\left(\frac{y_1 - y_N}{x_1 - x_N}\right), \quad 0° \leq \varphi_{N1} < 360° \tag{10.6}$$
$$\alpha_{12} = \begin{cases} \varphi_{12} - \theta_1 & , \varphi_{12} \geq \theta_1 \\ \varphi_{12} - \theta_1 + 360° & , \varphi_{12} < \theta_1 \end{cases}, \quad 0° \leq \alpha_{12} < 360° \tag{10.7}$$
$$\alpha_{23} = \begin{cases} \varphi_{23} - \theta_2 & , \varphi_{23} \geq \theta_2 \\ \varphi_{23} - \theta_2 + 360° & , \varphi_{23} < \theta_2 \end{cases}, \quad 0° \leq \alpha_{23} < 360° \tag{10.8}$$
$$\vdots$$

$$\boldsymbol{\alpha_{N1}} = \begin{Bmatrix} \varphi_{N1} - \theta_N & , \varphi_{N1} \geq \theta_N \\ \varphi_{N1} - \theta_N + 360° & , \varphi_{N1} < \theta_N \end{Bmatrix}, \quad 0° \leq \alpha_{N1} < 360° \tag{10.9}$$

$$\boldsymbol{\beta_{12}} = \begin{Bmatrix} \varphi_{12} - \theta_2 & , \varphi_{12} \geq \theta_2 \\ \varphi_{12} - \theta_2 + 360° & , \varphi_{12} < \theta_2 \end{Bmatrix}, \quad 0° \leq \beta_{12} < 360° \tag{10.10}$$

$$\boldsymbol{\beta_{23}} = \begin{Bmatrix} \varphi_{23} - \theta_3 & , \varphi_{23} \geq \theta_3 \\ \varphi_{23} - \theta_3 + 360° & , \varphi_{23} < \theta_3 \end{Bmatrix}, \quad 0° \leq \beta_{23} < 360° \tag{10.11}$$

$$\vdots$$

$$\boldsymbol{\beta_{N1}} = \begin{Bmatrix} \varphi_{N1} - \theta_1 & , \varphi_{N1} \geq \theta_1 \\ \varphi_{N1} - \theta_1 + 360° & , \varphi_{N1} < \theta_1 \end{Bmatrix}, \quad 0° \leq \beta_{N1} < 360° \tag{10.12}$$

$$x^{initial} = \frac{1}{N}\sum_{i=1}^{N} x_i \tag{10.13}$$

$$y^{initial} = \frac{1}{N}\sum_{i=1}^{N} y_i \tag{10.14}$$

$$d = \sqrt{(x^{initial} - x_c)^2 + (y^{initial} - y_c)^2} \tag{10.15}$$

$$\theta_l = \tan^{-1}\left(\frac{y^{initial} - y_c}{x^{initial} - x_c}\right), \quad 0° \leq \theta_l < 360° \tag{11.16}$$

$$\gamma = \begin{Bmatrix} \theta_l - \theta_c & , \theta_l \geq \theta_c \\ \theta_l - \theta_c + 360° & , \theta_l < \theta_c \end{Bmatrix}, \quad 0° \leq \gamma < 360° \tag{10.17}$$

$$\boldsymbol{x} = d\cos\gamma \tag{10.18}$$

$$\boldsymbol{y} = d\sin\gamma \tag{10.19}$$

$$\boldsymbol{\omega} = \begin{Bmatrix} \varphi_{12} - \theta_c & , \varphi_{12} \geq \theta_c \\ \varphi_{12} - \theta_c + 360° & , \varphi_{12} < \theta_c \end{Bmatrix}, \quad 0° \leq \omega < 360° \tag{10.20}$$

To make it easier for the reader to recall what these equations represent, Figure 10.1 illustrates the l, α, β, x, y, and ω attributes of a 4-node Pattern, which would be stored in the corresponding feature vector, v. The d and γ attributes, which are used to calculate x and y, are also illustrated.

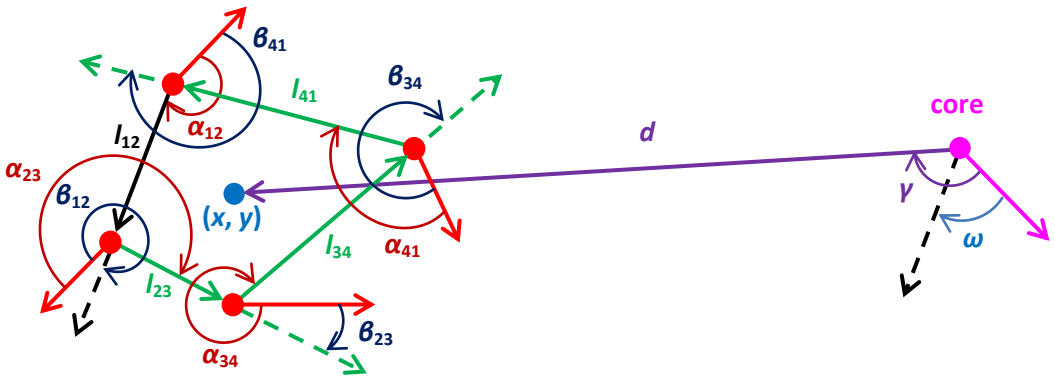


Figure 10.1: Attributes of an example 4-node Pattern.

In Equations (10.1) to (10.20), the constants **in red and bold** are known from v; therefore, we appear to have a total of **$3N + 3$ known values:** l_{12}, l_{23}, …, l_{N1}, α_{12}, α_{23}, …, α_{N1}, β_{12}, β_{23}, …, β_{N1}, x, y, and ω. In addition, we appear to have a total of **$4N + 8$ unknown values:** x_1, x_2, …, x_N, y_1, y_2, …, y_N, θ_1, θ_2, …, θ_N, φ_{12}, φ_{23}, …, φ_{N1}, x^{initial}, y^{initial}, x_c, y_c, θ_c, θ_l, d, and γ. On closer inspection, however, we realise that Equations (10.18) and (10.19) can be rearranged to determine γ and d using Equations (10.21) and (10.22), respectively:

$$\gamma = \tan^{-1}\left(\frac{y}{x}\right) \tag{10.21}$$

$$d = \frac{x}{\cos\gamma} \quad or \quad d = \frac{y}{\sin\gamma} \tag{10.22}$$

So, we actually have a total of **$3N + 5$ known values** (l_{12}, l_{23}, …, l_{N1}, α_{12}, α_{23}, …, α_{N1}, β_{12}, β_{23}, …, β_{N1}, x, y, ω, d, and γ) and **$4N + 6$ unknown values** (x_1, x_2, …, x_N, y_1, y_2, …, y_N, θ_1, θ_2, …, θ_N, φ_{12}, φ_{23}, …, φ_{N1}, x^{initial}, y^{initial}, x_c, y_c, θ_c, and θ_l). Since there are more *unknowns* than there are *knowns*, this means that we have an underdetermined set of equations and we thus cannot recover the *original* m_1, m_2, …, m_N attributes. This answer was anticipated, since the Pattern's location and orientation are expressed *relative to the core*, which means that its location and orientation in the *original* image coordinate system have been lost.

Although we cannot recover the *original* (x, y, θ) attributes of the N minutiae that were used to construct v, it might be possible to figure out these attributes *relative to the core* (x, y, θ). If the attributes are relative to the core, then it does not matter what the core attributes actually are; so, we can assume that x_c, y_c and θ_c are **known**, since they can essentially take on any value. Consequently, we now have a total of **$3N + 8$ known values** (l_{12}, l_{23}, …, l_{N1}, α_{12}, α_{23}, …, α_{N1}, β_{12}, β_{23}, …, β_{N1}, x, y, ω, d, γ, x_c, y_c, and θ_c) and **$4N + 3$ unknown values** (x_1, x_2, …, x_N, y_1, y_2, …, y_N, θ_1, θ_2, …, θ_N, φ_{12}, φ_{23}, …, φ_{N1}, x^{initial}, y^{initial}, and θ_l). Since there are now fewer *unknowns* than there are *knowns*, we may conclude that it is indeed possible to recover the attributes of m_1, m_2, …, m_N *relative to the core*. For example, since θ_c and ω are known, we may begin by using Equation (10.20) to determine φ_{12} as follows:

- If $\omega + \theta_c < 360°$, then $\varphi_{12} = \omega + \theta_c$
- If $\omega + \theta_c \geq 360°$, then $\varphi_{12} = \omega + \theta_c - 360°$

The value obtained for φ_{12} can then be plugged into Equations (10.7) and (10.10) to determine θ_1 and θ_2, respectively, as follows:

- If $\varphi_{12} - \alpha_{12} \geq 0°$, then $\theta_1 = \varphi_{12} - \alpha_{12}$
- If $\varphi_{12} - \alpha_{12} < 0°$, then $\theta_1 = \varphi_{12} - \alpha_{12} + 360°$

- If $\varphi_{12} - \beta_{12} \geq 0°$, then $\theta_2 = \varphi_{12} - \beta_{12}$
- If $\varphi_{12} - \beta_{12} < 0°$, then $\theta_2 = \varphi_{12} - \beta_{12} + 360°$

In a similar way, the value obtained for θ_2 can then be plugged into Equation (10.8) to determine φ_{23}. Once φ_{23} is known, we can use it in Equation (10.11) to establish θ_3. This process can be continued until φ_{12}, φ_{23}, …, φ_{N1}, θ_1, θ_2, …, θ_N have all been determined. Then, we can proceed to calculate the *x*- and *y*-coordinates of the minutiae relative to the core. Firstly, we can establish θ_l from Equation (10.17) as follows:

- If $\gamma + \theta_c < 360°$, then $\theta_l = \gamma + \theta_c$
- If $\gamma + \theta_c \geq 360°$, then $\theta_l = \gamma + \theta_c - 360°$

Since Equation (10.15) and Equation (10.16) now together have only 2 *unknowns* (x^{initial} and y^{initial}) and 4 *knowns* (d, θ_l, x_c and y_c), they can be used simultaneously to figure out x^{initial} and y^{initial} as follows:

$$d = \sqrt{(x^{initial} - x_c)^2 + (y^{initial} - y_c)^2} \tag{10.15}$$

$$d^2 = \left(x^{initial} - x_c\right)^2 + \left(y^{initial} - y_c\right)^2 \tag{10.23}$$

$$\left(x^{initial} - x_c\right)^2 = d^2 - \left(y^{initial} - y_c\right)^2 \tag{10.24}$$

$$x^{initial} - x_c = \sqrt{d^2 - (y^{initial} - y_c)^2} \tag{10.25}$$

Substitute Equation (10.25) into Equation (10.16):

$$\theta_l = \tan^{-1}\left(\frac{y^{initial} - y_c}{x^{initial} - x_c}\right) \tag{10.16}$$

$$\theta_l = \tan^{-1}\left(\frac{y^{initial} - y_c}{\sqrt{d^2 - (y^{initial} - y_c)^2}}\right) \tag{10.26}$$

$$\frac{y^{initial} - y_c}{\sqrt{d^2 - (y^{initial} - y_c)^2}} = \tan(\theta_l) \tag{10.27}$$

Square both sides of Equation (10.27):

$$\frac{\left(y^{initial} - y_c\right)^2}{d^2 - (y^{initial} - y_c)^2} = \tan^2(\theta_l) \tag{10.28}$$

$$\left(y^{initial} - y_c\right)^2 = \tan^2(\theta_l)\left(d^2 - \left(y^{initial} - y_c\right)^2\right) \tag{10.29}$$

$$\left(y^{initial} - y_c\right)^2 = d^2 \tan^2(\theta_l) - \left(y^{initial} - y_c\right)^2 \tan^2(\theta_l) \tag{10.30}$$

$$\left(y^{initial} - y_c\right)^2 + \left(y^{initial} - y_c\right)^2 \tan^2(\theta_l) = d^2 \tan^2(\theta_l) \tag{10.31}$$

$$\left(y^{initial} - y_c\right)^2 (1 + \tan^2(\theta_l)) = d^2 \tan^2(\theta_l) \tag{10.32}$$

$$\left(y^{initial} - y_c\right)^2 = \frac{d^2 \tan^2(\theta_l)}{(1 + \tan^2(\theta_l))} \tag{10.33}$$

$$y^{initial} - y_c = \sqrt{\frac{d^2 \tan^2(\theta_l)}{(1 + \tan^2(\theta_l))}} \tag{10.34}$$

$$\therefore y^{initial} = y_c + \sqrt{\frac{d^2 \tan^2(\theta_l)}{(1 + \tan^2(\theta_l))}} \tag{10.35}$$

Re-arrange Equation (10.25) to get the following:

$$x^{initial} = x_c + \sqrt{d^2 - (y^{initial} - y_c)^2} \tag{10.36}$$

Substitute Equation (10.33) into Equation (10.36):

$$\therefore x^{initial} = x_c + \sqrt{d^2 - \frac{d^2 \tan^2(\theta_l)}{(1 + \tan^2(\theta_l))}} \tag{10.37}$$

At this point, we have **5*N* + 11 known values** (l_{12}, l_{23}, …, l_{N1}, α_{12}, α_{23}, …, α_{N1}, β_{12}, β_{23}, …, β_{N1}, x, y, ω, d, γ, x_c, y_c, θ_c, φ_{12}, φ_{23}, …, φ_{N1}, $x^{initial}$, $y^{initial}$, θ_l, θ_1, θ_2, …, θ_N) and **2*N* unknown values** (x_1, x_2, …, x_N, y_1, y_2, …, y_N). In other words, the only *unknown* variables are the x and y attributes of the N constituent minutiae (relative to the core). Since φ_{12}, φ_{23}, …, φ_{N1} are known, and since these variables denote the orientations of the corresponding Pattern connection lines, we can use φ_{12}, φ_{23}, …, φ_{N1} to calculate the angle between each pair of connecting lines. Let δ_{ijk} denote the *inner* (i.e., smallest) angle between the connection line joining minutia i and minutia j, and the connection line joining minutia j to minutia k. Then Equation (10.38) can be used to calculate δ_{ijk}:

$$\delta_{ijk} = \min(|\varphi_{ij} - \varphi_{jk}|, 360° - |\varphi_{ij} - \varphi_{jk}|) \tag{10.38}$$

Substituting our previously established values for φ_{12} and φ_{23} into Equation (10.38) will then give us δ_{123}, the inner angle between the connection line joining minutia 1 to minutia 2, and

the connection line joining minutia 2 to minutia 3. We may proceed to calculate the remaining inner angles in a similar way. The result will be a Pattern whose *shape* is known.

At this point, we know the *shape*[42] of the *N*-node Pattern (i.e., the lengths of the connection lines, the angles between the connection lines, and thus the coordinates of the *N* minutiae relative to each other), the *location* of the Pattern in terms of its centroid, and the *orientation* of the Pattern in terms of the orientation of the line connecting minutia 1 to minutia 2. The final step is to use this information to determine the (x, y) coordinates of the shape's vertices, such that when those vertices are used to calculate the shape's centroid via Equations (10.13) and (10.14), the centroid would be equal to the coordinates specified by $(x^{\text{initial}}, y^{\text{initial}})$. This may be solved by considering the fact that the shape centroid is located at the point of intersection between the *N* lines starting at each of the *N* vertices and finishing at the midpoint of the opposite connection line (see Figure 10.2).

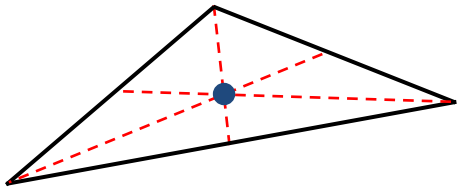


Figure 10.2: Illustration of the fact that the centroid of a shape is located at the point of intersection of the N lines extending from each of its N vertices to the midpoint of the opposite line (NB: In this figure, N = 3).

If we choose one of the connection lines to be the reference line, we can then initiate a new coordinate system with its origin at the midpoint of that line. Each of the Pattern's *N* vertices can then be expressed relative to that new coordinate system and the resulting coordinates can be averaged to obtain a new centroid. The *x*- and *y*-coordinate offset of the new centroid from the original centroid can then be calculated, and each of the *N* vertex coordinates can be corrected by the same amount to determine their (x, y) coordinates in the same coordinate system as the original Pattern centroid. At this point, we would have succeeded in recovering the (x, y) coordinates of each of the Pattern's *N* minutiae relative to the core.

In summary, the analysis in this section has shown that:

- The feature vector of an $\mathbf{FC}_{360°}$ *N*-node Pattern *cannot* be used to determine the *original* (x, y, θ) attributes of the *N* constituent minutiae. This is because the Pattern's location and orientation are expressed *relative to the core*, which means that its location and orientation in the *original* image coordinate system have been lost.

[42] The definition of an *N*-node Pattern in Section 5.2.1 specifies that a Pattern consists of a few minutiae connected in a particular order via *straight lines*. This means that the *shape* of an *N*-node Pattern will always consist of *straight lines only*.

- The feature vector of an $\mathbf{FC}_{360^\circ}$ N-node Pattern *can* be used to determine the (x, y, θ) attributes of the N constituent minutiae *relative to the core*.

Based on this analysis, we may conclude that the feature vector of an $\mathbf{FC}_{360^\circ}$ N-node Pattern leaks N out of T minutiae from the underlying T-minutiae template.

10.3 NON-INVERTIBILITY OF AN $\mathbf{FC}_{360^\circ}$ *N*-NODE PATTERN

In Section 10.2, it was established that the feature vector of an $\mathbf{FC}_{360^\circ}$ N-node Pattern leaks N out of T minutiae from the underlying minutiae template, where T is used to denote the total number of minutiae available in the full minutiae template. We may, therefore, envision our proposed fingerprint construct as a T-to-N mapping of the minutiae in the minutiae template. Since $N << T$, it is not possible to uniquely determine the entire set of T minutiae from which the subset of N minutiae was extracted. In other words, the inverse mapping (i.e., N-to-T) is impossible, which immediately confirms that our proposed fingerprint construct is *non-invertible*.

In this section, we investigate the degree of *non-invertibility* that we may expect from our proposed fingerprint construct in its intended application scenario in practice (i.e., a cooperative-user civilian fingerprint recognition application). Section 10.3.1 considers *non-invertibility* from the point of view of the proportion of the entire minutiae template that remains unrevealed despite the N-minutiae leak by an N-node Pattern. Section 10.3.2 examines *non-invertibility* in terms of the complexity of recovering the entire minutiae template via a brute-force approach. Note that all of the experiments discussed in these two sections were conducted on our cooperative-user fingerprint database (see Chapter 6).

10.3.1 Non-invertibility Analysis 1: Proportion of Unrevealed Minutiae Template

This section empirically estimates the *non-invertibility* of our proposed fingerprint construct in terms of the proportion of a fingerprint's minutiae template that may be expected to remain unrevealed in practice despite the N minutiae that are leaked by the feature vector of an $\mathbf{FC}_{360^\circ}$ N-node Pattern. As before, let T denote the total number of minutiae available in a full minutiae template. The proportion of a minutiae template that remains unrevealed may thus be quantified as $1 - N/T$.

For each person in our cooperative-user fingerprint database, the proportion of their minutiae template that remains unrevealed by an $\mathbf{FC}_{360°}$ N-node Pattern's feature vector was computed separately for 3-node, 4-node, and 5-node Patterns. This was conducted as follows:

1. For each person, the constituent minutiae of 3-, 4-, and 5-node Patterns were selected from the corresponding reference minutiae set. The reference minutiae set for a particular person consisted of those minutiae that appear in *all* of the first 5 samples of that person's fingerprint, as in the *Most Favourable Genuine User Authentication Scenario* that was established in Chapter 7. In order to free this analysis from the potential errors of automated minutiae extractors and matchers, we used *manually* extracted minutiae and their *manually* established correspondences (as for the analysis in Chapter 7).
2. The number of reference minutiae (i.e., those minutiae that appear in *all* of the first 5 samples of each person's fingerprint) was used as T (i.e., the total number of minutiae in the full minutiae template) in turn. We then evaluated the *non-invertibility* as $1 - N/T$ for $N = \{3, 4, 5\}$ in turn. The resulting *non-invertibility* values for all 100 people in our database were gathered into a distribution, such that there was a separate distribution for each N.

Figure 10.3 compares the *non-invertibility* distributions for $N = 3$, $N = 4$, and $N = 5$, in terms of box and whisker plots. Note that box and whisker plots were used because they provide an effective visualisation of the comparison between the distributions whilst simultaneously allowing us to readily quantify any differences.

The most evident trend from Figure 10.3 is that the non-invertibility of our proposed fingerprint construct decreases as the Pattern size, N, increases. This makes sense, because a larger Pattern reveals a larger portion of the entire minutiae template than does a smaller Pattern. For example, Figure 10.3 tells us that, for our cooperative-user fingerprint database, the median *non-invertibility* of a single 3-node Pattern is 0.93, that of a 4-node Pattern is 0.90, and that of a 5-node Pattern is 0.88. Note that, since the median in each of the box and whisker plots in Figure 10.3 is approximately half way between the corresponding upper and lower quartiles, this suggests that all the non-invertibility distributions are approximately symmetrical around the sample mean. Consequently, the mean of each distribution should be approximately equal to its median; indeed, a calculation of the means confirms this observation. We may, therefore, conclude that:

- An attacker with access to a 3-node Pattern generated from the fingerprint of a genuine user would, on average, be able to recover only 7% of the fingerprint's entire minutiae template. This means that, on average, 93% of the template would remain unrevealed.

- An attacker with access to a 4-node Pattern generated from the fingerprint of a genuine user would, on average, be able to recover only 10% of the fingerprint's entire minutiae template. This means that, on average, 90% of the template would remain unrevealed.
- An attacker with access to a 5-node Pattern generated from the fingerprint of a genuine user would, on average, be able to recover only 12% of the fingerprint's entire minutiae template. This means that, on average, 88% of the template would remain unrevealed.

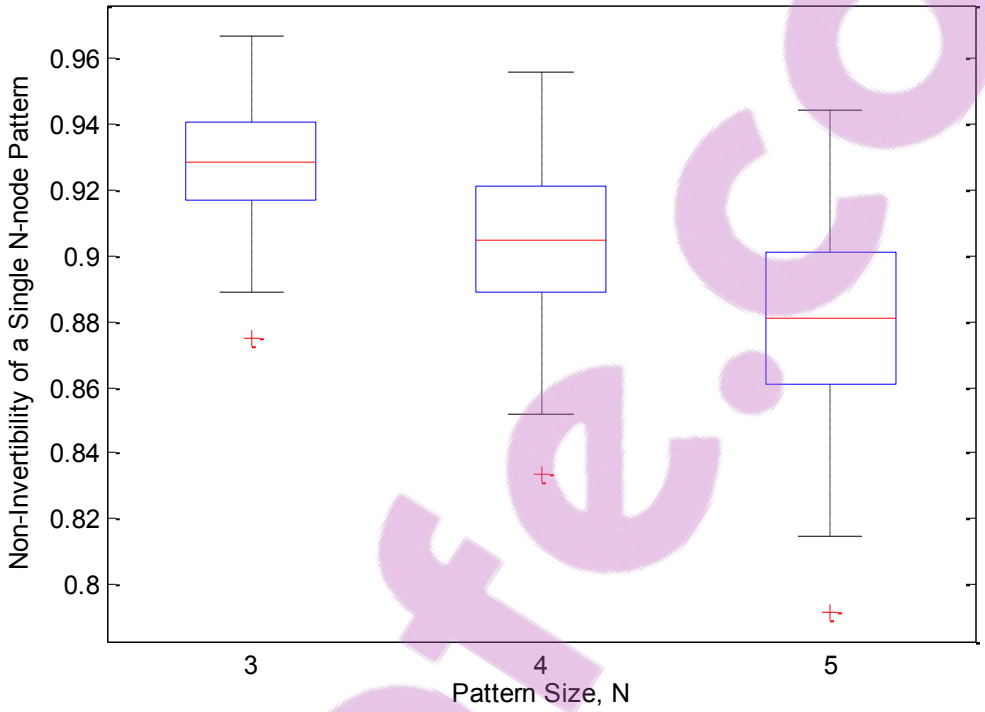


Figure 10.3: Box and whisker plots comparing the non-invertibility of an *N*-node Pattern (i.e., the proportion of the underlying minutiae template that is not revealed by an *N*-node Pattern feature vector) as the Pattern size, *N*, increases from 3 to 5.

These observations provide encouraging evidence to support the intuitively evident *non-invertibility* of our proposed fingerprint construct. In particular, the results in Figure 10.3 suggest that the average amount of information leaked by an *N*-node Pattern is small enough to ensure that the greater part of the corresponding minutiae template remains unrevealed in the event that an attacker steals a genuine user's reference *N*-node Pattern from the recognition system's database. Consequently, we may conclude that the amount of leaked information would be insufficient for an accurate reconstruction of the underlying fingerprint, which means that, even if an *N*-node Pattern is compromised, the underlying fingerprint could continue to be used for recognition purposes.

Note that our results for all three Pattern sizes are considerably better than the functional transform proposed in [132], where it was shown that only about 8% of the minutiae are concealed by the transform, meaning that 92% of the original minutiae template is

recoverable from the protected minutiae template. We may thus conclude that, compared to the functional transform in [132], our proposed fingerprint construct is able to secure, on average, 85% more of the minutiae template when 3-node Patterns are used, 82% more when 4-node Patterns are used, and 80% more when 5-node Patterns are used.

The results obtained for the *non-invertibility* of our proposed fingerprint construct at all three Pattern sizes are also better than [150], in terms of the proportion of the entire minutiae template that remains unrecoverable in the event that a single protected template is compromised. In [150], the authors showed that it is possible to recover at least 25.4% of the original minutiae template when attempting to invert the entire protected template, and that at least 30.5% of the original minutiae are recoverable from individual protected minutiae cylinders[43]. The results obtained for our proposed fingerprint construct, however, indicate that only 7% of the original minutiae template is recoverable from a 3-node Pattern, 10% from a 4-node Pattern, and 12% from a 5-node Pattern.

Note that the reason we have chosen to compare our proposed fingerprint construct to [132] and [150] is because these are the only two non-invertible transforms in the literature (as far as we know) for which non-invertibility is evaluated in terms of the percentage of recoverable minutiae.

10.3.2 Non-invertibility Analysis 2: Complexity of Reconstructing Full Minutiae Template

Section 10.3.1 evaluated the *non-invertibility* of our proposed fingerprint construct in terms of the proportion of the underlying minutiae template that remains unrevealed in the event that a genuine user's reference N-node Pattern is stolen from the database. It was found that, since $N << T$, we may generally expect most of the minutiae template to remain unrevealed, regardless of the Pattern size (limited to $N = \{3, 4, 5\}$). In the current section, we investigate the *non-invertibility* of our proposed fingerprint construct in terms of the complexity of recovering the unrevealed portion of the minutiae template using a brute-force approach.

Assuming the notation used in Section 10.3.1, let T denote the total number of minutiae available in a full minutiae template and let N denote the number of minutiae used to construct a person's reference N-node Pattern. Since Section 10.2 demonstrated that the attributes of a Pattern's N constituent minutiae can be recovered from the corresponding feature vector, this implies that $T - N$ of the minutiae in the underlying minutiae template are *not* leaked by the N-node Pattern. Consequently, if an attacker were interested in

[43] The percentage of recoverable minutiae was shown to depend on the values of certain parameters specific to the proposed method. The results reported here are the *best* results reported in the corresponding paper.

reconstructing a genuine user's entire minutiae template, they would essentially need to *guess* the remaining $T - N$ minutiae in the template. To get an idea of the maximum number of guesses required in this endeavour, it was first necessary to establish the size of the search space; in other words, we needed to determine the range of possible minutiae attributes and the likelihood of an attribute assuming a particular value in that range. Recall, from Section 10.2, that the attributes of a minutia used in the construction of an *N*-node Pattern are the minutia's (x, y) location and its orientation, θ, in the underlying fingerprint image. So, we began by estimating the distributions corresponding to minutiae x, y, and θ attributes. This was done by extracting all the minutiae and core points in every fingerprint image in our cooperative-user fingerprint database using the VeriFinger SDK [217] (as in the experiments discussed in Chapter 8), and then expressing each minutia relative to the corresponding core point[44]. Let $m_i = (x_i, y_i, \theta_i)$ denote the attributes of an arbitrary minutia in the original image coordinate system, let $c = (x_c, y_c, \theta_c)$ denote the attributes of the corresponding core point, and let $m'_i = (x'_i, y'_i, \theta'_i)$ denote the coordinates of minutia m_i relative to the core, c. To generate m'_i from m_i and c, Equations (10.39) to (10.43) were applied.

$$r = \sqrt{(x_i - x_c)^2 + (y_i - y_c)^2} \tag{10.39}$$

$$\sigma = \tan^{-1}\left(\frac{y_i - y_c}{x_i - x_c}\right), \quad 0° \leq \sigma < 360° \tag{10.40}$$

$$x'_i = r \cos\sigma \tag{10.41}$$

$$y'_i = r \sin\sigma \tag{10.42}$$

$$\theta'_i = \begin{cases} \theta_i - \theta_c, & \theta_i \geq \theta_c \\ \theta_i - \theta_c + 360°, & \theta_i < \theta_c \end{cases}, \quad 0° \leq \theta'_i < 360° \tag{10.43}$$

The resulting x', y', and θ' attributes of all the minutiae in all the fingerprints in our cooperative-user fingerprint database were then used to plot a separate probability distribution for each attribute. Figure 10.4 depicts the probability distribution of the minutiae x' attributes, Figure 10.5 illustrates the probability distribution of the minutiae y' attributes, and Figure 10.6 shows the probability distribution of the minutiae θ' attributes. Note that each of the x', y', and θ' attributes used to generate these probability distributions was rounded to the nearest whole number. This is because it is difficult to compute these attributes very precisely, which means that, in practice, minutiae attributes that differ by less than 1 would

[44] This was necessary to ensure that the minutiae across multiple samples of the same person's fingerprint were aligned relative to a common reference frame.

generally be considered to correspond to the *same* attribute; e.g., an x-coordinate of 5.9 and an x-coordinate of 5.7 would generally be considered to be the same x-coordinate.

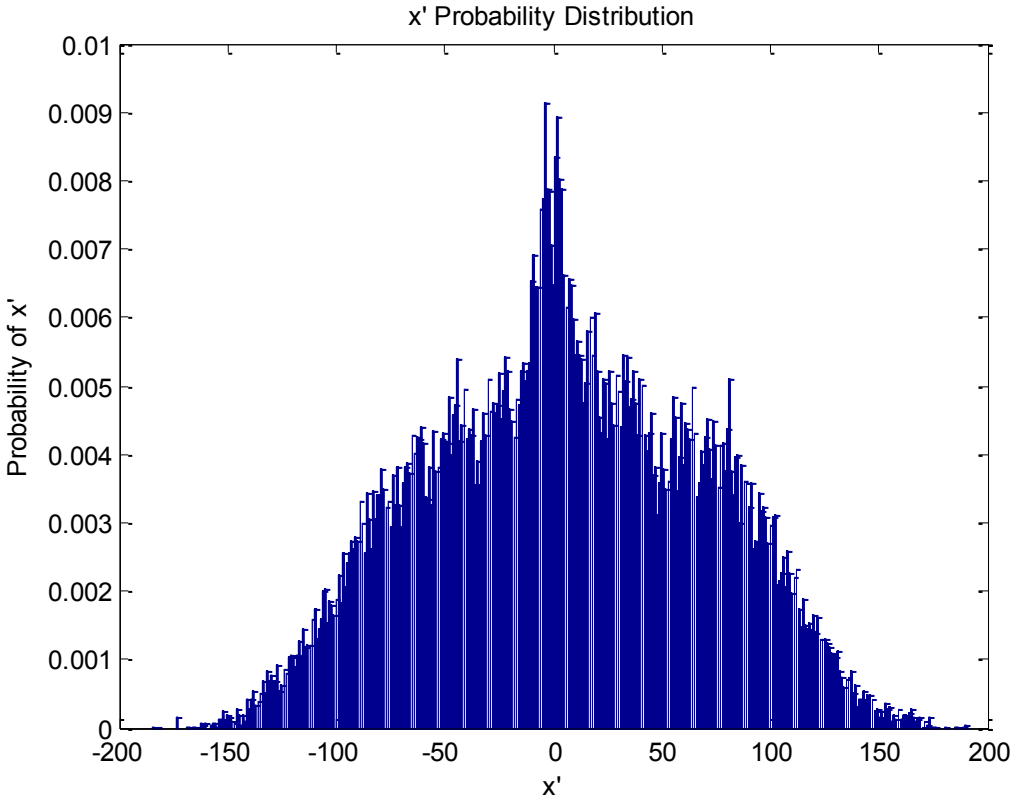


Figure 10.4: Probability distribution corresponding to minutiae x' attributes.

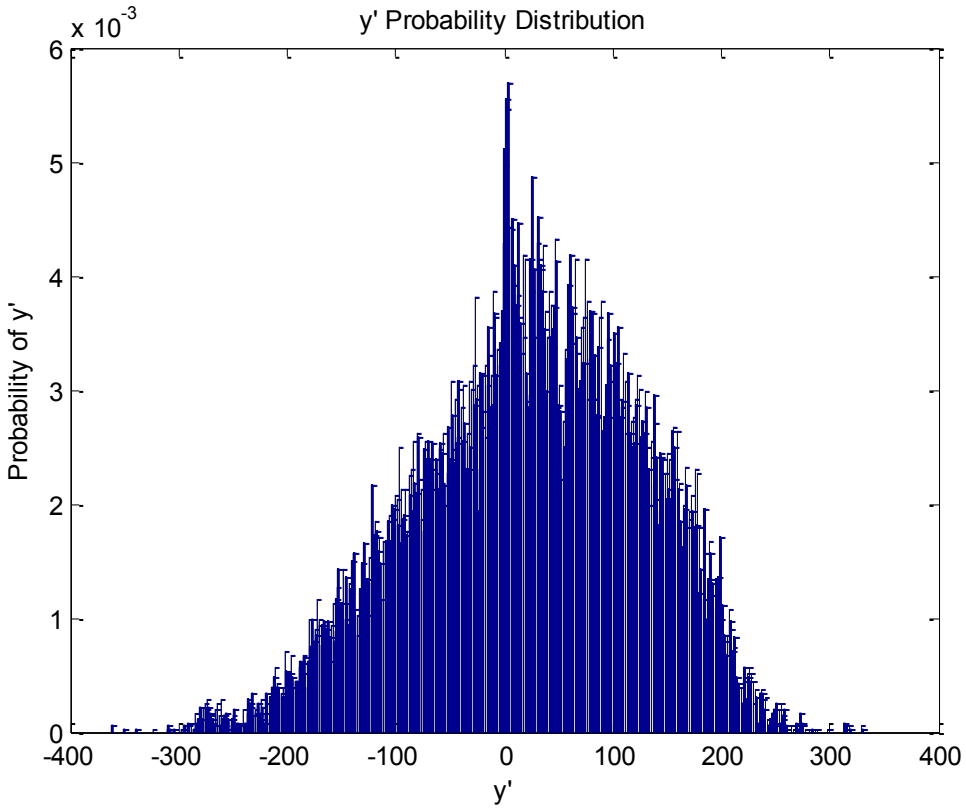


Figure 10.5: Probability distribution corresponding to minutiae y' attributes.

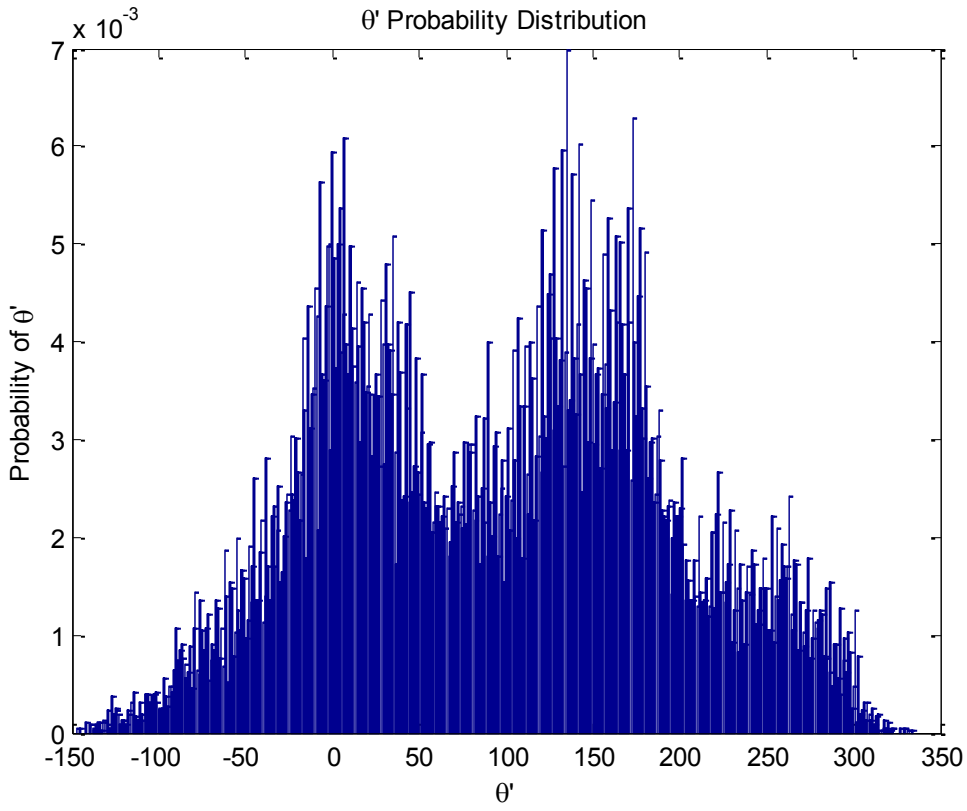


Figure 10.6: Probability distribution corresponding to minutiae θ' attributes.

In order to use each resulting probability distribution to estimate the difficulty of guessing $T - N$ minutiae, the information entropy of each distribution was computed. Let $H(X')$ denote the information entropy corresponding to the x' probability distribution, let $H(Y')$ indicate the information entropy pertaining to the y' probability distribution, and let $H(\theta')$ represent the information entropy of the θ' probability distribution. Furthermore, let $P(x'_i)$, $P(y'_i)$, and $P(\theta'_i)$ denote the probability of x'_i, y'_i, and θ'_i, respectively, occurring in a fingerprint. Then, Equations (10.44), (10.45), and (10.46) may be used to calculate $H(X')$, $H(Y')$, and $H(\theta')$, respectively.

$$H(X') = -\sum_i P(x'_i) \log_2 P(x'_i) \tag{10.44}$$

$$H(Y') = -\sum_i P(y'_i) \log_2 P(y'_i) \tag{10.45}$$

$$H(\theta') = -\sum_i P(\theta'_i) \log_2 P(\theta'_i) \tag{10.46}$$

Note that the use of base 2 for the logarithm in Equations (10.44) to (10.46) ensures that the resulting entropy is measured in *bits*. Table 10.1 summarises the information entropy of the probability distributions in Figures 10.4, 10.5, and 10.6, which was computed using Equations (10.44), (10.45), and (10.46), respectively.

Table 10.1: Information entropy of minutiae x', y', and θ' attributes.

$H(X')$	$H(Y')$	$H(\theta')$
7.99 bits	8.69 bits	8.50 bits

The results in Table 10.1 indicate that, on average, 7.99 bits are required to represent a minutia x-coordinate, 8.69 bits are needed to represent a minutia y-coordinate, and 8.50 bits are needed to represent a minutia angle, θ, when all three attributes are expressed relative to the corresponding fingerprint core. Consequently, we may estimate that the *total* number of bits required to represent an entire minutia, where $m'_i = (x'_i, y'_i, \theta'_i)$, is equal to 7.99 + 8.69 + 8.50 = **25.18 bits**, on average. This result indicates that, on average, there are $2^{[25.18]} = 2^{26} = 6.71 \times 10^7$ possible combinations of minutiae x', y', and θ' attributes, when each attribute is rounded to the nearest whole number. We may thus estimate that the *maximum* number of guesses required to guess a *single* minutia via a brute-force approach[45] is 6.71×10^7; therefore, the maximum number of guesses required to guess $T - N$ minutiae can be computed using Equation (10.47):

$$G_N^T = \left(2^{H(X')+H(Y')+H(\theta')}\right)^{T-N} = (6.71 \times 10^7)^{T-N} \tag{10.47}$$

Equation (10.47) was applied to our cooperative user fingerprint database for $N = \{3, 4, 5\}$, using the same T values as those employed in the experiment described in Section 10.3.1. The median of the maximum number of guesses computed for each of the 100 people in our database was calculated for each N. Table 10.2 summarises the resulting maximum number of guesses as N increases from 3 to 5.

Table 10.2: Maximum number of guesses required to recover the entire minutiae template as the Pattern size, *N*, increases.

	$N = 3$	$N = 4$	$N = 5$
Maximum Number of Guesses	1.75×10^{305}	2.60×10^{297}	3.88×10^{289}

[45] A brute-force approach involves trying out every possible solution until the correct one is found, assuming that there is some way of checking whether or not a guess is correct.

From Table 10.2, it is evident that the maximum number of guesses required to guess the unrevealed portion of a full minutiae template (i.e., $T - N$ minutiae) decreases as the Pattern size, N, increases. This makes sense, since a larger Pattern reveals a larger number of minutiae, which means that fewer minutiae remain in the unrevealed portion of the minutiae template. Having said this, the maximum number of guesses required for all three Pattern sizes in Table 10.2 is nevertheless extremely large. For example, if an attacker were to make a billion guesses per second, it would take about 1.23×10^{273} years to guess the entire minutiae template from the 5 minutiae revealed by a 5-node Pattern. This amount of time is significantly greater than the estimated age of the universe, which is approximately 13.8 billion years.

To further understand the significance of the results in Table 10.2, Table 10.3 shows the maximum number of guesses from Table 10.2 in terms of their equivalent "bit-strength", which is a commonly used measure to estimate the difficulty of guessing an n-bit decryption key in cryptographic data protection schemes in practice.

Table 10.3: The equivalent "bit-strength" of the maximum number of guesses from Table 10.2.

	$N = 3$	$N = 4$	$N = 5$
Bit Strength	$\log_2(1.75 \times 10^{305}) \approx$ **1,014**	$\log_2(2.60 \times 10^{297}) \approx$ **988**	$\log_2(3.88 \times 10^{289}) \approx$ **962**

From Table 10.3, we may draw the following conclusions:

- The difficulty of recovering an entire minutiae template via brute force when 3 minutiae are leaked by a 3-node Pattern is approximately equivalent to the difficulty of guessing a 1,014-bit decryption key. This is approximately $2^{(1014-128)} = 2^{886}$ times more difficult than guessing a 128-bit Advanced Encryption Standard (AES) decryption key[46], which is commonly used for present-day data encryption/decryption and is considered to provide more than enough protection to the encrypted data.
- The difficulty of recovering an entire minutiae template via brute force when 4 minutiae are leaked by a 4-node Pattern is approximately equivalent to the difficulty of guessing a 988-bit decryption key. This is approximately $2^{(988-128)} = 2^{860}$ times more difficult than guessing a 128-bit AES decryption key.
- The difficulty of recovering an entire minutiae template via brute force when 5 minutiae are leaked by a 5-node Pattern is equivalent to the difficulty of guessing a 962-bit

[46] Each additional bit doubles the guessing complexity.

decryption key. This is approximately $2^{(962-128)} = 2^{834}$ times more difficult than guessing a 128-bit AES decryption key.

The analysis above provides extremely encouraging support for the *non-invertibility* of our proposed fingerprint construct. It must be noted, however, that the guessing complexity of recovering an entire minutiae template via brute force would, in practice, be influenced by the desired accuracy of the recovered minutiae attributes. For example, in our analysis above, each minutia attribute was rounded to the nearest whole number. Consequently, the guessing complexity was based on the requirement that a minutia attribute would only be considered correct if it was exactly equal to the whole number equivalent of the true minutia attribute. If the recovered minutiae attributes need not be very precise, then we could consider a guessed minutia attribute to be correct if it was within $\pm\tau$ of the true minutia attribute, where τ denotes some pre-determined threshold. In this case, the guessing complexity would decrease, since the attacker would essentially need to check only every τ^{th} attribute value; however, the reconstructed minutiae template would be less accurate. In order for the recovered minutiae template to be as accurate as possible, τ would need to be small; therefore, we may reasonably conclude that the guessing complexity would remain high in practice, provided that τ is sensibly selected in order for the reconstruction to be accurate.

Note that, as mentioned in Section 4.5.1, the inconsistencies in evaluating the *non-invertibility* of a fingerprint template protection scheme in the literature mean that quantitative comparisons of the *non-invertibility* of different fingerprint template protection schemes are often avoided. A conceptual comparison, however, suggests that the sparsity of our proposed fingerprint construct makes it intuitively more *non-invertible* than schemes that utilise the entire minutiae template in generating the protected template. This is because our proposed fingerprint construct leaks only N out of the T minutiae available in the entire minutiae template (where $N << T$), meaning that the *non-invertibility* of our scheme essentially relies on fingerprint individuality, i.e., the difficulty of guessing the locations and orientations of the remaining minutiae in the template. Alternatively, fingerprint template protection schemes that utilise the entire minutiae template depend on the properties of the employed transform and its associated external parameters to impart *non-invertibility* to the protected fingerprint template. Unless the transform is able to effectively hide $T - N$ minutiae, as our proposed fingerprint construct is able to do, it cannot be expected to provide a greater level of *non-invertibility* to the underlying fingerprint than our proposed fingerprint construct. Since, to the best of our knowledge, essentially all non-invertible fingerprint template protection schemes existing in the literature employ the entire minutiae template, we may reason that our

proposed fingerprint construct has the potential to outperform existing fingerprint template protection schemes in terms of preserving the integrity of the underlying fingerprint.

In concluding the analysis on the non-invertibility of our proposed fingerprint construct, the following two final points should be noted:

- An analysis of the complexity of recovering a fingerprint's *entire* minutiae template may be too optimistic for fingerprint template protection schemes that use the full minutiae template. This is because, for those methods, it is usually sufficient to recover less than 100% of the minutiae template in order to approximately reconstruct the underlying fingerprint image and use it to fool the recognition system into accepting an impostor as a genuine user. Our proposed fingerprint construct, however, requires an attacker to have access to the *exact* set of N minutiae used to generate a genuine user's reference N-node Pattern. So, even if the attacker had access to 99% of the genuine user's full minutiae template, but one or more of the N minutiae used to generate the genuine user's reference Pattern was in the 1% missing portion of the template, the attacker would not be able to generate the required reference Pattern. The point of the non-invertibility analysis in this section was thus to determine the amount of effort required by an attacker to gain access to a user's specific minutiae 'alphabet', or the amount of effort needed to gain access to every identity (i.e., every Pattern) possible from the genuine user's fingerprint (even though access to the minutiae template itself does *not* reveal the user's chosen reference Pattern to the attacker).
- On the point of *guessing* minutiae, we should address the susceptibility of our proposed fingerprint construct to a Hill-climbing attack. In a traditional Hill-climbing attack, an attacker starts by artificially generating a set of minutiae attributes (i.e., x, y, θ) and submitting them to the recognition system. The attacker takes note of the match score returned by the recognition system and then adjusts their artificial minutiae set (by perturbing the x, y, θ attributes and/or by generating more minutiae points) in order to try to achieve a higher match score. Once the match score is high enough to allow the impostor to pass off as a genuine user, the impostor knows that their artificial minutiae set is sufficiently close to the minutiae set of the genuine user that they are impersonating. Our proposed fingerprint construct makes it difficult to launch a Hill-climbing attack for several reasons. Firstly, the attacker must guess the *exact* set of N minutiae (as opposed to *any* N minutiae) used in the genuine user's reference Pattern. Secondly, they must guess the correct *ordering* of those N minutiae (there are $N!$ ways to arrange N minutiae into an N-node Pattern). Thirdly, our matching method does not return a partial match score, i.e., the system would only return a "Match" or "No Match" decision depending on whether

the Pattern as a whole matches. So, the attacker would not know how close they are as they attempt to adjust their guessed minutiae set; they would only know whether or not they guessed the correct Pattern. Fourthly, in a practical implementation of our method, a user would only be allowed a minimum number of authentication attempts. Since this number is likely to be small (e.g., 3), an attacker is unlikely to be able to succeed in a Hill-climbing attack before being locked out of the system. Finally, even if the Hill-climbing attack were to succeed in one recognition system, the recovered *N*-node Pattern could only be used in that *one* system (provided that the user employs a different Pattern in each application) until the attack is detected and the victim changes their reference Pattern. In traditional fingerprint recognition, however, the minutiae recovered as a result of a successful Hill-climbing attack could be employed in *any* system in which that same fingerprint is employed, as long as the systems' matching thresholds are similar.

10.4 SUMMARY

This chapter evaluated the *non-invertibility* of our proposed fingerprint construct.

We showed that the feature vector of an $\mathbf{FC}_{360^\circ}$ *N*-node Pattern can be used to recover the attributes of the Pattern's *N* constituent minutiae relative to the corresponding fingerprint's core. It was thus established that an $\mathbf{FC}_{360^\circ}$ *N*-node Pattern leaks *N* minutiae from a *T*-minutiae template, where $N << T$.

Our proposed fingerprint construct was then likened to a *T*-to-*N* mapping. Since $N << T$, our construct was proven to be *non-invertible*. The degree of *non-invertibility* of our fingerprint construct was then evaluated in terms of the proportion of the underlying minutiae template that remains unrevealed despite the *N* minutiae leaked by an *N*-node Pattern. It was found that, on average, 93% of a fingerprint's minutiae template may be expected to remain unrevealed in the event that a 3-node Pattern originating from the corresponding fingerprint is stolen from the database, 90% would remain unrevealed with the compromise of a 4-node Pattern, and 88% would remain unrevealed with the compromise of a 5-node Pattern. This was followed by an estimation of the complexity of recovering the unrevealed portion of the minutiae template using a brute-force approach. It was shown that the sparsity of an $\mathbf{FC}_{360^\circ}$ *N*-node Pattern ensures that it is extremely difficult (in fact, practically impossible) to reconstruct the corresponding fingerprint's entire minutiae template, which further justified the degree of *non-invertibility* provided by our proposed fingerprint construct.

Overall, the analysis in this chapter produced strong evidence to support the *non-invertibility* of our proposed fingerprint construct, thereby further condoning its deployment as an effective fingerprint template protection scheme in practice.

Chapter 11

Susceptibility of $FC_{360°}$ to a Record Multiplicity Attack

Chapter 10 analysed the *non-invertibility* of our proposed fingerprint construct, $\mathbf{FC_{360°}}$. It was shown that a compromised *N*-node Pattern leaks only a small amount of information about the original minutiae template, thereby ensuring that it is practically impossible to reconstruct the entire minutiae template from this information leak. This chapter considers the possibility of recovering a fingerprint's entire minutiae template by collecting multiple *N*-node Patterns from the same user, thereby analysing the susceptibility of our proposed fingerprint construct to what has been termed the Record Multiplicity Attack in the literature.

11.1 INTRODUCTION

In Chapter 10, the *non-invertibility* of our proposed fingerprint construct was justified in terms of the significant portion of the original minutiae template that remains unrecoverable in the event that a reference *N*-node Pattern is stolen from the database. The fact that an *N*-node Pattern leaks a small portion of the minutiae template, however, suggests the possibility of collecting multiple reference *N*-node Patterns from the same fingerprint and piecing the *N* minutiae recovered from each Pattern together to reconstruct the entire minutiae template[47].

The reconstruction of the original fingerprint template from multiple versions of its protected counterpart is referred to as a Record Multiplicity Attack (ARM)[48] in the literature. In a Record Multiplicity Attack, it is assumed that an attacker is somehow able to acquire two or more secured versions of a fingerprint template from the same person and that they are then able to *correlate* those protected templates to reveal the original (unsecured) fingerprint template. Recall that the most important aspect of our proposed fingerprint construct, which

[47] Note that this would only be possible if an attacker were able to collect multiple protected templates corresponding to the same fingerprint and if they were able to guarantee that those Patterns do, in fact, originate from the same fingerprint. This should be very difficult to achieve in practice.

[48] See the discussion in Section 4.5.1.

sets it apart from other fingerprint template protection schemes in the literature, is that it uses only a small proportion of the entire minutiae template in the generation of a protected template; the remainder of the minutiae in the template do not contribute to the protected template in any way. This means that our fingerprint template protection scheme is not susceptible to a *correlation* attack as such, since it is impossible for two *N*-node Patterns to reveal the entire minutiae template. Nevertheless, as our analysis in Chapter 10 showed that an *N*-node Pattern leaks a small proportion of the entire minutiae template, this chapter considers the susceptibility of our proposed fingerprint construct to a Record Multiplicity Attack in terms of the complexity of reconstructing an entire minutiae template by piecing together multiple *N*-node Patterns from the same user.

We begin with an investigation into the number of $\mathbf{FC}_{360^\circ}$ *N*-node Patterns that an attacker would need to acquire in order to be able to reconstruct the entire minutiae template. We then propose a modification to $\mathbf{FC}_{360^\circ}$, which increases the difficulty of reconstructing a minutiae template from multiple *N*-node Patterns.

11.2 SUSCEPTIBILITY OF $\mathbf{FC}_{360^\circ}$ TO A RECORD MULTIPLICITY ATTACK

This section investigates the susceptibility of the $\mathbf{FC}_{360^\circ}$ version of our proposed fingerprint construct to a Record Multiplicity Attack. In particular, we estimate the number of *N*-node Patterns that an attacker would need to obtain from the same fingerprint in order to be able to reconstruct the fingerprint's entire minutiae template. Note that multiple Patterns from the same fingerprint could be obtained in one of two ways:

(i) An attacker steals the reference *N*-node Pattern of a particular person enrolled in a fingerprint recognition system. The affected user replaces their compromised *N*-node Pattern with a new *N*-node Pattern from the same fingerprint. The attacker then steals the replacement *N*-node Pattern corresponding to the same user, and so on, until they have collected enough *N*-node Patterns to reconstruct the user's entire minutiae template. Note that, in this scenario, it is assumed that the attacker knows that these multiple Patterns belong to the same user.

(ii) A person enrols into multiple applications, using a different *N*-node Pattern from the same fingerprint in each application. An attacker is somehow able to figure out which applications this person is enrolled in and which Pattern in each application belongs to this particular person. Consequently, the attacker proceeds to steal all of that person's

used Patterns and pieces them together in an attempt to reconstruct the underlying minutiae template.

While, in practice, it should be extremely difficult for an attacker to obtain multiple *N*-node Patterns from the same person in either of these two scenarios, scenario (i) would perhaps be more feasible than scenario (ii). This is because, in scenario (i), the attacker would know for sure that the compromised user is enrolled into the system after the first attack, which means that the multiple Patterns would all come from the same database. In scenario (ii), however, the person's Patterns would be distributed across multiple applications, so the attacker would need to start by finding out which applications the person has enrolled into.

In Chapter 10, it was established that an $\mathbf{FC}_{360°}$ *N*-node Pattern feature vector can be used to determine the attributes of the Pattern's *N* constituent minutiae, relative to the core. Therefore, the number of $\mathbf{FC}_{360°}$ *N*-node Patterns from the same fingerprint that would be needed to recover the fingerprint's entire minutiae template would depend on how many *new* minutiae are revealed by each successive *N*-node Pattern collected by the attacker. Let *T* denote the total number of minutiae in the full minutiae template. In the worst-case scenario (from the genuine user's point of view), each new Pattern collected by the attacker would consist of *N* new minutiae, in which case the total number of Patterns needed may be calculated using Equation (11.1):

$$Number\ of\ Patterns\ Required\ in\ Worst\ Case\ Scenario = \left\lceil \frac{T}{N} \right\rceil \qquad (11.1)$$

In the best-case scenario (from the genuine user's point of view), the attacker would first collect *N*! Patterns generated using the same set of *N* minutiae (the *N*! Patterns are created by connecting the *N* minutiae in a different order each time). Then, the $(N! + 1)^{th}$ Pattern would contain 1 new minutia, and this Pattern would be followed by the remaining $(N + 1)! - 1$ Patterns generated using that 1 new minutia and the *N* minutiae from the previous set. In this case, the total number of Patterns needed may be calculated using Equation (11.2):

$$Number\ of\ Patterns\ Required\ in\ Best\ Case\ Scenario = \frac{T!}{(T-N)!} - (N! - 1) \qquad (11.2)$$

To get an idea of the number of $\mathbf{FC}_{360°}$ *N*-node Patterns needed in the worst and best case scenarios in practice, Equations (11.1) and (11.2), respectively, were applied to our cooperative-user fingerprint database. For each person, the total number of minutiae in their

full minutiae template, T, was established in the same way as for the experiment discussed in Section 10.3.1. Each person's T value was then used in Equations (11.1) and (11.2) to determine the number of N-node Patterns required in the worst and best case scenarios, respectively, for $N = \{3, 4, 5\}$ in turn. The median of the resulting worst and best case scenario distributions for each N was then calculated. Table 11.1 depicts the resulting median number of N-node Patterns required to reconstruct the entire minutiae template in the worst and best case scenarios, for $N = \{3, 4, 5\}$.

Table 11.1: Median number of $FC_{360°}$ N-node Patterns needed to reconstruct the entire minutiae template in the worst-case and best-case scenarios.

	$N = 3$	$N = 4$	$N = 5$
Worst Case	14	11	9
Best Case	68,875	2,686,297	102,080,041

There are three important observations that must be drawn from Table 11.1:

- In the worst-case scenario, the number of N-node Patterns required to reconstruct the entire minutiae template *decreases* as the Pattern size, N, increases. This is expected, since a larger Pattern reveals a larger number of minutiae than does a smaller Pattern. Consequently, if each successive Pattern collected by the attacker contains N new minutiae, then, the larger the N-node Pattern, the fewer Patterns are needed to reconstruct the entire minutiae template.
- In the best-case scenario, the number of N-node Patterns needed to reconstruct the entire minutiae template *increases* as the Pattern size, N, increases. This is because a larger Pattern size enables a larger number of N-node Patterns to be constructed from a single fingerprint. Consequently, if an attacker must plough through all but the last $N! - 1$ N-node Patterns possible from a fingerprint, then, the larger the N, the more N-node Patterns are needed to reconstruct the entire minutiae template.
- The difference between the number of N-node Patterns needed in the worst-case scenario and the number of N-node Patterns required in the best-case scenario is extremely large, and this difference increases significantly as the Pattern size, N, increases. This suggests that, while the results in Table 11.1 are useful for estimating the range of the number of N-node Patterns needed, they do not provide an indication of the most likely number of N-node Patterns that we may expect would be required in practice.

To get an indication of the most likely number of **FC$_{360°}$** *N*-node Patterns that would be required in practice to reconstruct the entire minutiae template, the following experiment was conducted:

1. The most likely number of minutiae in the full minutiae template was calculated by computing the median of all the *T* values used in the previous experiment. The median *T* was found to be 42.
2. All the possible *permutations* of *N* integers in the range [1, 42] were established. The permutations represent all the possible ways in which every set of *N* out of *T* minutiae can be ordered to create an *N*-node Pattern. The total number of permutations was $\frac{42!}{(42-N)!}$. This step was conducted separately for each $N = \{3, 4, 5\}$.
3. Permutation sequences from Step 2 were randomly selected for each *N* separately, and care was taken to ensure that a particular permutation sequence was never selected more than once. For each new permutation sequence, the *new* integers (i.e., the ones that did not appear in previously selected permutation sequences) were established and placed into an array. Permutation sequences continued to be randomly selected until the aforementioned array contained all 42 integers. At this point, the total number of selected permutations for each *N* was recorded.
4. Step 3 was repeated 1,000 times for each $N = \{3, 4, 5\}$. The median of the total number of selected permutations was computed separately for each *N*. This number is meant to serve as an estimation of the median number of **FC$_{360°}$** *N*-node Patterns that we may expect would be required in practice to reconstruct an entire minutiae template, assuming that an attacker gains access to a set of randomly selected Patterns. The results are summarised in Table 11.2.

Table 11.2: Median number of FC$_{360°}$ *N*-node Patterns needed to reconstruct the entire minutiae template.

	$N = 3$	$N = 4$	$N = 5$
Median Number of FC$_{360°}$ *N*-node Patterns Required	56	42	33

The results in Table 11.2 indicate that, in general, we may expect an increase in the Pattern size to result in a decrease in the number of **FC$_{360°}$** *N*-node Patterns required to reconstruct the entire minutiae template. Note that these results depict the median number of **FC$_{360°}$** *N*-node Patterns needed to reconstruct the full minutiae template when the attacker collects Patterns of the *same size* only. This may be the case if all of the Patterns come from the same application, in which the Patterns are restricted to a common size, for example. In practice,

however, it may happen that the *N*-node Patterns stolen by an attacker are of mixed sizes. In this case, the number of Patterns required to reconstruct the entire minutiae template would be somewhere between the number of 5-node Patterns required (i.e., 33 in Table 11.2) and the number of 3-node Patterns required (i.e., 56 in Table 11.2).

The results in Table 11.2 are very encouraging. In particular, our findings suggest that an attacker would need to acquire a median number of 56 $\mathbf{FC}_{360°}$ 3-node Patterns or 42 $\mathbf{FC}_{360°}$ 4-node Patterns or 33 $\mathbf{FC}_{360°}$ 5-node Patterns or 33-56 mixed-size $\mathbf{FC}_{360°}$ *N*-node Patterns from the same fingerprint in order to succeed in reconstructing that fingerprint's entire minutiae template. This means that the attacker would either need to break into the same system's database a minimum of 33 and a maximum of 56 times, and figure out which replacement Pattern corresponds to the same fingerprint as the stolen Pattern that they have access to, or else they would need to identify a different Pattern from the same fingerprint used across a minimum of 33 and a maximum of 56 different applications, and then break into each of those system databases to steal the different Patterns. A single database breach would be regarded as a serious failure for any system, so 33 or more breaches of the *same* database may be considered very unlikely to occur in practice. Furthermore, aside from the fact that a regular person is unlikely to be enrolled into such a large number of different applications (particularly using the same fingerprint), the likelihood of an attacker being able to track a person across that many applications can, for practical reasons, be considered low.

While the above results indicate that it would be possible to reconstruct a person's entire minutiae template if enough $\mathbf{FC}_{360°}$ *N*-node Patterns from the same fingerprint are collected, the following points must be emphasized:

- A practical fingerprint recognition system would still benefit more from our proposed fingerprint construct than from the traditional mechanism of storing a user's entire minutiae template in the system's database. This is because, for the traditional storage mechanism, only a single database breach is necessary for an attacker to get their hands on a user's entire minutiae template. Conversely, as shown in Chapter 10, for our proposed fingerprint construct a single stolen *N*-node Pattern would ensure that the majority of the underlying fingerprint's original minutiae template (and thus the corresponding fingerprint image) remains unrecoverable. The results from Table 11.2 suggest that an attacker would need to invest considerably greater effort in recovering a user's full minutiae template when our proposed fingerprint construct is adopted compared to when the traditional fingerprint storage mechanism is used.
- The experimental results above indicate that the resistance of our proposed fingerprint construct to a Record Multiplicity Attack is significantly better than that of the Fuzzy

Vault scheme, which was found to succumb to the attack when only 2 different fuzzy vaults from the same user are accessed by an attacker [175]. Since the Fuzzy Vault scheme is still among the most popular fingerprint template protection schemes in the literature despite this drawback, it seems worthwhile to further mine the potential of our proposed fingerprint construct.

- As was stated in Section 4.5.1, a dedicated analysis of each method's resistance to a Record Multiplicity Attack is currently lacking in the literature pertaining to non-invertible fingerprint template protection schemes. While [209] presented a proof that four commonly-cited non-invertible transforms (i.e., [109, 134, 144, 221]) are susceptible to a Record Multiplicity Attack, that susceptibility has not been quantified. Consequently, it is currently not possible to perform a quantitative comparison between our method and other non-invertible fingerprint template protection schemes in the literature in terms of their resistance to a Record Multiplicity Attack. Nevertheless, the fact that our proposed fingerprint construct uses only a small portion of the entire minutiae template, while other non-invertible fingerprint template protection schemes focus on transforming the entire template, gives us reason to believe that our proposed fingerprint construct would require a larger number of database breaches to collect enough protected fingerprint templates to enable the reconstruction of the original minutiae template. We therefore conclude that our proposed fingerprint construct can be expected to be more resistant to a Record Multiplicity Attack than currently existing non-invertible fingerprint template protection schemes that produce a protected template using the entire minutiae template.
- While it is important to analyse the vulnerability of a fingerprint template protection scheme to a Record Multiplicity Attack in theory, in practice such an attack should be easily avoidable. For example, if a user's reference *N*-node Pattern is stored in the system database along with a user ID, then, in the event of compromise, that ID should be changed along with the user's reference *N*-node Pattern. Similarly, if a user enrols into multiple applications using a different *N*-node Pattern from the same fingerprint in each application, they should certainly avoid employing the same user ID across the different applications. These simple implementation steps would make it more difficult for an attacker to link multiple Patterns generated from the same fingerprint.
- While our analysis in this section has shown that our proposed fingerprint construct is susceptible to a Record Multiplicity Attack, an important advantage of our method lies in the fact that, even if a person's entire fingerprint is known to an attacker, the person can still continue to use that fingerprint for authentication purposes. This is because a large number of Patterns is possible from a single fingerprint; therefore, if our proposed

fingerprint construct is used in the recommended *Two-Factor Authentication* scenario[49], in which a person is required to present both their fingerprint and their chosen reference Pattern for authentication purposes[50], then revelation of their fingerprint to an attacker does not help the attacker deduce the user's Pattern. This may be compared to the PIN number system, for which everyone knows that the chosen PIN must consist of numbers in the range [0, 9], but this does not help an attacker determine the user's PIN.

Overall, the analysis in this section has shown that the resistance of our proposed fingerprint construct to a Record Multiplicity Attack in practice is promising. Section 11.3 demonstrates that this resistance can be strengthened by a simple, yet effective, modification to the $\mathbf{FC}_{360^\circ}$ version of our proposed fingerprint construct.

11.3 MODIFICATION TO $\mathbf{FC}_{360^\circ}$ TO STRENGTHEN ITS RESISTANCE TO A RECORD MULTIPLICITY ATTACK

This section proposes a modification to the $\mathbf{FC}_{360^\circ}$ version of our new fingerprint construct, which strengthens the method's resistance to a Record Multiplicity Attack. Section 11.3.1 describes the modification and explains why it strengthens our method's resistance to a Record Multiplicity Attack. Section 11.3.2 investigates the effect that this modification has on the recognition accuracy of our proposed fingerprint construct.

11.3.1 The Proposed Modification to $\mathbf{FC}_{360^\circ}$

In Section 11.2, it was found that the susceptibility of our proposed fingerprint construct to a Record Multiplicity Attack stems from the fact that the attributes of an $\mathbf{FC}_{360^\circ}$ N-node Pattern's N constituent minutiae, relative to the corresponding fingerprint core, can be recovered from the Pattern's feature vector. Consequently, provided that enough N-node Patterns from the same fingerprint are collected, the recovered minutiae can be easily pieced together to reconstruct the entire minutiae template. This hints at a possible modification to the $\mathbf{FC}_{360^\circ}$ construct, which could lower this vulnerability; namely, we could represent an N-node Pattern in such a way so as to make it impossible to recover the attributes of the Pattern's N constituent minutiae from a single Pattern alone. Such a representation can be achieved by removing the Pattern's global attributes (x, y, ω) and representing the Pattern in

[49] See Section 8.4.3.

[50] See Section 5.2.5.

terms of its local attributes (l, α, β) only. Recall, from Section 5.2.4, that the feature vector, v, of an N-node Pattern has the following form:

$$v = [l_{12}, \alpha_{12}, \beta_{12}, l_{23}, \alpha_{23}, \beta_{23}, \dots, l_{N1}, \alpha_{N1}, \beta_{N1}, x, y, \omega]$$

The modified version of this feature vector, after removing the Pattern's global attributes, would take the following form:

$$v' = [l_{12}, \alpha_{12}, \beta_{12}, l_{23}, \alpha_{23}, \beta_{23}, \dots, l_{N1}, \alpha_{N1}, \beta_{N1}]$$

Consequently, only Equations (11.1) to (11.12) become relevant in the calculation of the Pattern's attributes.

$$l_{12} = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2} \tag{11.1}$$

$$l_{23} = \sqrt{(x_3 - x_2)^2 + (y_3 - y_2)^2} \tag{11.2}$$

$$\vdots$$

$$l_{N1} = \sqrt{(x_1 - x_N)^2 + (y_1 - y_N)^2} \tag{11.3}$$

$$\varphi_{12} = \tan^{-1}\left(\frac{y_2 - y_1}{x_2 - x_1}\right), \quad 0° \le \varphi_{12} < 360° \tag{11.4}$$

$$\varphi_{23} = \tan^{-1}\left(\frac{y_3 - y_2}{x_3 - x_2}\right), \quad 0° \le \varphi_{23} < 360° \tag{11.5}$$

$$\vdots$$

$$\varphi_{N1} = \tan^{-1}\left(\frac{y_1 - y_N}{x_1 - x_N}\right), \quad 0° \le \varphi_{N1} < 360° \tag{11.6}$$

$$\alpha_{12} = \begin{Bmatrix} \varphi_{12} - \theta_1 & , \varphi_{12} \ge \theta_1 \\ \varphi_{12} - \theta_1 + 360° & , \varphi_{12} < \theta_1 \end{Bmatrix}, \quad 0° \le \alpha_{12} < 360° \tag{11.7}$$

$$\alpha_{23} = \begin{Bmatrix} \varphi_{23} - \theta_2 & , \varphi_{23} \ge \theta_2 \\ \varphi_{23} - \theta_2 + 360° & , \varphi_{23} < \theta_2 \end{Bmatrix}, \quad 0° \le \alpha_{23} < 360° \tag{11.8}$$

$$\vdots$$

$$\alpha_{N1} = \begin{Bmatrix} \varphi_{N1} - \theta_N & , \varphi_{N1} \ge \theta_N \\ \varphi_{N1} - \theta_N + 360° & , \varphi_{N1} < \theta_N \end{Bmatrix}, \quad 0° \le \alpha_{N1} < 360° \tag{11.9}$$

$$\beta_{12} = \begin{Bmatrix} \varphi_{12} - \theta_2 & , \varphi_{12} \ge \theta_2 \\ \varphi_{12} - \theta_2 + 360° & , \varphi_{12} < \theta_2 \end{Bmatrix}, \quad 0° \le \beta_{12} < 360° \tag{11.10}$$

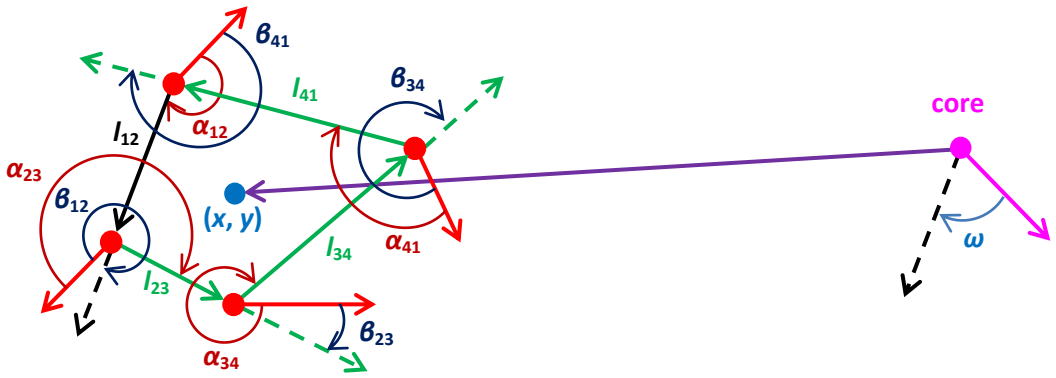
$$\beta_{23} = \begin{Bmatrix} \varphi_{23} - \theta_3 & , \varphi_{23} \ge \theta_3 \\ \varphi_{23} - \theta_3 + 360° & , \varphi_{23} < \theta_3 \end{Bmatrix}, \quad 0° \le \beta_{23} < 360° \tag{11.11}$$

$$\vdots$$

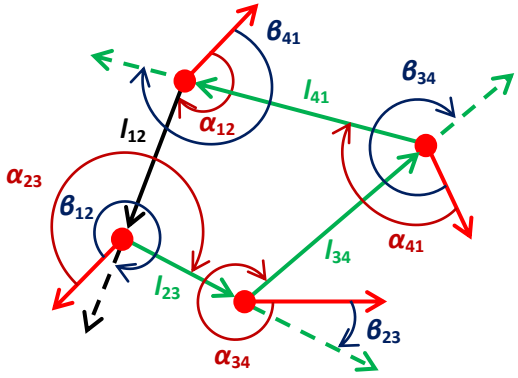
$$\beta_{N1} = \begin{Bmatrix} \varphi_{N1} - \theta_1 & , \varphi_{N1} \ge \theta_1 \\ \varphi_{N1} - \theta_1 + 360° & , \varphi_{N1} < \theta_1 \end{Bmatrix}, \quad 0° \le \beta_{N1} < 360° \tag{11.12}$$

We shall henceforth refer to the fingerprint construct that uses *all* the Pattern attributes (and which is represented by v, above) as *Fixed* $\mathbf{FC}_{360^\circ}$, and the modified construct, which does not use the global Pattern attributes and is represented by v', above, as *Floating* $\mathbf{FC}_{360^\circ}$. The word *fixed* refers to the fact that the *Fixed* $\mathbf{FC}_{360^\circ}$ fingerprint construct results in an N-node Pattern whose location and orientation are *fixed* relative to the location and orientation of the fingerprint core. Conversely, the word *floating* refers to the fact that the *Floating* $\mathbf{FC}_{360^\circ}$ fingerprint construct generates an N-node Pattern whose location and orientation are *not* fixed relative to the core, such that the Pattern's global location and orientation in the fingerprint can essentially take on any value. This means that *Floating* $\mathbf{FC}_{360^\circ}$ is free from core extraction errors in practice (since it does not use the core for generating the N-node Patterns), which is another important advantage over the *Fixed* $\mathbf{FC}_{360^\circ}$ version of our proposed fingerprint construct.

Figure 11.1 illustrates the difference between a *Fixed* $\mathbf{FC}_{360^\circ}$ 4-node Pattern (Figure 11.1 (a)) and a *Floating* $\mathbf{FC}_{360^\circ}$ 4-node Pattern (Figure 11.1 (b)).



(a)



(b)

Figure 11.1: A 4-node Pattern represented via the (a) *Fixed* $\mathbf{FC}_{360^\circ}$ and (b) *Floating* $\mathbf{FC}_{360^\circ}$ versions of our proposed fingerprint construct.

Since *Floating* $\mathbf{FC}_{360°}$ involves representing an N-node Pattern with fewer attributes than *Fixed* $\mathbf{FC}_{360°}$, we may reasonably expect the resulting Patterns to be less unique. Consequently, we may predict that using *Floating* $\mathbf{FC}_{360°}$ would result in a lower FRR and a higher FAR than would the use of *Fixed* $\mathbf{FC}_{360°}$. Section 11.3.2 investigates the recognition accuracy attainable using *Floating* $\mathbf{FC}_{360°}$ and compares it to that attainable using *Fixed* $\mathbf{FC}_{360°}$.

Let us now attempt to use Equations (11.1) to (11.12) to recover the (x, y, θ) attributes of the Pattern's N constituent minutiae relative to the fingerprint core, i.e., we want to recover x_1, x_2, …, x_N, y_1, y_2, …, y_N, θ_1, θ_2, …, θ_N relative to (x_c, y_c, θ_c). Since the attributes of the fingerprint core (i.e., x_c, y_c, and θ_c) are not even used in Equations (11.1) to (11.12), we know straight away that we cannot recover the aforementioned minutiae attributes relative to the fingerprint core, as this information is no longer employed in the Pattern construction process. Nevertheless, this may be easily proven as follows. From observation of Equation (11.1) to Equation (11.12), we can see that we have **$3N$ known values** (l_{12}, l_{23}, …, l_{N1}, α_{12}, α_{23}, …, α_{N1}, β_{12}, β_{23}, …, β_{N1}) and **$4N$ unknown values** (x_1, x_2, .., x_N, y_1, y_2, …, y_N, θ_1, θ_2, …, θ_N, φ_{12}, φ_{23}, …, φ_{N1}). Since there are fewer *knowns* than *unknowns*, we have an undetermined system of equations, which means that it is impossible to determine m_1, m_2, …, m_N relative to the core (or relative to the original image coordinate system) from the Pattern feature vector. This important property of *Floating* $\mathbf{FC}_{360°}$ ensures that an attacker with access to multiple *Floating* $\mathbf{FC}_{360°}$ N-node Patterns from the same fingerprint would find it more difficult (compared to *Fixed* $\mathbf{FC}_{360°}$) to piece the Patterns together to reconstruct the fingerprint's entire minutiae template. For example, an attacker with access to two *Fixed* $\mathbf{FC}_{360°}$ N-node Patterns from the same fingerprint can immediately piece the constituent minutiae of those two Patterns together, since the attributes of the N minutiae recovered from each Pattern will be expressed relative to same reference frame (i.e., the location and orientation of the core point). Alternatively, the two sets of N minutiae recovered from two *Floating* $\mathbf{FC}_{360°}$ N-node Patterns will not be expressed with respect to the same reference frame, since the global location and orientation of these Patterns is not recorded in their corresponding feature vectors. Consequently, the only way that an attacker would be able to piece together the minutiae recovered from two *Floating* $\mathbf{FC}_{360°}$ N-node Pattern feature vectors would be if the two Patterns shared *at least two* of the same minutiae. This is because, in that case, the attacker would be able to use those two minutiae as a common reference frame between the two Patterns, after which the attributes of the two sets of minutiae from the two Patterns could be expressed relative to each other. Consequently, we would expect that a larger number of *Floating* $\mathbf{FC}_{360°}$ N-node Patterns than *Fixed* $\mathbf{FC}_{360°}$ N-node Patterns from the same fingerprint

would be required in order to reconstruct the fingerprint's entire minutiae template, and the resistance of our proposed fingerprint construct to a Record Multiplicity Attack would thus be expected to improve.

To investigate the resistance of the *Floating* $\mathbf{FC}_{360°}$ version of our proposed fingerprint construct to a Record Multiplicity Attack, the experiment in Section 11.2 on the most likely number of Patterns needed to reconstruct a fingerprint's entire minutiae template was repeated for *Floating* $\mathbf{FC}_{360°}$ *N*-node Patterns. Table 11.3 compares the resulting median number of *Floating* $\mathbf{FC}_{360°}$ *N*-node Patterns required to the median number of *Fixed* $\mathbf{FC}_{360°}$ *N*-node Patterns required.

Table 11.3: Median number of *Fixed* $FC_{360°}$ versus *Floating* $FC_{360°}$ *N*-node Patterns needed to reconstruct the entire minutiae template.

	$N = 3$	$N = 4$	$N = 5$
Fixed $\mathbf{FC}_{360°}$	56	42	33
Floating $\mathbf{FC}_{360°}$	463	88	37

From the results in Table 11.3, it is evident that, overall, the *Floating* $\mathbf{FC}_{360°}$ version of our proposed fingerprint construct is indeed more resistant to a Record Multiplicity Attack than the *Fixed* $\mathbf{FC}_{360°}$ version. Interestingly, as the Pattern size, *N*, increases, the number of *Floating* $\mathbf{FC}_{360°}$ *N*-node Patterns needed to reconstruct the entire minutiae template approaches the number of *Fixed* $\mathbf{FC}_{360°}$ *N*-node Patterns required. This may be attributed to the fact that larger Patterns selected from the same fingerprint's minutiae template are more likely to overlap in terms of their constituent minutiae; consequently, it becomes more likely that the Patterns collected by an attacker will share at least two of the same minutiae. Since, earlier, it was explained that an attacker would only be able to piece the minutiae from two *Floating* $\mathbf{FC}_{360°}$ *N*-node Patterns together provided that the two Patterns share at least two minutiae, it makes sense that, the larger the Patterns, the faster the minutiae template reconstruction process becomes. Therefore, we may conclude that the increased resistance to a Record Multiplicity Attack that is offered by the *Floating* $\mathbf{FC}_{360°}$ version of our proposed fingerprint construct would be most beneficial if all Patterns from the same fingerprint consisted of a small number of minutiae (e.g., 3 or 4, based on the results in Table 11.3), or if the Pattern sizes were mixed.

11.3.2 Recognition Accuracy of *Floating* $\mathbf{FC}_{360°}$ versus *Fixed* $\mathbf{FC}_{360°}$

This section evaluates the recognition accuracy attainable by *Floating* $\mathbf{FC}_{360°}$ and compares it to the recognition accuracy of *Fixed* $\mathbf{FC}_{360°}$ established in Section 8.5. To conduct this

investigation, we repeated the experiments from Section 8.5 on the *Floating* **FC$_{360°}$** version of our proposed fingerprint construct. Table 11.4 compares the resulting FAR and FRR for *Floating* **FC$_{360°}$** versus *Fixed* **FC$_{360°}$** under *Single-Factor Authentication*, when the first set of matching thresholds from Chapter 8 (i.e., $\tau_l = 16$, $\tau_{\alpha\beta} = 14°$, $\tau_{loc} = 22$, $\tau_{\omega} = 16°$) was employed for Pattern matching.

Table 11.4: Recognition accuracy of *Fixed* FC$_{360°}$ VS *Floating* FC$_{360°}$ under Single-Factor Authentication, when $\tau_l = 16$, $\tau_{\alpha\beta} = 14°$, $\tau_{loc} = 22$, $\tau_{\omega} = 16°$.

	$N = 3$		$N = 4$		$N = 5$	
	Fixed **FC$_{360°}$**	*Floating* **FC$_{360°}$**	*Fixed* **FC$_{360°}$**	*Floating* **FC$_{360°}$**	*Fixed* **FC$_{360°}$**	*Floating* **FC$_{360°}$**
FAR (%)	0.86	11.48	0.16	1.29	0.03	0.18
FRR (%)	2.61	1.20	3.46	2.57	4.10	3.38

Table 11.5 compares the resulting FAR and FRR for *Floating* **FC$_{360°}$** versus *Fixed* **FC$_{360°}$** under *Single-Factor Authentication*, when the second set of matching thresholds from Chapter 8 (i.e., $\tau_l = 20$, $\tau_{\alpha\beta} = 18°$, $\tau_{loc} = 26$, $\tau_{\omega} = 23°$) was employed for Pattern matching.

Table 11.5: Recognition accuracy of *Fixed* FC$_{360°}$ VS *Floating* FC$_{360°}$ under Single-Factor Authentication, when $\tau_l = 20$, $\tau_{\alpha\beta} = 18°$, $\tau_{loc} = 26$, $\tau_{\omega} = 23°$.

	$N = 3$		$N = 4$		$N = 5$	
	Fixed **FC$_{360°}$**	*Floating* **FC$_{360°}$**	*Fixed* **FC$_{360°}$**	*Floating* **FC$_{360°}$**	*Fixed* **FC$_{360°}$**	*Floating* **FC$_{360°}$**
FAR (%)	3.42	27.00	1.08	5.82	0.36	1.40
FRR (%)	1.50	0.71	2.17	1.48	2.63	2.18

Table 11.6 compares the resulting FAR and FRR for *Floating* **FC$_{360°}$** versus *Fixed* **FC$_{360°}$** under *Two-Factor Authentication*, when the first set of matching thresholds from Chapter 8 (i.e., $\tau_l = 16$, $\tau_{\alpha\beta} = 14°$, $\tau_{loc} = 22$, $\tau_{\omega} = 16°$) was employed for Pattern matching.

Table 11.6: Recognition accuracy of *Fixed* FC$_{360°}$ VS *Floating* FC$_{360°}$ under Two-Factor Authentication, when $\tau_l = 16$, $\tau_{\alpha\beta} = 14°$, $\tau_{loc} = 22$, $\tau_{\omega} = 16°$.

	$N = 3$		$N = 4$		$N = 5$	
	Fixed **FC$_{360°}$**	*Floating* **FC$_{360°}$**	*Fixed* **FC$_{360°}$**	*Floating* **FC$_{360°}$**	*Fixed* **FC$_{360°}$**	*Floating* **FC$_{360°}$**
FAR (%)	1.29×10^{-5}	1.82×10^{-4}	5.75×10^{-8}	4.43×10^{-7}	2.93×10^{-10}	1.59×10^{-9}
FRR (%)	2.61	1.20	3.46	2.57	4.10	3.38

Table 11.7 compares the resulting FAR and FRR for *Floating* **FC$_{360°}$** versus *Fixed* **FC$_{360°}$** under *Two-Factor Authentication*, when the second set of matching thresholds from Chapter 8 (i.e., $\tau_l = 20$, $\tau_{\alpha\beta} = 18°$, $\tau_{loc} = 26$, $\tau_{\omega} = 23°$) was employed for Pattern matching.

Table 11.7: Recognition accuracy of *Fixed* FC$_{360°}$ VS *Floating* FC$_{360°}$ under Two-Factor Authentication, when $\tau_l = 20$, $\tau_{\alpha\beta} = 18°$, $\tau_{loc} = 26$, $\tau_{\omega} = 23°$.

	N = 3		*N* = 4		*N* = 5	
	Fixed **FC$_{360°}$**	*Floating* **FC$_{360°}$**	*Fixed* **FC$_{360°}$**	*Floating* **FC$_{360°}$**	*Fixed* **FC$_{360°}$**	*Floating* **FC$_{360°}$**
FAR (%)	1.29×10^{-5}	6.35×10^{-4}	5.75×10^{-8}	2.83×10^{-6}	2.93×10^{-10}	1.67×10^{-8}
FRR (%)	2.61	0.71	3.46	1.48	4.10	2.18

From Tables 11.4 to 11.7, we can see that, regardless of the Pattern size (*N*), the *Floating* **FC$_{360°}$** version of our fingerprint construct produces a higher FAR and a lower FRR than the *Fixed* **FC$_{360°}$** version. As suggested in Section 11.3.1, this trend is due to the fact that *Floating* **FC$_{360°}$** involves the representation of an *N*-node Pattern using fewer attributes, so the resulting Pattern is less discriminative than that produced using *Fixed* **FC$_{360°}$**. Consequently, it becomes easier for a matching *N*-node Pattern to be found both in an impostor's fingerprint and in a query sample of the genuine user's fingerprint.

While the decrease in the FRR is good for ensuring that a genuine user is inconvenienced as little as possible, the increase in the FAR is not ideal for preventing impostor access to the recognition system. This latter point is especially important considering the FARs of 11.48% and 27.00%, obtained for *Floating* **FC$_{360°}$** 3-node Patterns in Tables 11.4 and 11.5, respectively, which are rather high (particularly the FAR of 27.00%). Having said this, the following points should be noted:

- While the FARs for *Floating* **FC$_{360°}$** 3-node Patterns in Tables 11.4 and 11.5 are quite high, the FAR of 11.48% would be acceptable in a low-security application, in which impostor access is unlikely. Furthermore, the FARs for 4- and 5-node *Floating* **FC$_{360°}$** Patterns in the same two tables would be acceptable in practice.
- The results in Tables 11.4 and 11.5 correspond to the *Single-Factor Authentication* scenario, which we do not recommend be used in practice[51], or the *Two-Factor Authentication* scenario in which an attacker knows a genuine user's reference *N*-node Pattern. In practice, however, the norm would be for an attacker *not* to know a genuine

[51] This recommendation was initially made in Section 8.4.3.

user's reference Pattern, which means that the FAR in *Two-Factor Authentication* would be considerably lower. Indeed, considering Tables 11.6 and 11.7, we can see that the FARs obtained for both the *Fixed* $\mathbf{FC}_{360°}$ and *Floating* $\mathbf{FC}_{360°}$ versions of our proposed fingerprint construct at all Pattern sizes indicate that the probability of an attacker succeeding in guessing a genuine user's Pattern from their own fingerprint may be expected to be extremely low. We may thus conclude that, if our proposed fingerprint construct is implemented in the recommended *Two-Factor Authentication* scenario, in normal operating conditions both the *Fixed* $\mathbf{FC}_{360°}$ and *Floating* $\mathbf{FC}_{360°}$ versions would ensure a high level of security in terms of rejecting impostors.

- The FAR can be further tuned by determining the optimal matching thresholds and Pattern size to suit the requirements of a particular application.

11.4 SUMMARY

This chapter analysed the vulnerability of our proposed fingerprint construct, $\mathbf{FC}_{360°}$, to a Record Multiplicity Attack, in terms of the possibility of reconstructing a fingerprint's entire minutiae template by piecing together the *N* minutiae recovered from multiple *N*-node Patterns originating from the same fingerprint.

We conducted an investigation into the number of *N*-node Patterns that an attacker would need to collect in order to be able to reconstruct a fingerprint's entire minutiae template. It was determined that this number may be expected to be sufficiently large to ensure that our proposed fingerprint construct will resist a Record Multiplicity Attack in practice. A lack of numerical analysis on the susceptibility of other non-invertible fingerprint template protection schemes to a Record Multiplicity Attack in the literature means that we cannot presently conduct a quantitative comparison to the resistance of $\mathbf{FC}_{360°}$. The fact that our fingerprint construct uses only a small portion of the entire minutiae template, however, gives us reason to believe that it is more resistant to a Record Multiplicity Attack than fingerprint template protection schemes that use the entire minutiae template in generating the protected fingerprint template (i.e., *correlation* in the latter case is more likely).

We then proposed a modification to our fingerprint construct, which further increases its resistance to a Record Multiplicity Attack. The modified construct was termed *Floating* $\mathbf{FC}_{360°}$ and the original[52] version was re-named *Fixed* $\mathbf{FC}_{360°}$. An evaluation of the recognition accuracy attainable by *Floating* $\mathbf{FC}_{360°}$ demonstrated that *Floating* $\mathbf{FC}_{360°}$ *N*-node

[52] In this context, *original* refers to the original version of $\mathbf{FC}_{360°}$, *not* $\mathbf{FC}_{180°}$.

Patterns are generally less discriminative than *Fixed* $\mathbf{FC}_{360^\circ}$ *N*-node Patterns; however, appropriate selection of the matching thresholds and the most suitable Pattern size for the requirements of a particular application would ensure that the recognition accuracy of both the *Fixed* $\mathbf{FC}_{360^\circ}$ and *Floating* $\mathbf{FC}_{360^\circ}$ versions of our proposed fingerprint construct are acceptable in practice. This would be the case particularly if the system designers heed our recommendation of adopting *Two-Factor Authentication* in favour of *Single-Factor Authentication*.

Overall, this chapter concludes the analysis pertaining to the *non-invertibility* of our proposed fingerprint construct on a high note.

Chapter 12

Cancellability and Diversity of $FC_{360°}$

The analysis in Chapters 5 through to 9 showed that our new fingerprint construct is capable of effectively discriminating between a genuine user and an impostor in cooperative-user scenarios in practice, particularly when the improved version, $\mathbf{FC_{360°}}$, is adopted. Chapter 10 provided strong evidence to suggest that $\mathbf{FC_{360°}}$ is *non-invertible*, and the analysis in Chapter 11 demonstrated that the resistance of $\mathbf{FC_{360°}}$ to a Record Multiplicity Attack may be expected to be high in practice, especially when the *Floating* $\mathbf{FC_{360°}}$ version is used in favour of the original *Fixed* $\mathbf{FC_{360°}}$. This chapter evaluates the ability of both the *Fixed* $\mathbf{FC_{360°}}$ and *Floating* $\mathbf{FC_{360°}}$ versions of our proposed fingerprint construct to satisfy the *cancellability* and *diversity* characteristics of an ideal fingerprint template protection scheme.

12.1 INTRODUCTION

As mentioned in our literature review on non-invertible fingerprint template protection schemes in Chapter 4, a fingerprint template protection scheme is considered to be *cancellable* if it is possible to replace a compromised template with a *different* template generated from the same fingerprint. A fingerprint template protection scheme is considered to satisfy the *diversity* property if it is capable of generating multiple uncorrelated templates from the same fingerprint, in order to enable a person to enrol into different applications using the same fingerprint without the risk of being cross-matched across those applications' databases. Since *diversity* effectively enables *cancellability*, these properties are generally considered to be synonymous in the literature, and thus the same analysis is usually applied towards confirming that a fingerprint template protection scheme is both *cancellable* and *diverse*.

As stated in Sections 4.5.2 and 4.5.3, our perusal of non-invertible fingerprint template protection schemes in the literature revealed that two main approaches are generally adopted for analysing the *cancellability* and *diversity* properties of a fingerprint template protection scheme. The first approach operates under the assumption that the ability of a fingerprint

template protection scheme to satisfy these two properties is self-explanatory, thereby simply stating that *cancellability* and *diversity* are easily achieved by changing the transformation and/or some associated external parameters. The second, more common approach, involves evaluating a method's *cancellability* and *diversity* by generating multiple protected templates from the *same* fingerprint and then attempting to match them. A low match score serves as evidence that the fingerprint template protection scheme is capable of establishing different identities from the same fingerprint, thereby confirming the scheme's ability to satisfy the *cancellability* and *diversity* requirements. While it would be useful to be able to quantify *cancellability* and *diversity* in terms of the number of different templates that can be generated from the same fingerprint, such an analysis is extremely rare in the literature[53].

This chapter analyses the *cancellability* and *diversity* of the *Fixed* $\mathbf{FC}_{360^\circ}$ and *Floating* $\mathbf{FC}_{360^\circ}$ versions of our new fingerprint construct in terms of the latter two evaluation techniques discussed above. However, we believe that, although similar, *cancellability* and *diversity* should be analysed in slightly different ways. We thus adopt the following definitions for the *cancellability* and *diversity* properties. We consider *cancellability* to pertain to the possibility of generating a *different N*-node Pattern from the same fingerprint in the event that a user's initial reference *N*-node Pattern is compromised (e.g., stolen or modified by an attacker). In this case, *different* implies non-matching. On the other hand, *diversity* is considered to correspond to the possibility of generating *different* and *unlinkable N*-node Patterns from the same fingerprint, so that a person cannot be tracked across the different applications they are enrolled in with those templates. The notion of *diversity* is thus defined in terms of the *cancellability* requirement plus the additional condition that multiple templates generated from the same fingerprint cannot be linked to each other in a way other than by a direct match.

We begin by evaluating the *cancellability* of the *Fixed* $\mathbf{FC}_{360^\circ}$ and *Floating* $\mathbf{FC}_{360^\circ}$ *N*-node Patterns generated using our proposed fingerprint template protection scheme, and this is followed by the *diversity* evaluation. Recall that, based on our analysis in Chapter 8, we recommended that *Two-Factor Authentication* be adopted instead of *Single-Factor Authentication* for our proposed fingerprint construct in practice. The analysis in this chapter assumes that this recommendation has been heeded.

[53] For more details on this point, see Section 4.5.2.

12.2 CANCELLABILLITY OF *FIXED* $FC_{360°}$ AND *FLOATING* $FC_{360°}$ *N*-NODE PATTERNS

This section evaluates the *cancellability* of a *Fixed* $\mathbf{FC_{360°}}$ and a *Floating* $\mathbf{FC_{360°}}$ *N*-node Pattern. Section 12.2.1 evaluates cancellability in terms of the probability that two different *N*-node Patterns originating from the same fingerprint will match. Section 12.2.2 quantifies cancellability in terms of the number of different *N*-node Patterns possible from a single fingerprint.

12.2.1 Cancellability Analysis 1: Probability of Two *N*-node Patterns from the Same Fingerprint Matching

In order for a compromised Pattern to be *cancellable*, it must be possible to replace the compromised *N*-node Pattern with a *different* *N*-node Pattern from the same fingerprint. Technically, *every* *N*-node Pattern possible from the minutiae in a fingerprint is *different* to every other Pattern in the same fingerprint. While this is true in theory, the use of matching thresholds for Pattern matching in practice (see Section 5.2.5) may occasionally make it possible that a replacement *N*-node Pattern matches a compromised *N*-node Pattern from the same fingerprint. We must, therefore, refine our definition so that it reads as follows: Two *N*-node Patterns are considered to be *different* if they do not match, where a match is determined based on the guidelines established in Section 5.2.5. These guidelines state that, in order for two *N*-node Patterns to match, the following conditions must be satisfied:

1. The Patterns must consist of the same number of nodes (minutiae). If we let N_1 denote the size of the first *N*-node Pattern and N_2 denote the size of the second *N*-node Pattern, then, in order for a match to be possible, $N_1 = N_2$ must hold.
2. *All* of the Pattern attributes in the first Pattern must match the corresponding attributes in the second Pattern. This means that:
    a. A partial match is not considered a match.
    b. The order of the Pattern attributes in the corresponding feature vector is important, which means that the same minutiae connected in a different order would constitute a different Pattern.

For an *N*-node Pattern to be considered *cancellable* in practice, therefore, the probability of it matching another *N*-node Pattern in the same fingerprint must be very low. To evaluate the *cancellability* of a *Fixed* $\mathbf{FC_{360°}}$ and a *Floating* $\mathbf{FC_{360°}}$ *N*-node Pattern in practice, the following experiment was conducted:

1. In essentially the same way as for Experiment 3b in Section 8.4.3, 100 different *Fixed* $\mathbf{FC}_{360^\circ}$ and 100 different *Floating* $\mathbf{FC}_{360^\circ}$ *N*-node Patterns were randomly generated for each person in our cooperative-user fingerprint database.
2. For every person, each of their 100 *Fixed* $\mathbf{FC}_{360^\circ}$ *N*-node Patterns was compared to each of the remaining 99 *Fixed* $\mathbf{FC}_{360^\circ}$ *N*-node Patterns **from the same person**, and each of the person's 100 *Floating* $\mathbf{FC}_{360^\circ}$ *N*-node Patterns was similarly compared to each of the remaining 99 *Floating* $\mathbf{FC}_{360^\circ}$ *N*-node Patterns **from the same person.** Therefore, in total, 990,000 Pattern comparisons (100 Reference Patterns × 99 Query Patterns × 100 People) were conducted for each of the two versions of our proposed fingerprint construct. Let *M* denote the total number of matches. Then, the probability of a replacement *N*-node Pattern matching a compromised *N*-node Pattern, when both Patterns come from the same fingerprint, was calculated using Equation (12.1):

$$P(match) = \frac{M}{990{,}000} \tag{12.1}$$

3. Steps 1-2 were repeated 100 times, and the average probability of a replacement *N*-node Pattern matching a compromised *N*-node Pattern was computed separately for the *Fixed* $\mathbf{FC}_{360^\circ}$ and *Floating* $\mathbf{FC}_{360^\circ}$ versions of our proposed fingerprint construct.

Note that, in this experiment, 100 *N*-node Patterns were used in order to calculate the probability of a replacement *N*-node Pattern matching a compromised *N*-node Pattern from the same fingerprint when a person's Pattern is compromised 99 times. Since such a large number of Pattern cancellations should *never* be needed in practice[54], the aim of the experiment was to show that the number of cancellations provided by our proposed fingerprint construct should be more than sufficient for practical purposes.

Steps 1-3 were performed for $N = \{3, 4, 5\}$ and for each of the two sets of matching thresholds established in Section 8.4.1, in turn. Table 12.1 and Table 12.2 depict the resulting *average* probability of a replacement *Fixed* $\mathbf{FC}_{360^\circ}$ *N*-node Pattern matching a compromised *Fixed* $\mathbf{FC}_{360^\circ}$ *N*-node Pattern from the same fingerprint and the *average* probability of a replacement *Floating* $\mathbf{FC}_{360^\circ}$ *N*-node Pattern matching a compromised *Floating* $\mathbf{FC}_{360^\circ}$ *N*-

[54] Any practical system that stores its users' personal information in a database would be expected to incorporate a sufficiently high level of security into the system, such that a database breach is extremely unlikely. This is because a *single* database breach in practice would indicate a lack of concern for the users' privacy, which would reflect badly on the corresponding system and would thus discourage people from using it.

node Pattern from the same fingerprint, for each $N = \{3, 4, 5\}$ at each of the two sets of matching thresholds, respectively.

Table 12.1: Probability of a replacement *N*-node Pattern matching a compromised *N*-node Pattern from the same fingerprint, when $\tau_l = 16$, $\tau_{\alpha\beta} = 14°$, $\tau_{loc} = 22$, $\tau_\omega = 16°$.

	$N = 3$	$N = 4$	$N = 5$
Fixed $\mathbf{FC}_{360°}$	9.01×10^{-5}	1.58×10^{-5}	3.33×10^{-6}
Floating $\mathbf{FC}_{360°}$	9.96×10^{-5}	1.60×10^{-5}	3.33×10^{-6}

Table 12.2: Probability of a replacement *N*-node Pattern matching a compromised *N*-node Pattern from the same fingerprint, when $\tau_l = 20$, $\tau_{\alpha\beta} = 18°$, $\tau_{loc} = 26$, $\tau_\omega = 23°$.

	$N = 3$	$N = 4$	$N = 5$
Fixed $\mathbf{FC}_{360°}$	2.37×10^{-4}	4.62×10^{-5}	1.22×10^{-5}
Floating $\mathbf{FC}_{360°}$	2.67×10^{-4}	4.68×10^{-5}	1.22×10^{-5}

We begin by making the following general observations from Table 12.1 and Table 12.2:

- The probability of a replacement *N*-node Pattern matching a compromised *N*-node Pattern from the same fingerprint, for both the *Fixed* $\mathbf{FC}_{360°}$ and *Floating* $\mathbf{FC}_{360°}$ versions of our proposed fingerprint construct, is extremely small, provided that the matching thresholds are sensibly selected.
- The probability of a replacement *N*-node Pattern matching a compromised *N*-node Pattern from the same fingerprint increases as the matching thresholds increase. This is expected, since the use of larger matching thresholds effectively reduces Pattern *uniqueness*, thus making it more likely that two different Patterns from the same fingerprint will match.
- The probability of a replacement *N*-node Pattern matching a compromised *N*-node Pattern from the same fingerprint decreases as the Pattern size, *N*, increases. This makes sense, because a larger Pattern is more discriminative than a smaller Pattern, which makes it less likely that two larger Patterns within the same fingerprint will match.
- The probability of a replacement *Fixed* $\mathbf{FC}_{360°}$ *N*-node Pattern matching a compromised *Fixed* $\mathbf{FC}_{360°}$ *N*-node Pattern from the same fingerprint is smaller than the probability of a replacement *Floating* $\mathbf{FC}_{360°}$ *N*-node Pattern matching a compromised *Floating* $\mathbf{FC}_{360°}$ *N*-node Pattern from the same fingerprint. This is due to the fact that a *Floating* $\mathbf{FC}_{360°}$ *N*-node Pattern consists of fewer Pattern attributes than a *Fixed* $\mathbf{FC}_{360°}$ *N*-node Pattern, which makes the former Pattern type less discriminative than the latter Pattern type. Nevertheless, our results in Tables 12.1 and 12.2 indicate that the difference in the probabilities resulting from these two versions of our proposed fingerprint construct is

fairly insignificant in terms of the overall probability, and the difference becomes less significant as the Pattern size, *N*, increases.

Since a partial Pattern match would not count as a match in practice (see the matching guidelines established in Section 5.2.5), we can round the values in Table 12.1 and Table 12.2 down to the nearest whole number. We may then draw the following conclusions regarding the *cancellability* of our proposed fingerprint construct in practice when the *Two-Factor Authentication* method is used:

- There is a 9 in 100,000 chance of a replacement 3-node Pattern matching a compromised 3-node Pattern from the same fingerprint when the first set of matching thresholds is used, and a 2 in 10,000 chance when the second set of thresholds is used, regardless of whether the *Fixed* $\mathbf{FC_{360°}}$ or the *Floating* $\mathbf{FC_{360°}}$ version of our proposed fingerprint construct is adopted. This suggests that a person's Pattern would need to be compromised over 11,000 times[55] when the first set of matching thresholds is used, and about 5,000 times when the second set of matching thresholds is used, in order for there to be a chance that the replacement 3-node Pattern would match the compromised 3-node Pattern. Since such a scenario should be extremely unlikely in practice[56], the probability of a replacement 3-node Pattern matching a compromised 3-node Pattern from the same fingerprint should effectively be zero in practice.
- There is a 1 in 100,000 chance of a replacement 4-node Pattern matching a compromised 4-node Pattern from the same fingerprint when the first set of matching thresholds is used, and a 4 in 100,000 chance when the second set of thresholds is used, regardless of whether the *Fixed* $\mathbf{FC_{360°}}$ or the *Floating* $\mathbf{FC_{360°}}$ version of our proposed fingerprint construct is adopted. This implies that a person's Pattern would need to be compromised about 100,000 times when the first set of matching thresholds is used, and about 25,000 times when the second set of matching thresholds is used, in order for there to be a chance that the replacement 4-node Pattern would match the compromised 4-node Pattern. As such a scenario should be extremely unlikely in practice, we may conclude that the probability of a replacement 4-node Pattern matching a compromised 4-node Pattern from the same fingerprint should effectively be zero in practice.

[55] Let x = number of times that a person's Pattern is compromised. Then $\frac{9}{100{,}000}x = 1 \therefore x = 11{,}111.\dot{1}$

[56] Even if a user were required to replace their Pattern every 6 months for security purposes, it would take 2,500 years for the number of Pattern replacements to reach 5,000, and 5,500 years for the number of replacements to reach 11,000.

- There is a 3 in a million chance of a replacement 5-node Pattern matching a compromised 5-node Pattern from the same fingerprint when the first set of matching thresholds is used, and a 1 in 100,000 chance when the second set of thresholds is used, regardless of whether the *Fixed* $\mathbf{FC}_{360°}$ or the *Floating* $\mathbf{FC}_{360°}$ version of our proposed fingerprint construct is adopted. This indicates that a person's Pattern would need to be compromised over 333,000 times when the first set of matching thresholds is used, and about 100,000 times when the second set of matching thresholds is used, in order for there to be a chance that the replacement 5-node Pattern would match the compromised 5-node Pattern. Since such a large number of Pattern compromises should never occur in practice, we may reasonably conclude that the probability of a replacement 5-node Pattern matching a compromised 5-node Pattern from the same fingerprint should effectively be zero in practice.

Our results in Table 12.1 and Table 12.2 thus provide encouraging evidence to support the fact that an *N*-node Pattern generated using either the *Fixed* $\mathbf{FC}_{360°}$ or the *Floating* $\mathbf{FC}_{360°}$ version of our proposed fingerprint construct may be expected to be *cancellable* in practice. Section 12.2.2 considers the *number* of different *N*-node Patterns available in a fingerprint, which provides insight into the expected number of times that a person would be able to replace a compromised *N*-node Pattern in practice.

12.2.2 Cancellability Analysis 2: Number of Different *N*-node Patterns Available in a Fingerprint

The *true* number of different *N*-node Patterns available in a single fingerprint is equal to $\frac{T!}{(T-N)!}$, where *T* denotes the total number of minutiae in the fingerprint's entire minutiae template, and *N* denotes the number of minutiae used to construct a single *N*-node Pattern. Consequently, if a user's reference *N*-node Pattern is stolen from the database, they effectively have $\frac{T!}{(T-N)!} - 1$ replacement *N*-node Patterns from the same fingerprint to choose from. Using the median *T* value of 42, which was established in Chapter 11 for our cooperative-user fingerprint database, Equation (12.3) was used to estimate the median *true* number of *N*-node Patterns that we may expect would be available in a single fingerprint in practice.

$$True\ Number\ of\ Patterns = \frac{T!}{(T-N)!} \tag{12.3}$$

Table 12.3 summarises the results.

Table 12.3: Estimation of the median true number of *N*-node Patterns available in a single fingerprint.

	$N = 3$	$N = 4$	$N = 5$	Total
True Number of *N*-node Patterns in a Single Fingerprint	68,880	2,686,320	102,080,160	104,835,360

The results in Table 12.3 indicate that:

- If a 3-node Pattern is compromised, the median *true* number of 3-node Pattern replacements available in the same fingerprint would be 68,799. This is 68,799 times more cancellability than that offered by the traditional mechanism of representing a fingerprint by its entire minutiae template.
- If a 4-node Pattern is compromised, the median *true* number of 4-node Pattern replacements available in the same fingerprint would be 2,686,319. This is 2,686,319 times more cancellability than that offered by the traditional mechanism of representing a fingerprint by its entire minutiae template.
- If a 5-node Pattern is compromised, the median *true* number of 5-node Pattern replacements available in the same fingerprint would be 102,080,159. This is 102,080,159 times more cancellability than that offered by the traditional mechanism of representing a fingerprint by its entire minutiae template.
- If an application does not insist on one specific Pattern size, then the total number of *N*-node Patterns available in a single fingerprint is equal to the total number of 3-node *and* 4-node *and* 5-node Patterns. In this case, if a person's reference *N*-node Pattern is compromised, the results in Table 12.3 indicate that the median number of mixed-size Pattern replacements would be 104,835,359. This is 104,835,359 times more cancellability than that offered by the traditional mechanism of representing a fingerprint by its entire minutiae template.

From Table 12.3, it is evident that the number of *N*-node Patterns available in a fingerprint increases significantly as the Pattern size increases. This is because the number of possible ways of ordering a set of N minutiae increases by $(N+1)! - N! = (N+1)N! - N! = N!\,(N+1-1) = N! \times N$ with every increase in N by 1. Having made this observation, it is important to emphasize that even the true number of 3-node Patterns in Table 12.3 should be more than sufficient for cancellability purposes in practice, since no one's reference Pattern should be compromised anywhere near 68,880 times!

The results in Table 12.3 provide an estimation of the median *true* number of *N*-node Patterns available in a single fingerprint. This is based on the assumption that every *N*-node

Pattern possible from a fingerprint's minutiae is different to every other Pattern in the same fingerprint. However, as indicated in the analysis in Section 12.2.1, in practice the total number of *different* *N*-node Patterns available in a fingerprint may not be equal to the *true* number, due to the use of matching thresholds. Let *P* denote the probability of a replacement *N*-node Pattern matching a compromised *N*-node Pattern for some set of matching thresholds, and let *U* denote the true number of *N*-node Patterns available in a fingerprint. Then the *practical* number of different *N*-node Patterns available in a fingerprint may be calculated using Equation (12.4):

$$Practical\ Number\ of\ Patterns = U - \lfloor PU \rfloor = \frac{T!}{(T-N)!} - \left\lfloor P \times \frac{T!}{(T-N)!} \right\rfloor \qquad (12.4)$$

Using each value from Table 12.1 and Table 12.2 as *P* in turn, and combining it with the *U* value corresponding to the same Pattern size, *N*, from Table 12.3, Equation (12.4) was used to estimate the median *practical* number of *N*-node Patterns available in a fingerprint. Tables 12.4 and 12.5 summarise the results for the first and second sets of matching thresholds from Section 12.2.1, respectively.

Table 12.4: Median *practical* number of different *N*-node Patterns available in a fingerprint, when $\tau_l = 16$, $\tau_{\alpha\beta} = 14°$, $\tau_{loc} = 22$, $\tau_\omega = 16°$.

	$N = 3$	$N = 4$	$N = 5$
Fixed $FC_{360°}$	68,874	2,686,278	104,835,011
Floating $FC_{360°}$	68,874	2,686,278	104,835,011

Table 12.5: Median *practical* number of different *N*-node Patterns available in a fingerprint, when $\tau_l = 20$, $\tau_{\alpha\beta} = 18°$, $\tau_{loc} = 26$, $\tau_\omega = 23°$.

	$N = 3$	$N = 4$	$N = 5$
Fixed $FC_{360°}$	68,864	2,686,196	104,834,083
Floating $FC_{360°}$	68,862	2,686,195	104,834,079

The most important observations to be drawn from Tables 12.4 and 12.5 are the following:

- The *practical* number of *N*-node Patterns available in a fingerprint is smaller than the *true* number of available Patterns. The larger the matching thresholds, the greater the difference between the *true* and *practical* numbers of available *N*-node Patterns. This is because the use of larger matching thresholds makes it easier for two *N*-node Patterns to match, thereby effectively decreasing Pattern *uniqueness*.

- Although smaller than the *true* number of *N*-node Patterns, the *practical* number of *N*-node Patterns available in a fingerprint is nevertheless very large for all the considered Pattern sizes. This indicates that, provided that the matching thresholds are sensibly selected (i.e., not excessively large), the number of Pattern replacements available in practice should be much larger than needed. For example, no one's Pattern should be compromised anywhere near even 100 times in practice, let alone over 68,000 times!
- The larger the Pattern size, *N*, the larger the *practical* number of *N*-node Patterns available. This is due to the fact that there are more ways to order a larger number of minutiae than a smaller number of minutiae, so, the larger the Pattern, the greater the *true*, and therefore *practical*, number of *N*-node Patterns available in a fingerprint.
- The *practical* number of *N*-node Patterns available in a fingerprint is the same for both the *Fixed* $\mathbf{FC}_{360^\circ}$ and *Floating* $\mathbf{FC}_{360^\circ}$ versions of our proposed fingerprint construct in Table 12.4, but is smaller for the *Floating* $\mathbf{FC}_{360^\circ}$ version than the *Fixed* $\mathbf{FC}_{360^\circ}$ version in Table 12.5. Note that, as observed in Section 11.3 and Section 12.2.1, the fact that a *Floating* $\mathbf{FC}_{360^\circ}$ *N*-node Pattern consists of fewer attributes than a *Fixed* $\mathbf{FC}_{360^\circ}$ *N*-node Pattern of the same size (i.e., *N*) means that *Floating* $\mathbf{FC}_{360^\circ}$ Patterns tend to be less discriminative (i.e., *unique*) than *Fixed* $\mathbf{FC}_{360^\circ}$ Patterns; consequently, it is generally easier for two *Floating* $\mathbf{FC}_{360^\circ}$ *N*-node Patterns to match. While this trend is evident in Table 12.5, the results in Table 12.4 indicate no difference between the two versions. We may thus conclude that the difference between the practical number of *Fixed* $\mathbf{FC}_{360^\circ}$ *N*-node Patterns and *Floating* $\mathbf{FC}_{360^\circ}$ *N*-node Patterns available in a fingerprint would depend on the adopted matching thresholds in practice. In any case, however, the results in Table 12.5 indicate that, despite a possible difference between the two versions of our proposed fingerprint construct in this respect, both the practical number of different *Fixed* $\mathbf{FC}_{360^\circ}$ *N*-node Patterns and the practical number of *Floating* $\mathbf{FC}_{360^\circ}$ *N*-node Patterns available in a fingerprint should nevertheless be much larger than needed in practice.
- Overall, the results in Table 12.4 and Table 12.5 indicate that the median number of times that we may expect to be able to replace a compromised *N*-node Pattern in practice is more than sufficient, even for 3-node Patterns, because such a large number of Pattern compromises should never occur in a fingerprint recognition system operating in practice.

Our results in Tables 12.4 and 12.5 are thus extremely encouraging, suggesting that the *cancellability* of both the *Fixed* $\mathbf{FC}_{360^\circ}$ and *Floating* $\mathbf{FC}_{360^\circ}$ versions of our proposed fingerprint construct would be very satisfactory in practice, in terms of the large number of replacement Patterns possible from a single fingerprint. Furthermore, our analysis has shown

that the number of available *N*-node Patterns can be controlled to some extent by selecting appropriate matching thresholds and by choosing the most suitable Pattern size[57]. This analysis was based on defining *cancellability* in terms of the requirement that the replacement Pattern be *different* from the compromised Pattern, which was shown to be decided by matching thresholds in practice. While this is certainly a valid approach, and indeed the approach generally adopted for *cancellability* analysis in the associated literature[58], we take the investigation a step further. In particular, we consider whether each possible replacement Pattern is *safe* in terms of the ease with which an attacker can use the compromised Pattern to guess the replacement Pattern.

For this analysis, it is assumed that a replacement *N*-node Pattern can only be figured out from a compromised *N*-node Pattern if the *N* minutiae used to generate the replacement Pattern are the *same N* minutiae that were used to generate the compromised Pattern[59]. The guessing of a replacement Pattern from the compromised Pattern is thus assumed to involve trying out different ways to arrange the minutiae recovered from the compromised Pattern[60] (so, the guessing of one or more non-recovered minutiae is not considered in this analysis). If multiple Patterns from the same person have been compromised, then we assume the worst-case scenario in which the *same* attacker has gained access to all of those compromised Patterns. In this case, it is assumed that the attacker is able to piece those Patterns together to recover their constituent minutiae, as shown by our analysis in Chapter 11. It is important to keep in mind that, as demonstrated by the analysis in Chapter 11, it is more difficult to piece together multiple *Floating* $\mathbf{FC}_{360^\circ}$ *N*-node Patterns from the same fingerprint than it is to piece together multiple *Fixed* $\mathbf{FC}_{360^\circ}$ *N*-node Patterns. In this analysis, however, we shall not specify whether the compromised Pattern(s) are of the *Fixed* $\mathbf{FC}_{360^\circ}$ type or the *Floating* $\mathbf{FC}_{360^\circ}$ type. Instead, we are only interested in the number of recovered minutiae, which shall be denoted by *R*. The average[61] number of guesses required to figure out a replacement *N*-

[57] The number of *N*-node Patterns available in a fingerprint is fundamentally controlled by the total number of minutiae available in that fingerprint, *T*. However, that number will vary with the selected Pattern size, *N*, and the adopted matching thresholds.

[58] As stated in Section 4.5.2 and Section 12.1, an analysis on the *number* of possible cancellations is currently lacking in the literature corresponding to non-invertible fingerprint template protection schemes. However, cancellability *is* evaluated in terms of the probability that two protected templates generated from the same fingerprint are sufficiently *different* from each other to enable the replacement of one by the other.

[59] Recall, from Section 5.2.5, that arranging a subset of *N* minutiae in a different order constitutes a new Pattern.

[60] Recall that our analysis in Section 10.2 showed that it is possible to recover the attributes of the *N* minutiae used to construct an *N*-node Pattern.

[61] The average number of guesses is assumed to be equal to half the maximum number of guesses.

node Pattern that has been generated using *all* N minutiae from the subset of R recovered minutiae is then calculated using Equation (12.5):

$$Average\ Number\ of\ Guesses = \frac{R!}{2(R-N)!} \tag{12.5}$$

Equation (12.5) was used to calculate the average number of guesses required to figure out a replacement N-node Pattern from one or more compromised Patterns originating from the same fingerprint, as R increases from N to the total number of minutiae available in the fingerprint, T. As in the analysis in Section 12.2.2, here we used $T = 42$, since this represents the median number of minutiae available in a reference minutiae template for our cooperative-user fingerprint database. For the sake of simplicity, it was assumed that all the replacement N-node Patterns are of the same size[62], so each N (where $N = \{3, 4, 5\}$) was dealt with in turn. Figures 12.1 to 12.3 depict the plots corresponding to the average number of guesses required for $N = 3$, $N = 4$, and $N = 5$, respectively.

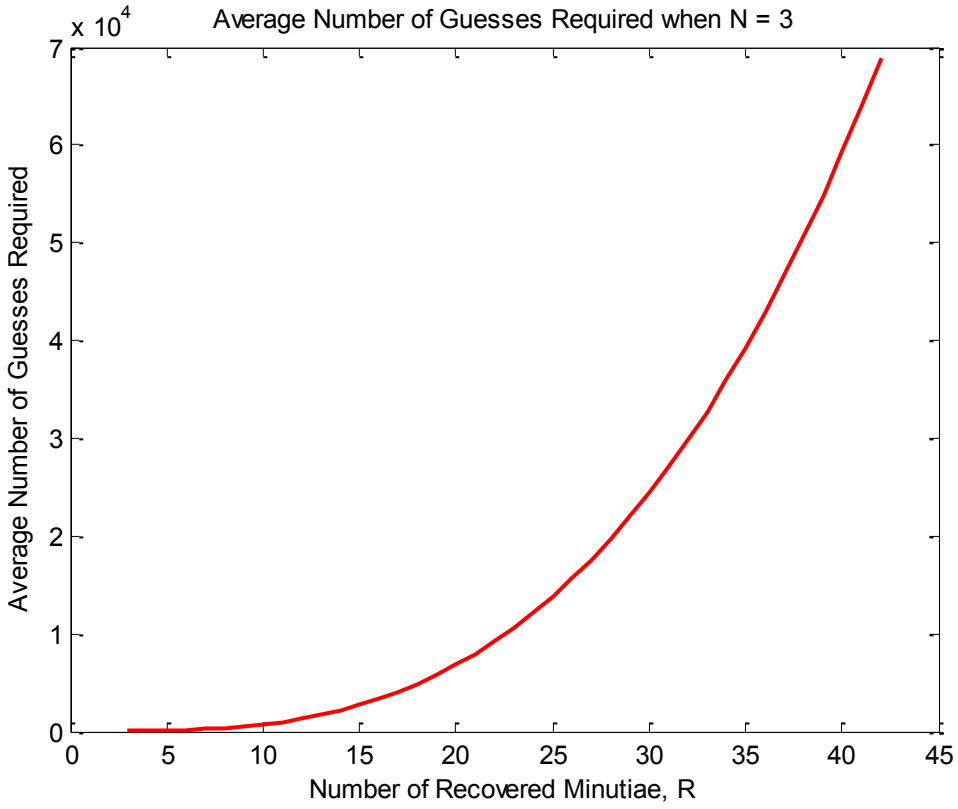


Figure 12.1: Average number of guesses required to guess the replacement 3-node Pattern as the number of recovered minutiae increases.

[62] If the Pattern sizes are mixed, then the average number of guesses required would lie somewhere between the average number of guesses required for 3-node Patterns and the average number of guesses required for 5-node Patterns.

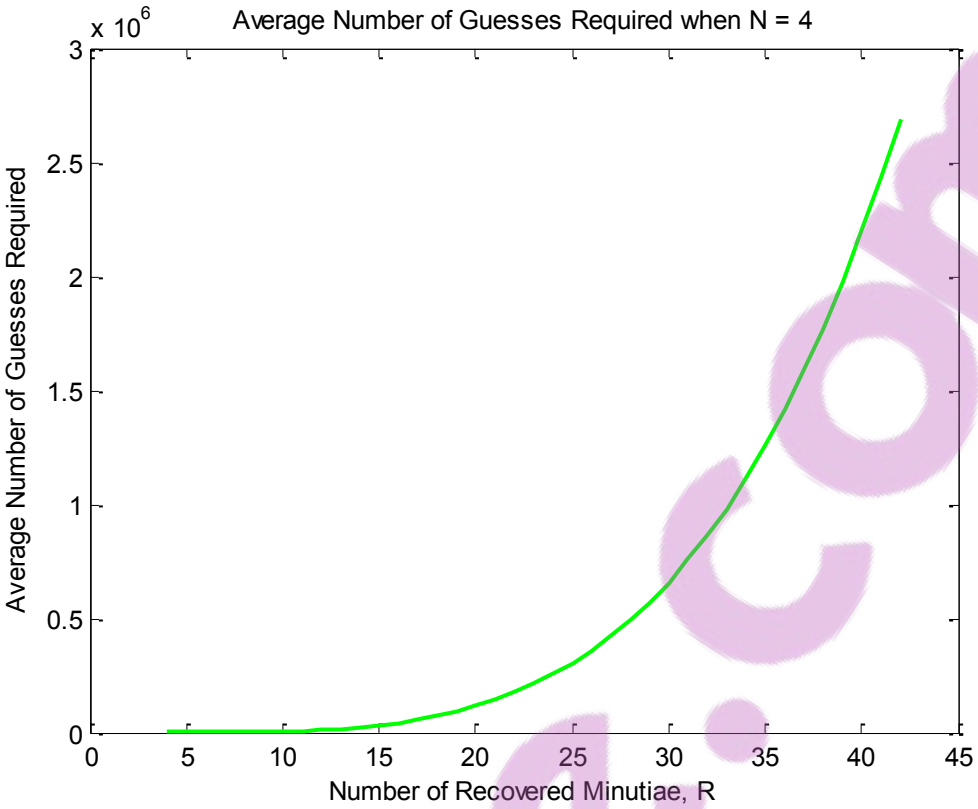


Figure 12.2: Average number of guesses required to guess the replacement 4-node Pattern as the number of recovered minutiae increases.

Figure 12.3: Average number of guesses required to guess the replacement 5-node Pattern as the number of recovered minutiae increases.

Two trends are evident from observing Figures 12.1 to 12.3. Firstly, for each N, the average number of guesses required to figure out the replacement N-node Pattern increases exponentially with the number of recovered minutiae, R. In fact, from Equation (12.5), we may conclude that an increase in R by 1 causes the average number of required guesses to increase by $\frac{(R+1)!}{2(R+1-N)!} - \frac{R!}{2(R-N)!}$. This increase is expected, because the more minutiae that an attacker has access to, the more difficult it is to figure out which N of those R minutiae were used and in what order they were arranged to construct the replacement N-node pattern.

The second observation from Figures 12.1 to 12.3 is that the average number of guesses required to figure out the replacement N-node Pattern from the recovered minutiae increases with an increase in the Pattern size, N. From Equation (12.5), we may deduce that an increase in N by 1 causes the average number of required guesses to increase by $\frac{R!}{2(R-N+1)!} - \frac{R!}{2(R-N)!}$. This increase is logical, because the larger the N, the larger the number of ways to arrange the N minutiae into an N-node Pattern.

Let us now use Figures 12.1 to 12.3 to estimate the point at which it becomes *safe* for a person to generate their replacement N-node Pattern using *all* N minutiae from the set of R recovered minutiae (i.e., no *new* minutiae). The "safe point" shall be determined in terms of the number of recovered minutiae, R. Note that the safe point will be different for each person and each application in practice, so the aim of this analysis is simply to provide an example of how the safe point can be calculated.

Assume that a person has enrolled into a particular application using an N-node Pattern. Let D denote the number of days during which the user is expected to be enrolled in this particular application. In this example, we shall set D = 10 years = 3,650 days. Further, let A denote the maximum number of times *per day* that a user is allowed to input their Pattern during authentication, before they are locked out of their account for 24 hours on suspicion of being an impostor. In this example, we shall set $A = 5$. Now, say that an impostor steals the user's reference N-node Pattern from this application's database, forcing the user to replace the compromised Pattern with a new N-node Pattern from the same fingerprint. If this happens more than once, we may assume that the attacker has recovered R of the user's T minutiae, where $R \geq N$. The attacker then tries to re-arrange the subset of R recovered minutiae to guess the user's replacement N-node Pattern. Assuming that the attacker makes 5 guesses per day, the total number of guesses that they will have made after 10 years would be equal to $D \times A = 3{,}650 \times 5 = 18{,}250$. The "safe point" then refers to the total number of minutiae that must be recovered, R, such that the average number of guesses required to figure out the replacement Pattern from the R recovered minutiae exceeds 18,250. To figure out R

for this example, we refer to Figures 12.4 to 12.6, which are zoomed-in versions of Figures 12.1 to 12.3, respectively, with the *R* corresponding to 18,250 marked on each plot.

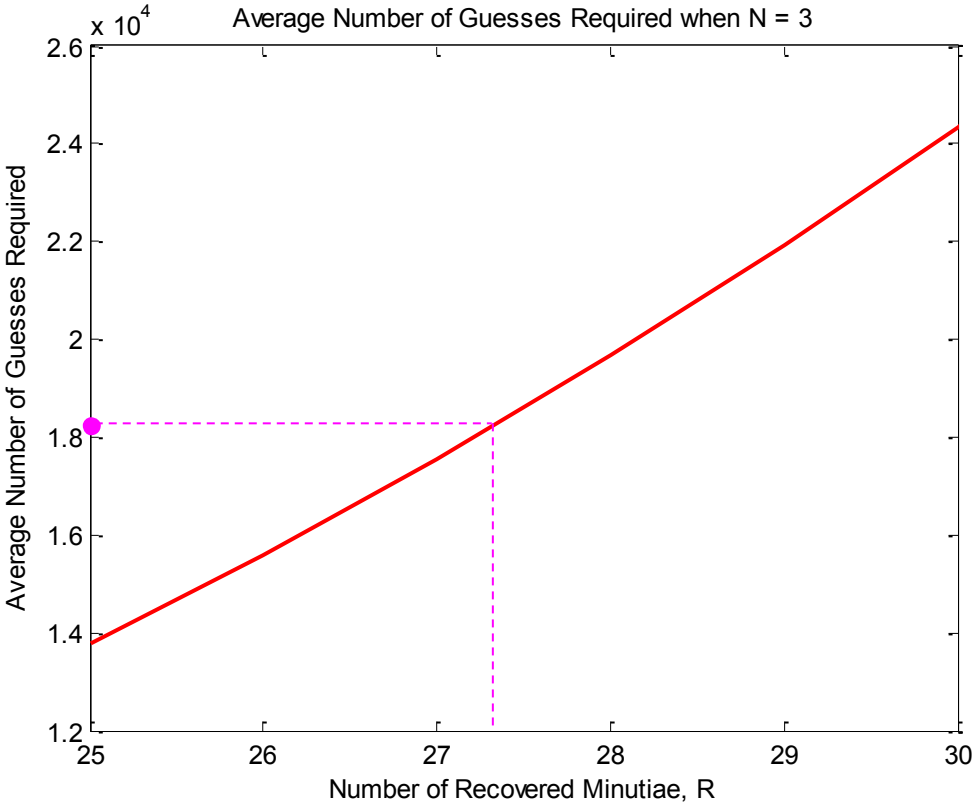


Figure 12.4: Plot showing the minimum number of minutiae that must be recovered in order for the number of guesses required to guess the replacement 3-node Pattern to exceed 18,250.

From Figure 12.4, we can see that, in order for the average number of guesses required to figure out the replacement 3-node Pattern to exceed 18,250, at least 27 minutiae must have already been recovered. We may thus conclude that, in this example, it would be safe to generate a replacement 3-node Pattern using 3 minutiae from the recovered set of minutiae after at least 27 minutiae have been recovered. If fewer than 27 minutiae have been recovered, we recommend that the replacement 3-node Pattern be generated using *at least 1 new* minutia (i.e., at least 1 of the 3 minutiae must not have been previously recovered). Note that the number of compromised 3-node Patterns corresponding to 27 compromised minutiae would depend on how many *new* minutiae are used in each successive replacement 3-node Pattern, and whether we're dealing with *Fixed* $\mathbf{FC}_{360^\circ}$ Patterns or *Floating* $\mathbf{FC}_{360^\circ}$ Patterns.

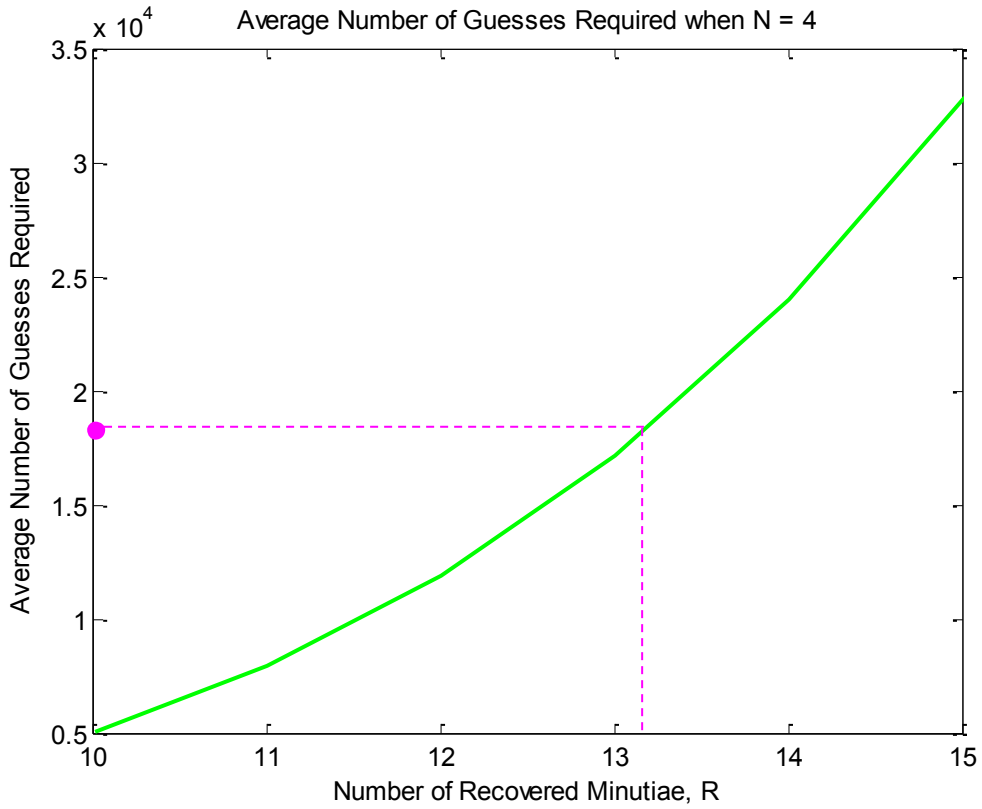


Figure 12.5: Plot showing the minimum number of minutiae that must be recovered in order for the number of guesses required to guess the replacement 4-node Pattern to exceed 18,250.

From Figure 12.5, we can see that, in order for the average number of guesses required to figure out the replacement 4-node Pattern to exceed 18,250, at least 13 minutiae must have already been recovered. We may thus conclude that, in this example, it would be safe to generate a replacement 4-node Pattern using 4 minutiae from the recovered set of minutiae after at least 13 minutiae have been recovered. If fewer than 13 minutiae have been recovered, we recommend that the replacement 4-node Pattern be generated using *at least 1 new* minutia (i.e., at least 1 of the 4 minutiae must not have been previously recovered). Note that the number of compromised 4-node Patterns corresponding to 13 compromised minutiae would depend on how many *new* minutiae are used in each successive replacement 4-node Pattern, and whether we're dealing with *Fixed* $\mathbf{FC}_{360°}$ Patterns or *Floating* $\mathbf{FC}_{360°}$ Patterns.

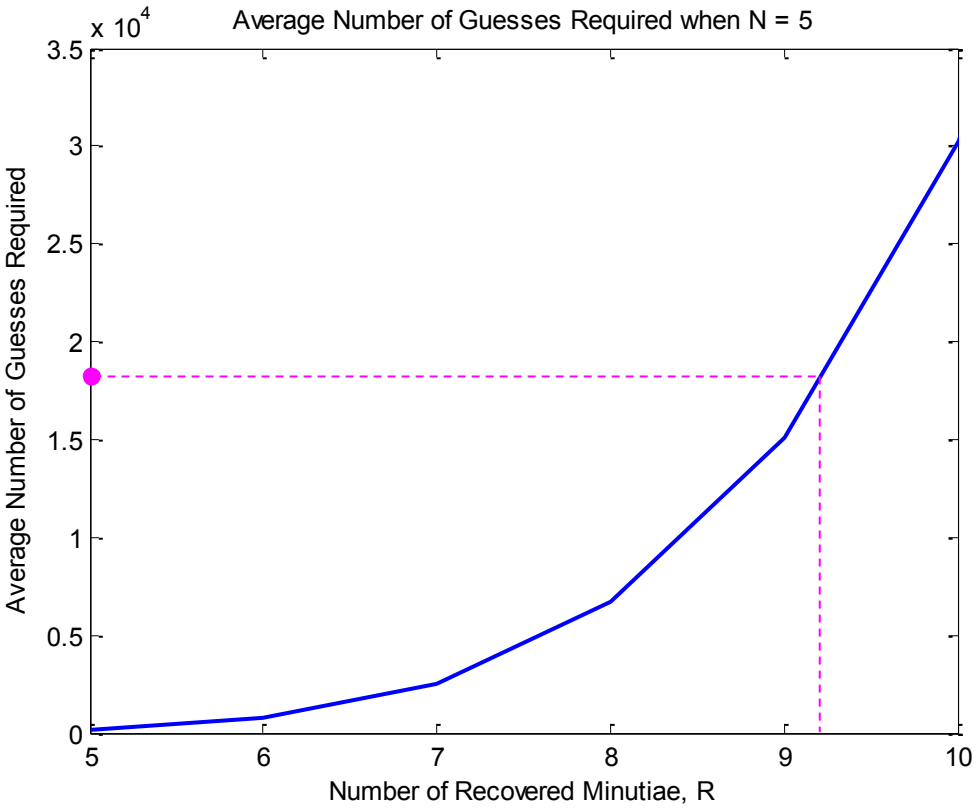


Figure 12.6: Plot showing the minimum number of minutiae that must be recovered in order for the number of guesses required to guess the replacement 5-node Pattern to exceed 18,250.

From Figure 12.6, we can see that, in order for the average number of guesses required to figure out the replacement 5-node Pattern to exceed 18,250, at least 9 minutiae must have already been recovered. We may thus conclude that, in this example, it would be safe to generate a replacement 5-node Pattern using 5 minutiae from the recovered set of minutiae after at least 9 minutiae have been recovered. If fewer than 9 minutiae have been recovered, we recommend that the replacement 5-node Pattern be generated using *at least 1 new* minutia (i.e., at least 1 of the 5 minutiae must not have been previously recovered). Note that the number of compromised 5-node Patterns corresponding to 9 compromised minutiae would depend on how many *new* minutiae are used in each successive replacement 5-node Pattern, and whether we're dealing with *Fixed* $\mathbf{FC}_{360°}$ Patterns or *Floating* $\mathbf{FC}_{360°}$ Patterns.

Overall, this analysis suggests that, to be on the safe side, a compromised *N*-node Pattern should be replaced by an *N*-node Pattern consisting of *at least 1 new minutia* every time. The exception to this rule is the scenario in which the "safe point" has been reached, in which case it would not matter if *all N* minutiae used to construct the replacement *N*-node Pattern have been recovered by the attacker[63]. This indicates that another important advantage of our

[63] As stated in the analysis of Figures 12.4 to 12.6, the number of actual *Patterns* that would need to be compromised in order to reach this safe point would depend on the number of new minutiae in each successive stolen Pattern, as well as on whether we are dealing with *Fixed* $\mathbf{FC}_{360°}$ or *Floating* $\mathbf{FC}_{360°}$ Patterns.

proposed fingerprint construct is that it ensures that an *N*-node Pattern is cancellable even if a person's entire minutiae template has been recovered. This may be likened to the PIN number system or the password system, where knowledge of the underlying alphabet is assumed, yet this, in itself, is not sufficient to enable an attacker to figure out a genuine user's PIN number or password.

12.3 DIVERSITY OF *FIXED* $\mathbf{FC_{360°}}$ AND *FLOATING* $\mathbf{FC_{360°}}$ *N*-NODE PATTERNS

Section 12.2 confirmed the *cancellability* of both the *Fixed* $\mathbf{FC_{360°}}$ and *Floating* $\mathbf{FC_{360°}}$ versions of our proposed fingerprint construct. The fact that our proposed fingerprint construct is *cancellable* further suggests the possibility of using different *N*-node Patterns from the same fingerprint to enrol into different applications without the risk of being tracked. This section explores this possibility by evaluating the *diversity* of *N*-node Patterns generated using the *Fixed* $\mathbf{FC_{360°}}$ and *Floating* $\mathbf{FC_{360°}}$ versions of our proposed fingerprint construct.

As mentioned in Section 4.5.3, in the literature the *diversity* criterion of an ideal fingerprint template protection scheme is generally considered to be synonymous with the *cancellability* criterion, such that the *same* analysis is applied towards proving both criteria simultaneously. The most common way of proving that a fingerprint template protection scheme satisfies the *diversity* requirement is by generating multiple protected templates from the same original template and then attempting to match the resulting set of protected templates: if the probability that two protected templates from the same fingerprint match is found to be very low, then the corresponding fingerprint template protection scheme is considered to satisfy both the *diversity* and *cancellability* requirements. Sometimes, a protected template is also matched to the original template in an attempt to prove that a failure to match indicates that the protected template cannot be linked to the original template. As explained in Section 4.5.2, however, this type of analysis is insufficient to prove whether or not two protected templates, or a protected template and the original template, are correlated on a deeper level than that indicated by the match score; indeed, since fingerprint template protection schemes in the literature generally use the entire minutiae template to generate the resulting protected template, there is bound to be some correlation between multiple protected templates originating from the same fingerprint, as well as between the protected and unprotected templates, regardless of whether or not this correlation is immediately obvious[64].

[64] If there is *no* link at all between a protected template and the original template, or two protected templates that come from the same original template, then this would indicate that all the fingerprint information has been

As mentioned in Section 4.5.1, [209] has presented evidence to show this observation to be true by proving that it is possible to correlate multiple protected templates originating from the same fingerprint to reconstruct the fingerprint's original template for four non-invertible fingerprint template protection schemes currently in the literature ([109, 134, 144, 221]).

Several times throughout this thesis, it has been emphasized that an important strength of our proposed fingerprint construct is that it does not use a fingerprint's entire minutiae template in the generation of the protected fingerprint template. We may, therefore, reason that multiple *N*-node Patterns originating from the same fingerprint are less likely to be correlated than multiple protected templates generated using the fingerprint's entire minutiae template. Nevertheless, it is possible that multiple *N*-node Patterns generated from the same fingerprint may exhibit some correlation, and it is the aim of this section to discuss the *diversity* of our proposed fingerprint construct in light of this possible correlation.

We consider the worst-case scenario, in which it is assumed that an attacker is able to break into multiple applications in which the same person is enrolled, and that they are able to steal each application's database, where the database consists of the reference *N*-node Patterns of all users enrolled in that particular application. We further assume that the attacker is able to recover each *N*-node Pattern's *N* constituent minutiae, as described in Section 10.2. Consequently, since the smallest Pattern considered in this thesis consists of 3 minutiae, and since it has been shown that 3-node Patterns are sufficiently *unique* for recognition purposes, we shall assume that it is possible to track a person across multiple applications if the reference Patterns enrolled in those applications share at least 3 minutiae. Keeping this point in mind, we may infer that multiple *N*-node Patterns from the same fingerprint will be *fully unlinkable* across multiple applications' databases as long as the Patterns all differ from each other by at least $N - 2$ minutiae. Equation (12.6) can then be used to estimate the maximum number of *diverse N*-node Patterns possible from a single fingerprint, where Patterns are considered to be *diverse* if they are *fully unlinkable*:

$$Maximum\ Number\ of\ Diverse\ Patterns = \left\lfloor \frac{T - N}{N - 2} \right\rfloor + 1 \tag{12.6}$$

Note that, in Equation (12.6), *T* denotes the total number of minutiae available in the entire minutiae template. Applying Equation (12.6) to our cooperative-user fingerprint database, for which the median *T* value was found to be 42 (see Section 12.2), the median maximum

---

destroyed in the generation of the protected template and thus the protected template consists of external information only.

number of diverse N-node Patterns for each value of $N = \{3, 4, 5\}$ was evaluated. The results are summarised in Table 12.6.

Table 12.6: The median of the maximum number of *diverse N*-node Patterns possible from a reference fingerprint for our cooperative-user fingerprint database.

	$N = 3$	$N = 4$	$N = 5$
Maximum Number of Diverse Patterns	40	20	13

From Table 12.6, we may conclude that, for our cooperative user fingerprint database, the central tendency for the N-node Pattern diversity suggests that:

- A user can enrol into a maximum of 40 applications using a different 3-node Pattern in each application, without the possibility of being tracked across the different applications. This is a 3,900% increase in diversity compared to the single identity provided by the traditional mechanism of storing the fingerprint as a raw minutiae template.
- A user can enrol into a maximum of 20 applications using a different 4-node Pattern in each application, without the possibility of being tracked across the different applications. This is a 1,900% increase in diversity compared to the single identity provided by the traditional mechanism of storing the fingerprint as a raw minutiae template.
- A user can enrol into a maximum of 13 applications using a different 5-node Pattern in each application, without the possibility of being tracked across the different applications. This is a 1,200% increase in diversity compared to the single identity provided by the traditional mechanism of storing the fingerprint as a raw minutiae template.

Note that, if a user enrols into multiple applications using Patterns of mixed sizes, then we may expect the maximum number of fully unlinkable Patterns to be no greater than the maximum number of diverse Patterns possible for the largest Pattern size used. For example, if the largest Pattern used consists of 5 minutiae, then the total number of mixed-size diverse Patterns cannot exceed 13 (based on the results in Table 12.6).

Overall, our results in Table 12.6 provide excellent support for the fact that our proposed fingerprint construct satisfies the *diversity* requirement of an ideal fingerprint template protection scheme. Note, however, that this analysis was based on the *worst-case* scenario only, in which it was assumed that an attacker is able to get hold of the databases corresponding to a large number of applications in which a particular person is enrolled. In practice, however, the tracking of a person across multiple applications is unlikely to be so extreme, especially in everyday civilian applications for which our proposed fingerprint

construct is intended[65]. It is more likely that tracking would be attempted by stealing one of a person's reference *N*-node Patterns from a particular application and then attempting to use that same Pattern to try to authenticate as that user in other applications in which that user may be enrolled. In this case, the definition of *diversity* essentially boils down to the definition of *cancellability*[66], where the requirement that two *N*-node Patterns from the same fingerprint are *different* (i.e., non-matching) is sufficient to ensure that those two Patterns cannot be linked to the same fingerprint. The ability of our proposed fingerprint construct to satisfy this more common, less strict definition of *diversity* has thus already been covered in Section 12.2.

12.4 SUMMARY

This chapter evaluated the *cancellability* and *diversity* of our proposed fingerprint construct.

The *cancellability* of both the *Fixed* $\mathbf{FC}_{360^\circ}$ and *Floating* $\mathbf{FC}_{360^\circ}$ versions of our proposed fingerprint construct was analysed via two approaches. First, we evaluated the probability that two *N*-node Patterns generated from the same fingerprint would match in practice. This probability was found to be very low for both the *Fixed* $\mathbf{FC}_{360^\circ}$ and *Floating* $\mathbf{FC}_{360^\circ}$ versions of our proposed fingerprint construct. This analysis confirmed that our new fingerprint construct is *cancellable*, in the sense that it would be possible to replace a compromised *N*-node Pattern with a new *N*-node Pattern from the same fingerprint, such that the two Patterns do not match. We then estimated the number of *N*-node Patterns available in a single fingerprint to provide insight on the number of times that a compromised *N*-node Pattern can be replaced with a new *N*-node Pattern from the same fingerprint. This number was found to be more than sufficient for cancellability purposes in practice. Finally, we considered whether every potential replacement Pattern from the same fingerprint would be a *safe* replacement, in terms of the difficulty for an attacker to determine the replacement Pattern from one or more compromised Patterns originating from the same fingerprint. It was established that there exists a "safe point" beyond which every potential replacement Pattern essentially becomes a safe replacement Pattern. Until this safe point is reached, however, we recommended that, to be on the safe side, each replacement *N*-node Pattern should consist of

[65] This is because, firstly, it should be very difficult for an attacker to be successful in breaking into such a large number of different applications, and, secondly, the amount of effort required in this endeavour is unlikely to be invested unless the attacker is desperate to track one particular person across every single application in which that person is enrolled.

[66] This is the definition that is commonly adopted in the associated literature, as mentioned in Section 4.5.2.

at least 1 new minutia (i.e., a minutia which was not used in *any* previous compromised Patterns).

We then analysed the *diversity* of our proposed fingerprint construct. The strict definition of *diversity* required that multiple *N*-node Patterns generated from the same fingerprint are *fully unlinkable*, which was assumed to be satisfied provided that the Patterns share fewer than 3 minutiae. It was shown that, in the worst-case scenario, where an attacker gains access to the databases of multiple applications in which a particular person is enrolled, our proposed fingerprint construct is able to satisfy the *diversity* requirement under the aforementioned strict definition. A more common, less strict definition of *diversity* considers two *N*-node Patterns to be *diverse* simply if they do not match each other. In this case, our *cancellability* analysis is sufficient to prove that both the *Fixed* $\mathbf{FC}_{360^\circ}$ and *Floating* $\mathbf{FC}_{360^\circ}$ versions of our proposed fingerprint construct satisfy the *diversity* requirement under this alternative definition, which is the definition commonly adopted for *diversity* analysis in the literature.

Overall, the analysis in this chapter has been fruitful in terms of confirming that our proposed fingerprint construct satisfies the *cancellability* and *diversity* characteristics of an ideal fingerprint template protection scheme. Our analysis in previous chapters (i.e., Chapter 5 onwards) has proven that our new fingerprint construct also satisfies the *non-invertibility* and *performance* requirements. We may thus conclude that this fingerprint construct has potential to be used as an effective fingerprint template protection scheme in practice.

Chapter 13

Conclusions and Future Work

The fundamental objective of this thesis was to develop a new non-invertible fingerprint template protection scheme for cooperative-user civilian fingerprint recognition applications, with the particular intention of presenting a fresh point of view on tackling this challenging task. This objective has been achieved via the proposal and subsequent development of a novel fingerprint construct, which, by its very nature, characterises a non-invertible fingerprint template protection scheme. This chapter summarises the milestones achieved in this journey and suggests avenues for future work directed at further mining the potential of our promising new fingerprint template protection scheme.

13.1 CONCLUSIONS

The current popularity of fingerprint recognition technologies, as well as the foreseen pervasiveness of this means of authentication, urgently calls for an effective mechanism for securely storing the growing fingerprint databases. The magnitude of the consequences resulting from a fingerprint database breach is largely influenced by the biological nature of fingerprints, which means that a compromised fingerprint is compromised for life, thereby potentially indicating a lifelong threat to a person's privacy and security. The unsuitability of traditional data protection mechanisms for the purpose of securing fingerprint templates has forced the creation of an exciting new research field, which is dedicated to finding an effective solution specifically suited to the nature of fingerprint data. Although a variety of creative techniques have been proposed in the associated literature, there is, as of yet, no agreed-upon solution. This means that the design of an ideal fingerprint template protection scheme remains an open problem. The difficulty of this problem, combined with the urgency of developing an effective solution, has motivated the research presented in this thesis.

An in-depth review of the associated literature revealed that existing non-invertible fingerprint template protection schemes tend to use the entire minutiae template in generating the protected fingerprint template. The issue with this approach is that the security of the

fingerprint template relies only on the effectiveness of the employed non-invertible transform. Since it is difficult to design a good non-invertible transform, often the non-invertibility of the resulting protected template is limited under the assumption that an attacker has access to the transform and any associated external parameters. To remove the danger of recovering a fingerprint's original template as a result of a poorly-designed non-invertible transform applied to an entire minutiae template, this thesis presents an alternative point of view on securing fingerprint templates. In particular, we investigated the use of only a *small portion* of the entire minutiae template to generate the protected template. The underlying premise was to achieve non-invertibility via the simple fact that most of the information required to reconstruct the original minutiae template is literally missing from the protected template stored in the database.

The main contributions of this thesis are the proposal and subsequent analysis of a new fingerprint construct, which is a non-invertible fingerprint template protection scheme by its very nature. The crux of our scheme entails the representation of a fingerprint by a single N-node Pattern constructed using a small subset of N minutiae from the underlying minutiae template. A Pattern consisting of N minutiae is referred to as an N-node Pattern. The smallest Pattern size considered in this thesis is $N = 3$ and the largest Pattern size is $N = 5$. An N-node Pattern consists of a set of *local features*, which describe the Pattern's *shape*, and a set of *global features*, which denote the Pattern's location and orientation in the fingerprint relative to the fingerprint's core. The proposed fingerprint construct has been designed with the aim of being deployed as a more secure alternative to full minutiae templates in cooperative-user civilian fingerprint recognition applications.

The greater part of this thesis was dedicated to a rigorous analysis of our proposed fingerprint construct, in order to gauge its suitability to serve as an effective fingerprint template protection scheme in practice. The evaluation was focused on the four characteristics that define an ideal fingerprint template protection scheme: *performance*, *non-invertibility*, *cancellability*, and *diversity*.

Performance:

In order for a fingerprint template protection scheme to satisfy the *performance* requirement, the use of the resulting protected templates should not reduce the recognition accuracy that is attainable by the use of the unprotected templates. Since it is mathematically impossible to generate a non-invertible fingerprint template without some loss of information, however, a reduction in the recognition accuracy is expected with any non-invertible fingerprint template protection scheme. A comparison of the performance of our fingerprint construct to that of a

fingerprint recognition system that uses full, unprotected minutiae templates thus confirmed this expected loss in the attainable recognition accuracy. Nevertheless, extensive analysis on the performance of our new fingerprint construct indicates that, despite its sparsity, and *N*-node Pattern may be expected to have acceptable recognition accuracy in the cooperative-user scenario for which it is intended. In particular, we found that an *N*-node Pattern is capable of effectively discriminating between a genuine user and an impostor. The best recognition accuracy was shown to be attainable using the *Fixed* $\mathbf{FC}_{360^\circ}$ version of our proposed fingerprint construct in the *Two-Factor Authentication* scenario.

We also demonstrated that the ability of an *N*-node Pattern to discriminate between a genuine user and an impostor improves as the Pattern size increases and the matching thresholds decrease. We thus recommended using a 3-node Pattern combined with larger matching thresholds for low-security applications, and a 5-node Pattern along with smaller matching thresholds for high-security applications. A comparison of the performance of *Fixed* $\mathbf{FC}_{360^\circ}$ to the reported performance of non-invertible fingerprint template protection schemes that use the full minutiae template showed favourable results, suggesting that our proposed fingerprint construct is a significant competitor despite using only a small portion of the entire minutiae template.

Non-invertibility:

A fingerprint template protection scheme is considered to satisfy the *non-invertibility* requirement if it is impossible (or computationally infeasible) to reconstruct the original minutiae template from the protected template. Our proposed fingerprint construct was shown to satisfy the *non-invertibility* requirement from three perspectives.

Firstly, we showed that, since an $\mathbf{FC}_{360^\circ}$ *N*-node Pattern leaks *N* out of *T* minutiae from a *T*-minutiae template, it may be likened a *T*-to-*N* mapping. Since $N << T$, this analysis served as proof that our proposed fingerprint construct is indeed *non-invertible*.

Secondly, we evaluated the degree of *non-invertibility* of our fingerprint construct in terms of the proportion of the underlying minutiae template that remains unrevealed despite the *N* minutiae leaked by an *N*-node Pattern. It was found that, on average, 93% of a fingerprint's minutiae template may be expected to remain unrevealed in the event that a 3-node Pattern originating from the corresponding fingerprint is stolen from the database, 90% would remain unrevealed with the compromise of a 4-node Pattern, and 88% would remain unrevealed with the compromise of a 5-node Pattern. Since, to the best of our knowledge, no other existing non-invertible fingerprint template protection scheme is able to secure such a

large portion of the original minutiae template, we may surmise that our proposed fingerprint construct is superior in this respect.

Thirdly, we estimated the complexity of recovering the unrevealed portion of the minutiae template using a brute-force approach. This analysis showed that the sparsity of an $\mathbf{FC}_{360^\circ}$ *N*-node Pattern ensures that it is practically impossible to reconstruct the corresponding fingerprint's entire minutiae template, thereby justifying the degree of *non-invertibility* provided by our proposed fingerprint construct.

The three veins of analysis summarised above considered *non-invertibility* from the point of view where the attacker gains access to only *one* *N*-node Pattern from a particular user. We also analysed the susceptibility of an $\mathbf{FC}_{360^\circ}$ *N*-node Pattern to a Record Multiplicity Attack, in the scenario in which an attacker is able to piece together the minutiae recovered from multiple *N*-node Patterns originating from the same fingerprint to reconstruct the original minutiae template. Experimental analysis indicated that the number of $\mathbf{FC}_{360^\circ}$ *N*-node Patterns that an attacker would need to acquire to be successful in this endeavour may be expected to be large enough to ensure that our proposed fingerprint construct is able to resist a Record Multiplicity Attack in practice. Furthermore, the fact that our fingerprint construct uses only a small portion of the entire minutiae template gives us reason to believe that it is more resistant to a Record Multiplicity Attack than fingerprint template protection schemes that use the entire minutiae template in generating the protected fingerprint template (i.e., *correlation* in the latter case is more likely). A lack of numerical analysis on the susceptibility of other non-invertible fingerprint template protection schemes to a Record Multiplicity Attack in the literature, however, means that we cannot present a quantitative comparison to the resistance of $\mathbf{FC}_{360^\circ}$.

The last stage of the non-invertibility analysis in this thesis was the proposal and analysis of a modification to our fingerprint construct, which further increases its resistance to a Record Multiplicity Attack. The modification involved removing the Pattern's global attributes and expressing the Pattern in terms of its local attributes only. The modified construct was termed *Floating* $\mathbf{FC}_{360^\circ}$ and the prior version was re-named *Fixed* $\mathbf{FC}_{360^\circ}$. An evaluation of the recognition accuracy attainable by *Floating* $\mathbf{FC}_{360^\circ}$ demonstrated that *Floating* $\mathbf{FC}_{360^\circ}$ *N*-node Patterns are generally less discriminative than *Fixed* $\mathbf{FC}_{360^\circ}$ *N*-node Patterns; however, appropriate selection of the matching thresholds and the most suitable Pattern size for the requirements of a particular application would ensure that the recognition accuracy of both the *Fixed* $\mathbf{FC}_{360^\circ}$ and *Floating* $\mathbf{FC}_{360^\circ}$ versions of our proposed fingerprint construct are acceptable in practice.

Cancellability:

A fingerprint template protection scheme is considered to possess the *cancellability* trait if it is possible to cancel (revoke) a compromised protected template and replace it with a new protected template from the same unprotected template. The *cancellability* of both the *Fixed* $\mathbf{FC}_{360°}$ and *Floating* $\mathbf{FC}_{360°}$ versions of our proposed fingerprint construct was analysed via two approaches.

Firstly, we evaluated the probability that two *N*-node Patterns generated from the same fingerprint would match in practice. This probability was found to be effectively zero for both the *Fixed* $\mathbf{FC}_{360°}$ and *Floating* $\mathbf{FC}_{360°}$ versions of our proposed fingerprint construct. This analysis confirmed that our new fingerprint construct is *cancellable*.

Secondly, we estimated the number of *N*-node Patterns available in a single fingerprint to provide insight into the number of times that a compromised *N*-node Pattern would be able to be replaced with a new *N*-node Pattern from the same fingerprint. It was found that a median of over 68,000 3-node Pattern replacements, over 2,600,000 4-node Pattern replacements, and over 104,800,000 5-node Pattern replacements would be possible from a single fingerprint, provided that the matching thresholds are not excessively large. These numbers should be more than sufficient for cancellability purposes in practice.

We then considered whether every potential replacement Pattern from the same fingerprint would be a *safe* replacement, in terms of the difficulty for an attacker to determine the replacement Pattern from one or more compromised Patterns originating from the same fingerprint. It was established that there exists a "safe point" beyond which every potential replacement Pattern essentially becomes a safe replacement Pattern. Until this safe point is reached, however, we recommended that, to be on the safe side, each replacement *N*-node Pattern should consist of *at least 1 new minutia* (i.e., a minutia which was not used in *any* previous compromised Patterns). This analysis revealed that an additional benefit of our proposed fingerprint construct is that it enables a person's fingerprint to continue to be used for recognition purposes even if an attacker were to get hold of that person's entire fingerprint and therefore their full minutiae template. This is because, under *Two-Factor Authentication*, knowledge of a person's minutiae template does not reveal their chosen reference Pattern.

Diversity:

A fingerprint template protection scheme is considered to satisfy the *diversity* property if it is possible for a user to enrol into multiple applications using a different protected template generated from the same unprotected template in each application, such that it is not possible to cross-match the user across these applications' databases. To evaluate the *diversity* of our

proposed fingerprint construct, we defined *diversity* in terms of the requirement that multiple *N*-node Patterns generated from the same fingerprint are *fully unlinkable*. This was assumed to be satisfied provided that the Patterns share fewer than 3 minutiae. Our analysis showed that, in the worst-case scenario, where an attacker gains access to the databases of multiple applications in which a particular person is enrolled, our proposed fingerprint construct is able to satisfy the *diversity* requirement under the aforementioned strict definition.

A more common, less strict definition of *diversity* would consider two *N*-node Patterns to be *diverse* simply if they do not match each other. In this case, the definition of *diversity* essentially boils down to the definition of *cancellability*, in which case our *cancellability* analysis becomes sufficient to prove that both the *Fixed* $\mathbf{FC_{360°}}$ and *Floating* $\mathbf{FC_{360°}}$ versions of our proposed fingerprint construct satisfy the *diversity* requirement.

General conclusions on the four properties:

The extensive analysis of our proposed fingerprint construct presented in this thesis indicates that it is able to comply satisfactorily with the four properties of an ideal fingerprint template protection scheme: *performance*, *non-invertibility*, *cancellability*, and *diversity*. Although it was shown that the performance of our fingerprint construct may be expected to degrade the recognition accuracy attainable using full minutiae templates, this is expected with any non-invertible fingerprint template protection scheme and thus should not be used to argue that our scheme does not satisfy the *performance* requirement. In fact, the important point to focus on with regards to the *performance* requirement is that our fingerprint construct was shown to be capable of achieving satisfactory recognition accuracy in the cooperative-user scenario for which it is intended. We may thus conclude that our proposed fingerprint construct indeed satisfies the requirements of *performance*, *non-invertibility*, *cancellability*, and *diversity*. Although this does not imply that our technique is perfect, it does indicate that it has potential to serve as an effective fingerprint template protection scheme in practice.

Other advantages of the proposed fingerprint construct:

Other advantages of the proposed fingerprint construct include:

- Its low storage requirements;
- the fact that an *N*-node Pattern is essentially a self-aligned structure, so pre-alignment against a query *N*-node Pattern during authentication is unnecessary;
- the *Floating* $\mathbf{FC_{360°}}$ version does not employ the core point, so core extraction errors are avoided;
- the scheme is easy to implement and simple to use;

- the user is required to memorise their reference *N*-node Pattern, so they are more knowledgeable about what area of the fingerprint must be captured during authentication and they are likely to be more aware of providing a good quality fingerprint during each authentication attempt;
- and, in our opinion, an authentication method based on our proposed fingerprint construct would be fun to use!

Limitations of the proposed fingerprint construct:

A limitation of our proposed fingerprint construct is its reliance on all *N* minutiae in a user's reference *N*-node Pattern being present during every authentication attempt, which may be affected by missing minutiae in practice. Our dedicated investigation into the consistency of cooperative users in placing their finger on the fingerprint scanner indicated that a median of over 96% of the reference minutiae may be expected to be present in a query sample of the same fingerprint presented during authentication. Furthermore, we empirically demonstrated that increasing the number of reference fingerprints during enrolment to establish the most reliable reference minutiae set is capable of improving minutiae persistence to a median of 100% when 3 or more reference fingerprints are used. While errors in automated minutiae extraction and matching were shown to have a detrimental effect on minutiae persistence among multiple samples of the same fingerprint, user consistency was found to be the most influential factor. Applying the insights gained from this investigation to our proposed fingerprint construct, we established that a genuine user would have the best chance of being successfully authenticated if 5 samples of their fingerprint were used during enrolment to establish the most reliable minutiae set to use for reference Pattern construction, and the user was allowed a maximum of 3 authentication attempts. In this scenario, the *true* FRR of our proposed fingerprint construct (i.e., the FRR due to one or more of a reference *N*-node Pattern's *N* minutiae physically missing in the query fingerprint) was found to be 0.16% for 3-node Patterns, 0.20% for 4-node Patterns, and 0.24% for 5-node Patterns. This analysis indicates that, provided that a robust minutiae extractor is used, our proposed fingerprint construct should seldom suffer from missing minutiae as a result of user inconsistency in practice. Furthermore, we recommended incorporating a live scan of the user's finger during authentication, such that the user is able to see what the scanner sees. In this case, the user could ensure that the part of their fingerprint in which their reference Pattern is contained is captured during their first authentication attempt.

A second limitation of our proposed fingerprint construct is the reliance of the *Fixed* $\mathbf{FC}_{360°}$ version on a reliable core extraction algorithm. While we acknowledge that this

limitation exists, the proposed fingerprint template protection scheme is intended for use in cooperative-user civilian authentication applications, in which case it is fair to assume that:

(i) The users will be cooperative, so the core area should be captured each time,

(ii) and a quality check will be performed on the acquired fingerprint images, so if a fingerprint image is of too poor a quality (in which case the core point may be detected incorrectly or not detected at all), the user would be asked to re-scan their finger.

To ensure that a valid core point can be found in fingerprints from *all* pattern classes (even the Arch types), we recommend defining the core point as the ridge point with the highest curvature. This is a common definition adopted in many automated fingerprint recognition algorithms. As an alternative, the *Floating* $\mathbf{FC}_{360^\circ}$ version of our proposed fingerprint construct, which does not use the core point at all, may be adopted instead.

Note that several publications have been produced from the work presented in this thesis (see Section 1.6).

13.2 FUTURE WORK

The extensive analysis of our new fingerprint construct in this thesis suggests that our method is promising in terms of its ability to function as an effective fingerprint template protection scheme in practice. In this section, we suggest several avenues for future work directed at further mining the potential of this promising new fingerprint template protection scheme.

Publication of the recent analysis:

Note that the conceptualisation and preliminary analysis of our new fingerprint construct has been published in [21]. However, much of the subsequent rigorous analysis, which considered the new construct on a deeper level, was conducted towards the end of this PhD, which means that we have not yet had the opportunity to publish all the results. As part of future work following the submission of this thesis, we aim to combine the results of the analysis presented in Chapters 7 through to 12 into at least 3 papers: the first on the performance of our new fingerprint construct, the second on our method's non-invertibility, and the third on the cancellability and diversity analysis of our method. Additionally, we aim to publish the material from Chapter 4 as a comprehensive discussion of the non-invertible fingerprint transform techniques existing in the literature and the means used to evaluate them.

Threshold optimisation:

The *Fixed* $\mathbf{FC_{360°}}$ version of our proposed fingerprint construct relies on four matching thresholds (τ_l, $\tau_{\alpha\beta}$, τ_{loc}, τ_{ω}) and the *Floating* $\mathbf{FC_{360°}}$ version relies on two (τ_l, $\tau_{\alpha\beta}$). Depending upon the requirements of a particular application (i.e., the desired balance between the FAR and FRR), a robust method of determining the optimal mix of those thresholds is a recommended next step in the development of our new fingerprint template protection scheme. This will depend on which Pattern attributes can be more/less reliably computed, which will in turn depend on the quality of the adopted feature extractor, the user-friendliness of the fingerprint scanner, etc. Since this will be influenced by a number of factors that are specific to the nature of a particular application and its adopted fingerprint processing algorithms, the development of a robust threshold selection algorithm may be most useful in the context of a specific application.

Investigation into the complexity of remembering a reference Pattern:

During the conception of our proposed fingerprint construct, it was envisioned that it would operate under *Two-Factor Authentication*, in which a user is required to memorise their reference *N*-node Pattern and present it along with their fingerprint during authentication. Subsequent analysis on the recognition accuracy attainable by our proposed fingerprint construct confirmed that *Two-Factor Authentication* is a more secure alternative to *Single-Factor Authentication*, in which a user is required to present only their fingerprint for authentication. Based on this analysis, we recommended that *Two-Factor Authentication* be adopted in practice. Since a user is required to memorise their reference Pattern for *Two-Factor Authentication*, it is important to consider the complexity of this task. In other words, a dedicated investigation into the ease with which a person is able to recall their reference Pattern during an authentication attempt should be conducted. Since a person's ability to memorise their reference Pattern will depend on factors such as the person's age, the complexity of their chosen Pattern, the frequency with which they use that Pattern for authentication purposes, etc., this investigation would be most useful in the context of a particular application.

Development of more robust minutiae and core extraction algorithms:

In order for our proposed fingerprint construct to function properly, it is important that the adopted minutiae and core extraction algorithms are robust. In other words, if a minutia is physically present in a fingerprint, it should always be identified by the minutiae extractor, and the detection of spurious minutiae should be minimised. Similarly, a suitable core point

should be identified in a consistent location in every sample of a person's fingerprint, even for Arch-type fingerprints. With the continual improvement in imaging technologies and the decreasing cost of fingerprint scanners, the development of more reliable fingerprint feature extraction algorithms should be a feasible goal.

Dealing with skin deformation:

Due to skin elasticity, a person's finger undergoes non-linear distortion when placed onto a fingerprint scanner. Depending on how the user places their finger on the scanner (e.g., the amount of pressure applied, the angle of the finger on the scanner surface, etc.), the acquired fingerprint image may be distorted in different ways during each authentication attempt. An investigation into the variation in this non-linear distortion across multiple samples of the same person's fingerprint in a cooperative-user scenario should be investigated in order to provide insight into how this phenomenon might affect our proposed fingerprint construct in practice. Alternatively, using the Pattern method with a touchless or 3D fingerprint scanner may avoid this distortion issue altogether.

Investigation into Pattern linkability:

In our analysis of the *diversity* of our proposed fingerprint construct, we assumed that two *N*-node Patterns are *unlinkable* if they share fewer than 3 minutiae. This is because an important design constraint for our fingerprint construct requires that the resulting *N*-node Pattern forms a closed shape, which means that the Pattern must consist of a minimum of 3 minutiae. Since a partial Pattern match does not count as a match, and because our analysis on Pattern *uniqueness* in this thesis only proves that Patterns consisting of 3 or more minutiae can be used for recognition purposes, we could not, at this stage, make claims on whether or not fewer than 3 common minutiae would be sufficient to conclude that two Patterns come from the same fingerprint. It would be interesting, however, to conduct an investigation into whether it is possible to link two *N*-node Patterns to the same fingerprint if the Patterns share fewer than 3 minutiae. We suspect that this would not be possible for *Floating* $\mathbf{FC}_{360°}$ *N*-node Patterns, but there may be a chance that it would be possible in some cases (e.g., if the matching thresholds are low enough) for *Fixed* $\mathbf{FC}_{360°}$ *N*-node patterns due to the additional use of the core point.

Investigation into collinear Pattern connection lines and mirror-image Patterns:

Our investigation in Section 8.5 showed that allowing a Pattern's angle attributes to lie in the range [0°, 360°), instead of restricting them to the [0°, 180°) range, is capable of reducing the

FAR by more than half. It was concluded that this may be due in part to the presence of collinear Pattern connection lines and Patterns that are mirror images of each other in terms of their locations and orientations relative to the core. An interesting area of future work would be to investigate the frequency of occurrence of collinear Pattern connection lines and mirror-image Patterns within the same fingerprint.

Using multiple reference Patterns to improve recognition accuracy:

A possible way to improve the recognition accuracy of our new fingerprint construct could be to store two or more reference Patterns per user instead of one. For example, storing two reference *N*-node Patterns per user would allow the user to have a 'back-up' Pattern in case one of the Patterns failed during authentication (for example, the constituent minutiae of that Pattern may not be captured or detected in the query fingerprint, the user may forget the Pattern, etc.). Alternatively, multiple *N*-node Patterns from the same person could be combined to make the resulting feature vector more discriminative. Although the multiple Patterns could originate either from the same finger or from different fingers, using different fingers would be a safer option. This is because multiple Patterns from the same fingerprint would reveal a larger portion of the underlying minutiae template than would a single Pattern, so if an attacker were to steal all those Patterns then they would be able to recover a larger portion of the full minutiae template. If the multiple Patterns came from different fingers, however, then an attacker with access to all the Patterns would not be any closer to recovering the full minutiae template of any of the underlying fingerprints than they would be if they stole a single Pattern.

Incorporation of the Pattern method into a multibiometric application:

The new fingerprint construct proposed in this thesis lends itself easily to essentially any point-set; therefore, there is potential in using our method for other biometric modalities besides fingerprints. For example, minutiae-like features are often extracted from finger and palm veins, and the face is usually characterised by a set of connected feature points. It may thus be possible to extend our fingerprint construct to a multibiometric application. For instance, the user could select one *N*-node Pattern from their fingerprint and another from their finger/palm vein feature set or from their face features. The two Patterns could then be fused together to create a more discriminative feature vector. An investigation into the effectiveness of our new fingerprint construct in a multibiometric application would form the subject of an exciting area of future work in this research direction.

Implementation of our proposed fingerprint construct in a practical application:

Finally, it would be exciting to see how our proposed fingerprint construct fares in a real-life practical application. For example, the pattern-based unlocking functionality currently existing in Android smartphones could be replaced by an unlocking mechanism that incorporates our proposed fingerprint construct. In particular, a user could scan their finger (via a scanner embedded into the smartphone), after which an image of their fingerprint, along with the automatically extracted minutiae points, would appear on the smartphone's screen. The user could then select a subset of their minutiae to construct a reference *N*-node Pattern, which would subsequently be stored in the phone's memory. To unlock the phone, the user would repeat the process, their reference *N*-node Pattern would be retrieved from memory, and the input Pattern would be compared to the reference Pattern: if the two Patterns match, authentication would be successful and the phone would unlock. Alternatively, the proposed fingerprint construct could be implemented for login purposes on a PC, or other everyday civilian authentication applications.

Appendix A

Fast Fingerprint Alignment Method based on Minutiae Orientation Histograms

The material presented in this appendix essentially replicates our associated publication, [24].

This appendix proposes a new alignment method, which uses histograms based on minutiae orientations to correct rotational differences between two samples of the same fingerprint. Experimentation shows that the proposed alignment method is promising in terms of its accuracy, particularly for coarse rotational alignment. Evaluation of the method's speed proves that it is considerably faster than the exhaustive search alignment technique, which is typically used for aligning minutiae templates. The results obtained motivate the possibility of a hybrid alignment approach, where the proposed technique is used for fast and coarse rotational alignment of two minutiae templates, followed by a finer alignment provided by a different alignment strategy (such as the exhaustive search technique) for increased accuracy.

A.1 INTRODUCTION

We live in an information-age, where easy access to proliferating amounts of data makes identity theft an increasingly difficult issue to contend with. Traditional security tokens, such as passwords and access cards, are becoming inadequate for the enhanced user authentication methods we urgently require. As our need for more robust security practices increases, we are beginning to witness a surge in automated biometric recognition technologies, whose global market is expected to increase almost three-fold in the next few years to an estimated $11 billion in 2015 [222]. The motivation behind the escalating use of biometrics for recognition purposes stems from their numerous benefits over traditional security tokens, including persistence over time, non-repudiability, uniqueness, universality, and the convenience they

offer to the public by providing highly secure and reliable authentication means without the need to memorize complex passwords or carry around access tokens [5].

While many different types of biometrics are constantly being investigated, fingerprints remain at the forefront of biometric recognition technologies, largely because of their maturity. Fingerprint recognition is most commonly based on matching small features called *minutiae* [5], the most prominent types of which are the bifurcation and the termination. A bifurcation is the point at which a fingerprint ridge splits into two ridges, and a termination is the point at which a ridge abruptly ends. Minutiae are thus typically represented as points in a two-dimensional plane, with the following attributes: $\{x, y, \theta\}$, where x and y correspond to the row and column indices, respectively, of a minutia in the fingerprint image, and θ represents the orientation of the minutia with respect to the horizontal axis. Minutiae-based fingerprint matching requires finding correspondences between the minutiae extracted from two fingerprint images. While on the surface this may appear to be a simple point pattern matching problem, in reality the minutiae extracted from two different samples of the same fingerprint are generally not identical. In particular, the query fingerprint is usually a rotated and/or translated version of the reference fingerprint, which means that the first step in minutiae-based matching is the alignment of the query minutiae with the reference minutiae. Minutiae are typically either pre-aligned (i.e., aligned prior to the matching stage) based on some agreed-upon reference point, or else alignment is performed via an exhaustive search strategy during matching. While many of the existing alignment techniques in these two categories have commendable accuracy, unfortunately this accuracy often comes at the price of a computationally expensive alignment strategy.

This appendix proposes a new alignment method, which uses histograms based on minutiae orientations to correct rotational differences between two samples of the same fingerprint. A preliminary investigation is carried out to test both the accuracy and the speed of the proposed alignment method. Accuracy results indicate that this technique is promising, provided that a sensible bin size is used in the creation of the histograms. In particular, experimentation shows that, 82% of the time, the proposed method is able to rotationally align two sets of minutiae with an accuracy of 5°, and an accuracy of 10° is attainable over 90% of the time. These results suggest that the proposed alignment strategy is particularly suitable for coarse rotational alignment. Experimental results on the speed of the proposed alignment method prove that it is considerably faster than the exhaustive search technique. In light of these results, the next step in this investigation should be the implementation of a hybrid alignment technique, which uses the proposed method for a fast, coarse rotational alignment

followed by a different alignment strategy (e.g., the exhaustive search technique) for fine-tuning the accuracy.

The remainder of this appendix is structured as follows: Section A.2 presents a brief literature review on commonly adopted minutiae-based fingerprint alignment techniques, Section A.3 outlines the proposed alignment method, Section A.4 presents experimental results on the accuracy and speed of the new alignment technique, and Section A.5 leaves the reader with a set of conclusions and suggestions for future work.

A.2 LITERATURE REVIEW

This section reviews the common methods adopted in the literature for aligning the minutiae extracted from two samples of the same fingerprint. Let R represent the set of minutiae extracted from a reference fingerprint (during enrolment) and let Q denote the set of minutiae extracted from a query fingerprint (during authentication). Consequently, Q' shall denote an aligned version of Q. Alignment is generally performed in one of two ways: Q and R are aligned prior to the matching stage, or else alignment is executed in conjunction with minutiae matching.

The former method of alignment is referred to as "pre-alignment", which is beneficial in speeding up the minutiae matching stage. Pre-alignment strategies may be classified into two types [5]: absolute pre-alignment and relative pre-alignment. Absolute pre-alignment involves aligning each minutiae template independently of the others, whereas relative pre-alignment requires pre-aligning Q with respect to R. While absolute pre-alignment is certainly more efficient (i.e., the alignment is done only once, whereas relative pre-alignment requires aligning T with every Q separately), relative pre-alignment tends to be more accurate [5].

Absolute pre-alignment is most commonly based on singular points in the fingerprint image. For example, all the minutiae in a fingerprint may be represented relative to the position of the corresponding core point (e.g., [223, 224]). In this way, translational offsets between Q and R are accounted for. A common way of dealing with rotational differences between the query and reference fingerprints is to compute the orientation of each fingerprint based on the orientation of the line between two singularities in the fingerprint (for example, a core and a delta point) [5]. The difference in the orientation of these lines in the two fingerprints indicates the rotational offset between Q and R. The problem with using singular points for pre-alignment is that they cannot always be reliably detected in fingerprint images, especially in partial and bad quality prints.

Relative pre-alignment is more popular than absolute pre-alignment, largely because of its superior accuracy. Relative pre-alignment is generally based on identifying reliable points in both the reference and query images, and then expressing all the minutiae in each image relative to the corresponding reliable point. Most often, the reliable point is chosen to be a minutia that is present in both Q and R. Let m_Q and m_R represent the reliable minutiae in Q and R, respectively. Then, all the remaining minutiae in Q are translated and rotated such that they lie in the reference frame specified by the location, (x, y), and orientation, θ, of m_Q; similarly for the minutiae in R using m_R. The reliable minutiae are often selected by constructing local minutiae structures in both Q and R and then comparing them. If SR and SQ denote the sets of structures in R and Q, respectively, and SR_i and SQ_j represent the closest matching structures, then the central minutia of SR_i is chosen to be m_R and the central minutia of SQ_j is chosen to be m_Q. Examples of literature in which this method of relative pre-alignment is adopted, include [225-228]. A similar method involves pre-alignment based on the rotational and translational differences between the associated ridges of the two most reliable minutiae, m_R and m_Q, which was proposed in [229]. The problem with relative pre-alignment based on reference points, such as minutiae, is that the process of establishing the most reliable reference points for alignment tends to be quite computationally intensive.

The latter method of alignment is generally based on an exhaustive matching process. Specifically, a number of different translations and rotations are applied to Q, and the match score between Q' and R is calculated at each iteration. The match score is usually a reflection of the number of minutiae correspondences between Q' and R. The combination of translation and rotation parameters, which result in the maximum match score, are deemed to be the optimal parameters for aligning Q with R. The most commonly cited publication that employs this exhaustive search strategy for alignment is [218], which uses a generalized Hough transform to accumulate evidence for each set of alignment parameters, where the evidence is based on the corresponding match score. The parameters that achieve the highest match score are used to align Q with R. Examples in literature that adopt the approach proposed in [218], include [230, 231]. Note that, while [218] does take into account scale differences between Q and R, often this parameter is ignored since it is assumed that the query fingerprint image has been captured using the same scanner that was used to acquire the reference fingerprint image. The exhaustive search method of alignment is very computationally expensive, since it requires calculating the match score for each rotation and translation of Q.

In this appendix, a new relative pre-alignment method is proposed to correct rotational differences between a reference and a query fingerprint, using histograms constructed from

the orientations of their corresponding minutiae. The simplicity of this method suggests that it faster than the exhaustive search alignment strategy as well as the relative pre-alignment method based on the establishment of reliable minutiae points. To the best of the authors' knowledge, this approach has not been previously proposed in the existing literature on fingerprint matching.

A.3 ROTATIONAL ALIGNMENT USING MINUTIAE ORIENTATION HISTOGRAMS

Let R and Q represent the sets of minutiae extracted from the reference and query fingerprints, respectively. This section presents a new alignment method for correcting rotational differences between Q and R, using the orientations of the minutiae in the two sets. The proposed alignment method is a three-step process: (1) creating minutiae orientation histograms for R and Q, (2) aligning the two histograms via circular correlation, and (3) aligning Q with R using the rotation parameter computed from the result of the correlation in (2). These steps are detailed in sections A.3.1, A.3.2, and A.3.3, respectively.

A.3.1 Creating Minutiae Orientation Histograms

In this stage, a histogram of minutiae orientations is generated for each fingerprint image. Let H^R and H^Q represent the histograms obtained for the reference, R, and query, Q, minutiae templates, respectively. The creation of each histogram is a three-step process, which may be summarized as follows:

Step 1: Divide the entire possible range of minutiae angles into a certain number of "bins". For example, if the minutiae orientations lie in the range [0°, 360°), then this range may be divided into 72 bins of 5° each.

Step 2: Place each minutia orientation into the appropriate bin. Continuing with the example from *Step 1*, a minutia angle of 13° would be placed into bin 3, which contains all the orientations within the range [10°, 15°), i.e., $10° \leq \theta \leq 14°$.

Step 3: After all the minutiae orientations have been binned, count the number of angles contained in each bin. The result is effectively a histogram of the minutiae orientations. For example, if H^R = [1, 0, 8, 5, …, 2], this means that R consists of 1 minutia whose orientation

is within the range [0°, 5°), zero minutiae with orientations within the range [5°, 10°), …, and 2 minutiae that have orientations within the range [355°, 360°).

A.3.2 Correlating the Histograms

The resulting H^R and H^Q are next aligned via a circular correlation process. This may be summarized in the following four steps:

Step 1: Place H^Q below H^R such that their first bins are aligned, and calculate the correlation between the two histograms.

Step 2: Circularly shift H^Q to the right a pre-determined number of times, calculating the correlation between H^Q and H^R at each shift.

Step 3: Repeat *Step 2* for the same number of shifts to the left.

Step 4: Record the shift amount (and direction) that resulted in the largest correlation value.

These four steps are depicted in Figure A.1, in which it is assumed (for ease of illustration) that minutiae angles lie in the range of 0° to 25°. Figure A.1 (a) shows *Step 1* of the process, in which H^Q is initially placed below H^R such that their first bins are aligned, and the correlation, C, between the two histograms is computed using Equation (A.1), where n represents the number of bins in each histogram, which is 5 in this example.

$$C = \sum_{i=1}^{n} H_i^R H_i^Q \tag{A.1}$$

Figures A.1 (b) and (c) then depict *Step 2*, where the correlation between H^R and H^Q is calculated at each of a number of circular shifts of H^Q to the right, while Figures A.1 (d) and (e) illustrate shifts to the left (*Step 3*). The maximum correlation result occurs at a shift of 2 to the left (i.e., in Figure A.1 (e)), so we record this shift amount and its sign (which is negative in this example) as per *Step 4*. Note that the number of shifts applied in each direction is determined by the maximum rotational difference that is deemed to be possible between Q and R. This point is elaborated on in Section A.3.3.

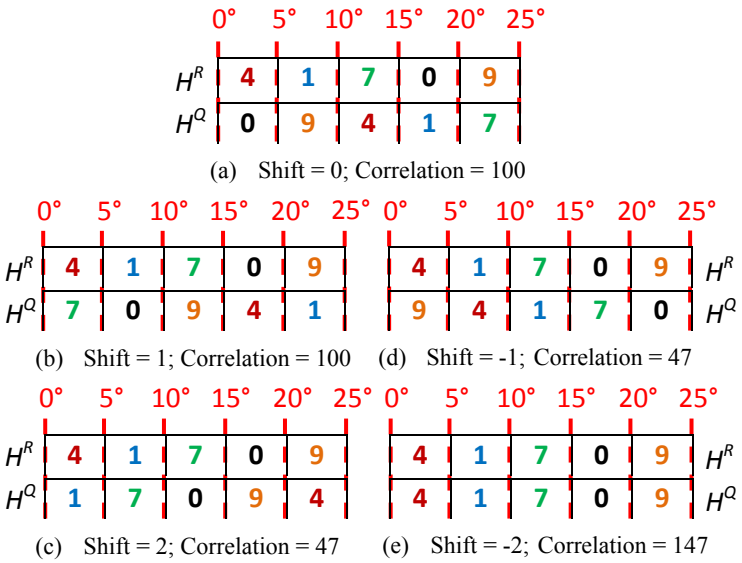


Figure A.1: Correlation results at different shifts of H^Q.

A.3.3 Aligning the Query and Reference Minutiae

In *Step 4* of the correlation process, the shift amount (and direction) resulting in the maximum correlation value is recorded. This shift amount and its corresponding direction are now used to calculate the rotation amount and direction that must be applied to the query minutiae, Q, in order to align them with the reference minutiae, R. To compute the rotation parameter, the shift amount (along with its direction sign) is multiplied by the bin size of the histograms. For the example in Figure A.1, the optimum shift was found to be -2. Since the bin size of the histograms in this example is 5°, the resulting rotation parameter is: -2 x 5° = -10°. This means that the query minutiae must be rotated by -10° in order to align them with the reference minutiae. Note that, if the minutiae orientations were initially calculated with the assumption that the angles increase in a clockwise direction, then the rotation parameter of -10° would suggest that the query fingerprint is initially rotated by 10° (clockwise) relative to the reference fingerprint, so the query minutiae, Q, must be rotated by -10° (anticlockwise) in order to align them with the minutiae in R. The centre of rotation is the origin of the minutiae coordinate system, assuming that translational offsets have been accounted for.

Assume that the maximum possible rotation of the query fingerprint with respect to the reference fingerprint is 60° in both the clockwise and anticlockwise directions. In this case, a maximum of 60°/bin-size shifts to the left and the same number of shifts to the right are required during the correlation of the query and reference histograms. If Q was originally rotated anticlockwise with respect to R, then H^Q would be a left-shifted version of H^R; therefore, shifts of H^Q to the right attempt to correct for this initial anticlockwise rotation. Similarly for shifts to the left, which check for an initial clockwise rotation of Q with respect

to R. While it is possible to simply shift H^Q in only one direction until we get back to the starting arrangement of the two histograms, this is unnecessary, since a query fingerprint is highly unlikely to be rotated by more than a certain amount (e.g., 60°) in either the clockwise or anticlockwise direction with respect to the reference fingerprint. For example, the query fingerprint will most probably never be placed upside down on the scanner surface. Hence, any shifts past the maximum required number to the left or right are generally redundant.

A.4 EXPERIMENTAL RESULTS

Two experiments were conducted to gauge the accuracy of the proposed alignment method at various bin sizes. The first experiment focused on the ideal scenario, where the minutiae in Q are the same as the minutiae in R, except rotated by some amount. The second experiment was conducted in a more realistic environment, using a public database of real fingerprint images. A third experiment evaluated the speed of the proposed alignment method and compared it to the speed of a typical exhaustive search alignment strategy. All experiments were executed in MATLAB 2011a. The methodology and results of each of these experiments are discussed in sections A.4.1, A.4.2, and A.4.3, respectively.

A.4.1 Experiment 1: Accuracy – Ideal Scenario

The aim of this experiment was to test the accuracy of the proposed alignment method for various bin sizes in the ideal scenario, where the minutiae in Q are simply a rotated version of the minutiae in R. While this is an unrealistic assumption in a typical fingerprint recognition system, and despite the seemingly trivial nature of this experiment, the ideal scenario is important for determining the optimal accuracy of the proposed alignment method. The ideal scenario was simulated by first choosing an arbitrary fingerprint sample from FVC2002 DB1_A and extracting its minutiae using the VeriFinger SDK tutorial [211]. Let R_θ denote the orientations associated with the set of extracted minutiae; then Q_θ was created by adding a certain angle to all the minutiae orientations in R_θ. Specifically, 121 versions of Q_θ were generated by adding angles in steps of 1° from -60° to 60° to all the orientations in R_θ. The proposed alignment method was then used to determine the difference between R_θ and each Q_θ, and thereby suggest by how much and in what direction the minutiae in each corresponding Q should be rotated to arrive at the set R (note that this rotation was not actually implemented here, since only the orientations of the minutiae in each Q were generated). For example, if $Q_{\theta 1}$ consists of the minutiae orientations in R_θ increased by -60°, then a rotation parameter of 60° is required to align Q_1 with R. Let θ_T denote the true rotation

parameters for all sets of Q_θ, and let θ_D represent the rotation parameters determined by the proposed alignment method. In the previous example, the first element in θ_T would be 60°. The accuracy of the proposed alignment technique was then calculated based on how close each element in θ_D was to its corresponding element in θ_T. The percentage of determined rotation parameters that were fully accurate, and accurate within 2° and 3° of the true rotation parameters, were computed for bin sizes of 1° to 10°. The results are summarized in Table A.1.

Table A.1: Accuracy of the proposed alignment method in a simulated ideal scenario, for various bin sizes.

Bin Size	Fully Correct Rotation Parameters	Rotation Parameters Correct Within 2°	Rotation Parameters Correct Within 3°
1°	100%	100%	100%
2°	50%	100%	100%
3°	34%	100%	100%
4°	26%	100%	100%
5°	21%	80%	100%
6°	17%	67%	83%
7°	14%	70%	85%
8°	12%	62%	80%
9°	11%	54%	75%
10°	11%	50%	70%

From Table A.1, it is evident that, in general, the larger the bin size, the lower the accuracy of the proposed alignment method. This is expected, since using a larger bin size would result in an inability to detect smaller rotation differences, i.e., the alignment would be coarser. These results suggest that a bin size of 1° would produce the most accurate alignments. This makes sense because the proposed method is only able to suggest rotation parameters that are multiples of its bin size; for example, for a bin size of 1°, the determined rotation parameters will always be multiples of 1°, while for a bin size of 5° the rotation parameters will always be multiples of 5°. So, clearly, a smaller bin size would be expected to result in a more accurate rotation parameter. While this observation is certainly true in the simulated ideal scenario, a small bin size (such as 1°) is impractical for real fingerprint samples. This is because the numbers of extracted minutiae tend to vary between different acquisitions of the same fingerprint; so, using too small a bin size (such as 1°) would make the alignment method very sensitive to even small differences between the numbers of extracted minutiae. This realistic fingerprint scenario is considered in more detail in Section A.4.2; however, some sensible choices for the bin size could be estimated from the results in

Table A.1. In particular, it appears that a bin size of 5° may be suitable, since it is expected to determine all rotation parameters correctly within 3° in the ideal scenario while it would probably not be too sensitive to small changes in the numbers of extracted minutiae in a realistic scenario. For partial fingerprints or fingerprints of exceedingly poor quality, where the minutiae extraction may not be as reliable, a larger bin size may need to be used for the proposed alignment method. For example, a bin size of 10° will, in an ideal scenario, still result in alignments accurate within 3° for 70% of the possible rotational differences between -60° and 60°. This would help in attaining a rough rotational alignment, which could be followed by alternative alignment methods for increased accuracy.

A.4.2 Experiment 2: Accuracy – Realistic Scenario

The results obtained for the ideal scenario (Section A.4.1) clearly indicate that the proposed alignment method has promise in terms of the attainable accuracy. The next step is to test the accuracy of the proposed alignment method for various bin sizes on a database of real fingerprint images, which was the aim of the experiment discussed in this section. For this experiment, the first fingerprint sample of each person from FVC2002 DB1_A was used as that person's reference fingerprint, and their third fingerprint sample served as the corresponding query fingerprint. The reason behind choosing these two images is that the third fingerprint sample of each person in this database is generally a deliberately rotated version of their first image, so this is suitable for testing the rotational alignment capabilities of the proposed method. In total, 200 fingerprints (2 fingerprints per person from 100 people) were used in this experiment. Let R_i and Q_i denote the sets of minutiae extracted from the first [reference] and third [query] fingerprint samples, respectively, of the i^{th} person in the database, where all minutiae were extracted using the VeriFinger SDK tutorial [211]. The idea here was to use the proposed alignment method to determine the amount of rotation required to align Q_i with R_i. This was repeated for each person in the database.

In order to decide whether or not the determined rotation parameter was correct, the "true" amount of rotation that must be applied to Q_i in order to align it with the corresponding R_i was manually calculated for each (Q, R) pair. This was done in the following way:

1) The same two points, p_1 and p_2, were chosen inside the reference fingerprint image (i.e., the first sample) and the query fingerprint image (i.e., the third sample). These two points were selected as follows:
   i. If both the core and delta points were present and their locations were correctly identified (by VeriFinger [211]) in both fingerprint images, then the core was assigned to p_1 and the delta to p_2.

ii. If only a core (delta) point was present and its location was correctly identified in both fingerprint images, then the core (delta) was assigned to p_1 and a minutia that was present in both fingerprint images was chosen and assigned to p_2.

iii. If neither a core nor a delta point was present (or their locations were incorrectly determined) in both of the fingerprint images, then two minutiae points that were present in both fingerprint images were chosen and assigned to p_1 and p_2.

2) The orientation (with respect to the horizontal axis) of the line connecting p_1 and p_2 inside each fingerprint image was calculated. Let R_iL_θ and Q_iL_θ represent the orientation of this line in the reference and query fingerprints, respectively, of person i.

3) The amount of rotation that must be applied to the minutiae in Q_i to [rotationally] align them with the minutiae in R_i was then calculated as $R_iL_\theta - Q_iL_\theta$. Note that a negative (positive) sign in front of the resulting rotation amount indicates that an anticlockwise (clockwise) rotation is required.

Upon determining the true rotation parameters, the rotation parameter for each (Q, R) pair was calculated using the proposed alignment method at three different bin sizes. The accuracy of the alignment method was computed based on how close the determined rotation parameter was to the true rotation parameter calculated for that particular pair of images. In total, 100 rotation parameters were determined and their accuracy was evaluated. The results are presented in Table A.2, and Table A.3 provides a comparison of the results from the realistic and ideal scenarios.

Table A.2: Accuracy of the proposed alignment method on a real fingerprint database, for various bin sizes.

Accuracy	Bin Size		
	3°	5°	10°
Within 1°	29%	34%	20%
Within 2°	48%	49%	33%
Within 3°	60%	66%	49%
Within 5°	79%	82%	73%
Within 10°	94%	93%	94%

Table A.3: A comparison of the accuracy of the proposed alignment method in the ideal and realistic scenarios.

Accuracy	Bin Size					
	3°		5°		10°	
	Ideal	Real	Ideal	Real	Ideal	Real
Within 2°	100%	48%	80%	49%	50%	33%
Within 3°	100%	60%	100%	66%	70%	49%

Several comments may be made regarding the results presented in Tables A.2 and A.3. From Table A.3, it is evident that the accuracy of the proposed alignment method is worse in this realistic scenario than in the ideal scenario, which is particularly striking for the bin size of 3°. However, the accuracy of the proposed alignment method in the realistic scenario appears to approach that of the ideal scenario as the bin size increases. This trend is expected, since, as mentioned in Section A.4.1, a small bin size is not likely to work well with real fingerprint images, due to its sensitivity to the varying numbers of minutiae between the reference and query fingerprints. The main reasons for the varying number of minutiae between multiple acquisitions of the same fingerprint are bad quality prints, from which it is difficult to reliably extract all the minutiae, and partial fingerprint images that do not contain all the minutiae extracted in a previous sample of the same fingerprint. One way to increase the accuracy of the proposed alignment method could be to consider only those minutiae that occur in overlapping parts of the reference and query fingerprint images. In this way, it is more likely that the same minutiae will be used for constructing the minutiae orientation histograms.

A more positive result from Table A.2 is that, for a bin size of 5°, the proposed alignment method was able to determine a rotation parameter that was within 5° of the correct rotation parameter in 82% of the tests. The bin size of 5° appears to be the most suitable choice for the rotational alignment of real fingerprint images using the proposed method, since it is able to provide the highest accuracy amongst the three bin sizes tested. This suggests that the alignment method has potential, provided that the bin size is carefully determined.

The most encouraging result from Table A.2 is that, at all three bin sizes, the proposed alignment method was able to determine a rotation parameter that was correct within 10° in more than 90% of the cases. This is very promising, because it suggests the possibility of using the proposed alignment method for a coarse rotational alignment (e.g., within 10°), which may then be fine-tuned using a more accurate alignment strategy.

A.4.3 Experiment 3: Speed

The aim of this experiment was to get an indication of how fast the proposed alignment method is, compared to a typical minutiae alignment technique. The exhaustive search alignment strategy was implemented in MATLAB and used to determine the rotation parameter for each of the 100 (Q, R) pairs employed in Experiment 2 (Section A.4.2). This method involved rotating Q_i from -60° to 60° in pre-determined step sizes, and calculating the number of minutiae correspondences (i.e., the match score) between R_i and Q_i at each rotation

of Q_i. The rotation that resulted in the maximum match score was determined to be the correct rotation parameter.

Note that in this experiment we were not interested in comparing the actual rotation parameters determined by the implemented typical alignment method and the proposed alignment method; we were only interested in how long it took to obtain those rotation parameters using each method. Fairness of the comparison was ensured in two ways. Firstly, the step size of the rotations used in the typical alignment method was set to be the same as the bin size of the proposed alignment technique. For example, the time taken to obtain the rotation parameters using the proposed alignment method with a bin size of 5° was compared to the time taken to determine the rotation parameters using the typical alignment method in which Q_i was rotated in steps of 5°. Secondly, the rotation parameter determined by the proposed alignment method was subsequently used to rotate the corresponding Q_i, and the match score between the aligned (Q_i, R_i) pair was calculated using the same matching function as the one employed in the implementation of the typical alignment method. Thus, the time taken for matching was incorporated into both of the alignment techniques being compared. The difference between the methods lies in the fact that the typical alignment technique calculates the match score at each possible rotation of Q_i, while the proposed alignment method calculates the match score only once, after the rotation parameter has been determined.

The time taken to determine the rotation parameters for all 100 (Q, R) pairs, as well as to match them, was estimated simply by using the "tic-toc" operation in MATLAB. Each set of 100 tests was run 10 times for bin/step sizes of 3°, 5°, and 10°, and the average time taken by the set of 100 tests for each bin/step size was calculated. The results are shown in Table A.4.

Table A.4: Average amount of time taken by the proposed and typical alignment methods (in MATLAB) to determine the rotation parameters for 100 (Q, R) pairs.

Alignment Method	Bin/Step Size		
	3°	5°	10°
Typical	17.48 sec	10.88 sec	7.20 sec
Proposed	1.49 sec	1.36 sec	1.27 sec

From Table A.4, it is evident that the proposed alignment method is considerably faster than the implemented exhaustive search alignment technique. This was expected, since the exhaustive search strategy must calculate the match score at each possible rotation of Q_i, while the proposed alignment technique calculates the match score only once, after the rotation parameter has been determined. From the results in Table A.4, it may be concluded

that the proposed alignment technique is able to determine the rotation parameter approximately 12, 8, and 7 times faster than the implemented exhaustive search alignment method for bin/step sizes of 3°, 5°, and 10°, respectively. It is evident that both alignment methods get faster with increasing bin/step size. This is because fewer rotations of Q_i (and therefore a smaller number of matching operations) need to be executed for the typical alignment method when using a larger step size, and a smaller number of shifts of H^{Qi} must be applied during the correlation stage in the proposed alignment method when a larger bin size is used for the histograms. While this speed-up is very obvious for the typical alignment method in Table A.4 as the step size increases from 3° to 10°, it is less evident for the proposed alignment method. This suggests that the speed of the proposed alignment method is not significantly affected by the bin size of the minutiae orientation histograms.

A.5 CONCLUSIONS AND FUTURE WORK

This appendix proposes a new method for correcting rotational offsets between two different samples of the same fingerprint. The proposed alignment method is based on circular correlations of histograms constructed using the orientations of the minutiae extracted from each fingerprint image.

The performance of the proposed alignment method was analysed via three experiments. The aim of the first experiment was to determine the accuracy of the alignment method in the ideal case, where the query minutiae are simply a rotated version of the reference minutiae. Results indicate extremely high accuracy for smaller bin sizes, and decreasing accuracy as the bin size is increased. This trend is a consequence of the alignment becoming coarser as the bin size gets larger. The second experiment tested the accuracy of the proposed alignment method on a real fingerprint database, for bin sizes of 3°, 5°, and 10°. As expected, the results were worse than in the ideal case, due to the sensitivity of this alignment method to varying numbers of minutiae between the reference and query fingerprints. However, it was shown that the proposed alignment technique has promise, provided that the histogram bin size is appropriately selected. In particular, a bin size of 5° was found to provide the highest accuracy out of the three bin sizes tested. A notable result is that, for a bin size of 5°, the proposed alignment method was able to determine the rotation parameter correctly within 5° in 82% of the cases tested. The most encouraging results obtained from this experiment indicate that the proposed alignment technique is capable of determining the rotation parameter with an accuracy of 10° over 90% of the time for all the bin sizes tested. This suggests the suitability of the proposed method for coarse rotational alignment. The third

experiment estimated the speed of the proposed alignment technique and compared it to an implementation of the typical exhaustive search alignment strategy. The proposed method was found to be considerably faster.

The results from all three experiments together indicate that the proposed alignment method is promising, and they suggest the possibility of a hybrid alignment technique, which employs the proposed method for fast, coarse rotational alignment of two minutiae templates, followed by fine-tuning using a more computationally expensive alignment strategy (such as the exhaustive search technique) for increased accuracy. This hybrid alignment method will be investigated as part of our future work, in which the accuracy of the implemented exhaustive search alignment strategy versus the accuracy of the proposed alignment technique will also be empirically evaluated.

The proposed alignment technique was designed with the initial aim of applying it only towards correcting rotational differences between two fingerprint samples. A potential avenue for future work could be an extension of the proposed alignment strategy to correct for translational offsets between two fingerprints as well. A possible way of implementing this extension could be via the construction of a 3D histogram, in which minutiae *x*-coordinates are binned along one dimension, minutiae *y*-coordinates are binned along another dimension, and minutiae orientations are binned along the third dimension. Correlation of 3D histograms generated from the reference and query fingerprint minutiae could then be used to simultaneously determine the horizontal and vertical translation, as well as the rotation, between the two fingerprint samples[67].

Note: *Our proposed alignment method is useful for fingerprint matching algorithms in general; therefore, we may assume that it may also be beneficial for the development of fingerprint template protection schemes. However, because the proposed method is a pre-alignment strategy (i.e., alignment first, followed by matching), this suggests that the minutiae orientation histogram corresponding to the reference fingerprint would need to be stored along with the protected fingerprint template in the database. While this would leak information pertaining to the minutiae orientations, minutiae locations would not be revealed (unless the proposed alignment method is extended to account for translational offsets too, as mentioned above). Nevertheless, a safe way of storing these minutiae histograms would need to be developed, or perhaps the minutiae histograms themselves could be used to generate the protected fingerprint template. While these are interesting research directions, they are not*

[67] Note that this point came to mind after the publication of the paper on which this appendix is based.

focused on in this thesis. This is because the major contribution of this thesis is a fingerprint template protection technique that uses only a small portion of a fingerprint's entire minutiae template; consequently, the alignment method proposed in [24] (and replicated in this appendix) cannot be applied in this context. Nevertheless, many other fingerprint template protection techniques do, and will in the future, require alignment, which is why we believe that the proposed alignment technique may be considered an important contribution to the development of fingerprint template protection schemes in general.

Appendix B

A Dissection of Fingerprint Fuzzy Vault Schemes

The material presented in this appendix essentially replicates our associated publication, [25].

The fuzzy vault construction is one of the most widely adopted approaches for the protection of fingerprint data. The popularity of this scheme stems from its ability to deal with unordered sets of fingerprint features, as well as its tolerance to missing or spurious feature elements across multiple acquisitions of the same fingerprint. While a considerable number of fingerprint-based fuzzy vault implementations have been reported in the literature, a review of these schemes does not yet exist, to the best of the authors' knowledge. This appendix, therefore, dissects existing fingerprint fuzzy vault schemes and provides a comprehensive discussion of what fingerprint features have been used, and how the locking and unlocking processes have been adapted to suit the nature of the fingerprint features employed.

B.1 INTRODUCTION

With the escalating interest in biometric recognition technologies, considerable growth in databases containing sensitive biometric information of millions of citizens is inevitable. This raises serious concerns about the security of the biometric data stored in these databases and consequently our privacy, particularly because a compromised biometric is forever rendered useless as a reliable identifier. In light of this issue, effective means of securing the biometric data during storage in a database are urgently required. The main challenge in the advancement of such protection schemes lies in the intra-class variability exhibited by biometric measurements acquired during different presentations to a recognition system. This hinders the direct use of traditional, well-established cryptographic techniques, which are

sensitive to small changes in the input data. For this reason, in the past decade there has been increasing focus on developing protection schemes specifically suited to the nature of biometric data, and an excellent review of these techniques is presented in [2].

While none of the methods proposed thus far has been embraced as the standard biometric protection mechanism in practice, several techniques have become the focus of much attention in the academic arena. One of the most widely adopted approaches for securing biometric data is the fuzzy vault scheme, which was first proposed in [154] in 2002. The attraction of using the fuzzy vault construction for securing biometric data lies in its ability to deal with unordered sets of features, which are common in biometric measurements, as well as its tolerance to erasures in the feature vector obtained during different acquisitions of the biometric trait. Since 2002, fuzzy vault implementations for a number of biometric modalities have been reported in the literature, with a predominant focus on fingerprints due to their maturity as a biometric identifier. To the best of the authors' knowledge, there does not yet exist a review of these fingerprint-based fuzzy vaults; so, this appendix presents a discussion of the methods employed in constructing the fingerprint fuzzy vault implementations documented in the literature. Section B.2 provides a step-by-step outline of the locking and unlocking processes in a general fuzzy vault construction; Section B.3 dissects existing fingerprint fuzzy vault schemes to show how the fuzzy vault construction has been adapted to suit the nature of fingerprint features; and Section B.4 imparts some concluding remarks.

B.2 THE FUZZY VAULT SCHEME

The fuzzy vault scheme is an error-tolerant cryptographic construction, which binds a secret, S, with an encryption key, E, such that both S and E are secured. If $E(S)$ denotes the encrypted version of S, then a corresponding decryption key, D, can be used to recover S from $E(S)$ if D is close enough to E. The encryption and decryption keys each consist of a set of finite field elements, and an important strength of the fuzzy vault scheme is that these elements need not be ordered; hence, E and D are true sets rather than sequences. The error-tolerance of the fuzzy vault scheme may then be formulated as follows: If ϵ is the maximum number of elements that are allowed to differ between sets E and D, then $D(E(S)) = S$ iff $|E - D| \leq \epsilon$, where ϵ is referred to as the "fuzziness" of the scheme. The encryption and decryption processes in the fuzzy vault construction are referred to as "locking" and "unlocking" the vault, respectively, and these are explained in more detail in sections B.2.1 and B.2.2.

B.2.1 Locking the Vault

Locking the fuzzy vault may be thought of as a three-step process, which essentially involves binding an encryption key, E, with a secret, S, and then adding lots of "noise" to $E(S)$ to conceal it from an attacker. It is assumed that E and S each consist of a set of elements from a finite field, $\mathbb{F}$. The elements in E may be arbitrarily ordered, while the elements in S form a specific sequence. As a toy example, consider the vault locking process when $S = <7, 2, 5, 1>$, $E = \{2, 9, 6, 1, 5\}$, and $\mathbb{F} = GF(2^{16})$.

Step 1: Select a k-degree polynomial, $p(x)$, and embed S into the coefficients of $p(x)$. Let $p(x) = \mathbf{7}x^3 + \mathbf{2}x^2 + \mathbf{5}x + \mathbf{1}$.

Step 2: Evaluate $p(x)$ on each element of E; so, the elements of E are treated as the abscissa values of $p(x)$. The resulting (x, y) coordinates are referred to as the genuine points, and these points form the locking set, L, of the fuzzy vault. Continuing with our example, $L = \{(E_i, p(E_i))\}$, $i = 1, 2, \ldots, 5$. So, $L = \{(2, 59), (9, 3921), (6, 351), (1, 1), (5, 409)\}$. Note that there should be more than k points in the locking set, because during vault unlocking at least $k + 1$ points must be identified in L in order to reconstruct $p(x)$. At this stage, the fuzzy vault, V, consists of only the locking set of elements, L, which is analogous to $E(S)$. However, storing L on its own is insecure, since it can be used to directly reconstruct $p(x)$ and thus obtain S from its coefficients.

Step 3: Add a set of random chaff points, C, to the vault in order to conceal L. Let $C = \{(3, 7), (58, 13), (1000, 79), (9876, 44), (88, 313)\}$. Note that all the x-coordinates in C are different from the x-coordinates in L, and none of the chaff points lie on the polynomial, i.e., $p(Cx) \neq Cy$, where $Cx = \{3, 58, 1000, 9876, 88\}$ and $Cy = \{7, 13, 79, 44, 313\}$.

Finally, the fuzzy vault, V, is a collection of all the genuine and chaff points in $\mathbb{F}$. This can be represented as $V = L \cup C$.

B.2.2 Unlocking the Vault

Unlocking the vault may be thought of as a two-step process, which essentially requires using a decryption key, D, to identify as many genuine points in the fuzzy vault as possible. The resulting points are referred to as the unlocking set, U, and they are used to interpolate the

secret polynomial, $p(x)$. Going back to our example, consider the vault unlocking process when $D = \{1, 2, 5, 9\}$, (D consists of a set of elements from $\mathbb{F}$):

Step 1: Compare each element from D with the x-coordinate of every point in V to find matching points. Every vault point whose x-coordinate has been matched to an element from D is placed into an unlocking set, U. Ideally, $D = E$, which means that $U = L$; however, it is possible that U will contain some chaff points. In our example, the elements in D match the x-coordinates of the following vault points: {(1, 1), (2, 59), (5, 409), (9, 3921)}; so, this set of vault points forms the unlocking set, U.

Step 2: Use U to interpolate a k-degree polynomial, $p'(x)$ (e.g., via Lagrange interpolation), and concatenate the coefficients of $p'(x)$ to obtain S'. In general, U and L must intersect on at least $k + 1$ points in order to ensure that $p'(x) = p(x)$. In our trivial example, $k = 3$, so there must be at least 4 points in common between U and L. Since this is the case here, $p'(x) = p(x)$, so $S' = S$, which means that authentication is successful. Due to space constraints, the interpolation process is omitted in this example.

B.3 FINGERPRINT FUZZY VAULTS

The vault locking and unlocking processes may be implemented using fingerprints as the encryption and decryption keys. In this case, E is a set of fingerprint features collected from a user during enrolment, and D is a set of fingerprint features collected from a user during authentication.

This section presents a discussion on the methods employed in the literature for the different stages of a fingerprint fuzzy vault construction. Since the actual locking and unlocking processes described for a general fuzzy vault in Section B.2 are essentially the same for any implementation, this section focuses only on those details that are specific to the fingerprint modality. In particular, the following two stages of the locking process are considered: the formation of an encryption key, E, from fingerprint features, and the generation of chaff points, C. For the unlocking process, discussion is centred on the generation of a decryption key, D, from fingerprint features, and on the method of establishing correspondences between elements of D and the vault points in order to create an unlocking set, U.

Note that any reported performance results are not discussed in this paper. This is because the results generally indicate the performance of the system as a whole, but the focus

of this paper is only on the methods applied in certain stages of the overall process, and such partial results are usually not available.

B.3.1 Fingerprint Features Used

The features most commonly used for locking and unlocking a fingerprint fuzzy vault are minutiae. In practice, a minutia is usually represented as a stand-alone point, with some or all of the following attributes: $\{(x, y), \theta, t\}$, where (x, y) corresponds to its location in the 2D image plane, θ is its orientation relative to the horizontal axis, and t is its type (bifurcation or termination). Most fingerprint fuzzy vault implementations adopt this representation of minutiae as stand-alone points, but they differ according to what attributes they use. Examples of implementations that employ: (i) only (x, y) coordinates of each minutia in Cartesian or polar format, include: [156, 160, 163, 232-236]; (ii) $\{(x, y), \theta\}$ of each minutia, include: [2, 157, 174, 237, 238]; (iii) $\{(x, y), \theta, t\}$ of each minutia, include: [162, 239-244].

Instead of representing each minutia as a stand-alone point, a minutia may be considered to be part of a structure. A minutia structure describes the minutia in terms of its surroundings (e.g., in terms of the distances and relative angles between itself and its n-nearest neighbours), which may offer richer discriminatory information than that provided by stand-alone point descriptions. The use of minutiae structures for locking and unlocking a fingerprint fuzzy vault has been proposed in [159, 166, 245, 246].

To the authors' knowledge, the only fingerprint fuzzy vault implementation that uses non-minutiae based fingerprint features is [164]. Since the major part of the fingerprint fuzzy vault literature focuses on minutiae, henceforth this paper will only consider minutiae-based fingerprint fuzzy vaults.

B.3.2 Locking the Vault

This section considers two fingerprint-specific parts of the vault locking process: the generation of an encryption key, E, from fingerprint minutiae, and the generation of chaff points, C.

B.3.2.1 Generating the Encryption Key, E

Let T denote the set of [template] minutiae extracted from a user during enrolment. Generating E from T then involves two steps: selecting a subset of reliable minutiae, T', from T, and encoding the elements of T' into a finite field, $\mathbb{F}$, to generate E.

While it is certainly possible to generate E using the entire set T, a number of researchers have suggested selecting a subset of only the most reliable minutiae for this purpose. Reliable minutiae are those that are most likely to be encountered in another sample of the same fingerprint acquired during authentication. Using only the most reliable minutiae for E may help to improve the performance of the fingerprint fuzzy vault, since we are more likely to have those same minutiae in the unlocking set. In the current literature on fingerprint fuzzy vaults, the most popular method of selecting a reliable set of minutiae, T', from T involves the use of quality indices. Specifically, a quality index is computed for each minutia in T, and only the r top-quality minutiae are selected and placed in T'. The first fingerprint fuzzy vault implementation to adopt this technique is described in [157]. This approach has subsequently been used in [2, 174, 237, 238]. Alternative ways of forming T', include: collecting multiple samples of the same fingerprint and selecting only those minutiae that appear in a certain number of these samples [156, 235]; sorting the minutiae coordinates in ascending order and then selecting the first r minutiae [163, 232]; and choosing only those minutiae that appear within a certain radius of the fingerprint core [235, 244]. Some implementations, such as [235], combine a number of filtering techniques. Note that most of the implementations referenced here also incorporate the additional constraint that the minutiae in T' must be well-separated (i.e., the minimum distance between any two minutiae must be greater than a pre-set threshold), which was first suggested in [157].

The minutiae in T' must next be encoded as elements of a finite field, F. For example, if the minutiae in T' are described simply in terms of their (x, y) coordinates, then a single finite field element must be generated from each (x, y) point. The set of all finite field elements resulting from the minutiae in T' shall be denoted as E, which is analogous to the encryption key, E, from Section B.2. Regardless of the format in which the minutiae in T' are represented, the most common approach of converting T' to E involves quantizing each minutia attribute and representing it as a bit string of a certain length, then concatenating the resulting bit strings to generate an m-bit element in $GF(2^m)$. The best explanation of this approach was first presented in [232], where each minutia x- and y-coordinate is quantized and represented as an 8-bit number, after which the resulting bit strings are concatenated to form a 16-bit element, $(x \mid y)$, of $GF(2^{16})$. For example, a minutia with location coordinates (15, 75) would be represented as 0000111101001011 (decimal value 3915), since the 8-bit binary values of 15 and 75 are 00001111 and 01001011, respectively. Fingerprint fuzzy vault implementations that adopt this encoding methodology, or a variant of it depending on what minutiae attributes they use, include: [2, 157, 160, 163, 174, 233, 234, 237, 238, 241, 242,

246]. A slightly different approach was presented in [245], where a hashing function is applied to attributes of a minutia structure to create 16-bit elements of E.

Note that the size of the finite field depends on the desired level of quantization of the minutiae attributes: the greater the field size, the finer the quantization. A finer quantization offers greater discrimination between minutiae, at the price of a lower tolerance to minutiae perturbations between different samples of the same fingerprint. Since minutiae attributes are extracted from a fingerprint image, a sensible quantization parameter would take into account the size of the underlying image. For instance, since minutiae (x, y) coordinates are deduced from the locations of the corresponding pixels in the image, then an 8-bit representation for each x- and y-coordinate is entirely sufficient for a 256×256 pixel image, since each x- and y-coordinate can take on 256 different values. Most fingerprint fuzzy vault implementations work in GF(2^{16}), and those approaches using additional minutiae attributes simply apply coarser quantization to each attribute; e.g., [157] represents a minutia in terms of its $\{(x, y), \theta\}$ and quantizes each attribute into a 6-bit, 5-bit, and 5-bit value, respectively, such that the concatenation $(x \mid y \mid \theta)$ produces a 16-bit element in GF(2^{16}). Compare this to the finer quantization of 8 bits per x- and y-coordinate used in [232].

An interesting variation in the generation of E from T' can be found in [235, 236]. In this approach, the minutiae from T' as well as the generated chaff points, C (see Section B.3.2.2), are first arranged in lexicographic order. Then, the *indices* of the points as they appear in that order, instead of the actual coordinates themselves, are encoded as elements of F. In this way, the size of the field is reduced to the range of the total number of points in the fuzzy vault (the field size would be the maximum index in the lexicographic order).

B.3.2.2 Generating Chaff Points, C

The final stage of the locking process involves generating a large number of chaff points in order to conceal the genuine points from an attacker. Chaff points are usually generated in one of two ways. One method entails the random generation of minutiae-like attributes for each chaff point, followed by the encoding of those attributes into F. For example, if each minutia in T' is represented by its (x, y) coordinates, then we begin by generating random (x, y) coordinates for each chaff point, followed by the encoding of each chaff (x, y) as an element in F. The resulting value in F is the chaff point's x-coordinate as it appears in the vault, and its corresponding vault y-coordinate is generated randomly in F; e.g., [157]. The second method is to simply generate the chaff point's vault (x, y) coordinates in F directly; e.g., [232].

The full version[68] of the seminal fuzzy vault paper [154] specifies two criteria that must be met when placing chaff points into the fuzzy vault: a chaff point's x-coordinate in $\mathcal{F}$ must not intersect with the x-coordinate in $\mathcal{F}$ of any genuine point, and the chaff points must not lie on the secret polynomial. In fingerprint fuzzy vault literature, the former constraint has generally been made stricter in order to minimize the chance of a query minutia being matched to a chaff point during vault unlocking. The first fingerprint fuzzy vault implementation published, [156], suggests randomly generating chaff points and keeping only those that are separated by a minimum distance, d, from any genuine minutiae in the vault as well as any previously added chaff points, where d is the distance inside which a query minutia and a vault point would be considered to match during authentication. This approach has subsequently been adopted as the standard way of chaff point generation in the fingerprint fuzzy vault literature, with any advancements being mostly variations of this method. However, [172] showed that placing chaff points in the manner described in [156] could enable an attacker to separate the genuine points from the chaff points faster than using a brute-force attack. Specifically, it is empirically shown that chaff points added later on tend to have a smaller free area than chaff points added earlier, and this information could enable an attacker to quickly filter out some of the chaff points. As a means of preventing such statistical attacks on the fuzzy vault, a modification of the chaff placement method from [156] was proposed in [247], where the authors suggest using two distance thresholds instead of one. The first distance threshold, d_1, places a constraint on the minimum separation between a chaff point and a genuine point, while the second distance threshold, d_2, controls the inter-chaff distance, where $d_2 < d_1$. In this way, chaff points may sometimes be close to each other, but they will be sufficiently separated from the genuine points to avoid a false match during authentication. The idea here is to generate chaff points such that their distribution more closely mimics that of the genuine minutiae (since minutiae may naturally appear close to other minutiae in some parts of the fingerprint), thereby making the chaff points more difficult to distinguish from the genuine points. Another suggestion to create a more 'natural' chaff point distribution was proposed in [246], where the chaff points are randomly selected from an estimated minutiae descriptor distribution. This is in contrast to picking chaff points randomly from a *uniform* distribution, as suggested in the full version of [154]. In [235], chaff points are placed only inside an ellipse centred on the fingerprint core, since the authors estimate that 83% of the minutiae in a fingerprint are concentrated in that region. While a minutiae-like chaff point distribution is certainly desirable, [236] observes that even if chaff

[68] Available at www.ari-juels.com or theory.lcs.mit.edu/-madhu

points are selected from a 'natural' distribution resembling that of minutiae, they are less likely to occupy the locations frequently occupied by the genuine minutiae in the vault, since the minutiae points claim those positions prior to chaff point selection.

The number of chaff points that can be added into the fuzzy vault is generally constrained by the size of the fingerprint image. Usually, the number of chaff points is chosen to be about an order of magnitude larger than the number of genuine minutiae, and a typical implementation parameter is around 200 chaff points [2, 157, 160, 162, 163, 174, 232, 233, 237, 238, 240, 241, 243]. A simple way of increasing the number of chaff points is to use a smaller distance threshold, d. However, this would be expected to have a negative effect on the recognition performance of the fuzzy vault system, since the chaff points may be too close to the genuine points. For this reason, [157] suggested using the orientation, θ, of each minutia point as well as its (x, y) coordinates, since this would allow for the generation of chaff points closer in location to genuine points but with different angles. This approach was adopted by [2, 174, 238], and [162, 240, 241, 243] incorporated the type attribute as well; however, most of these implementations still employed around 200 chaff points.

Two similar and interesting propositions to enable the addition of a larger number of chaff points were outlined in [239] and [242]. In [239], the authors use $\{x, y, z, \theta, t\}$ for each minutia point and thus each chaff point, where $z = \theta/\alpha$ (α was set to 18). They were able to generate 1,000 chaff points using this representation. In [242], a 3D lattice is first generated, where the x- and y-dimensions of the lattice correspond to the x- and y-dimensions of the fingerprint image, and the z-dimension corresponds to the minutiae orientations, θ, quantized into three sections of 120° each. The genuine minutiae points are placed into the lattice first and any remaining empty cells are filled by randomly generated chaff points. An average of 2,369 chaff points was added. While this approach of using additional information (such as minutiae angles) in order to generate more chaff points seems appealing at first glance, some evident drawbacks have been pointed out. For example, [235] states that randomly choosing the orientations of chaff points that are close in location to genuine points would, in some cases, reveal the chaff points. This is because minutiae in close proximity to one another tend to have similar orientations, so placing a chaff point with a very different orientation in the vicinity of a set of genuine minutiae would make it obvious that this is not a genuine point.

B.3.3 Unlocking the Vault

This section considers two fingerprint-specific parts of the vault unlocking process: the generation of a decryption key, D, from fingerprint minutiae, and the process of establishing

correspondences between elements of D and the vault points in order to create an unlocking set, U.

B.3.3.1 Generating the Decryption Key, D

Let Q denote the set of [query] minutiae extracted from a user who wishes to authenticate himself against a fuzzy vault stored in the system database. Generating D from Q then involves three steps: selecting a subset of reliable minutiae, Q', from Q, aligning Q' with the elements of T' to obtain the aligned query minutiae, Q_A', then encoding Q_A' as elements of $\mathbb{F}$ to obtain D.

The selection of a reliable subset, Q', of query minutiae to use in generating D has been proposed by some authors as a means of improving the performance of the fingerprint fuzzy vault. As in the generation of T' (see Section B.3.2.1), the most widely adopted approach for determining the reliable subset, Q', is to use quality indices in order to select only the r top-quality, well-separated minutiae from Q. This approach is used in [2, 157, 174, 235, 237, 238]. Two alternative ways of obtaining Q', include: dividing the fingerprint image into several areas and attempting to unlock the vault using a subset of minutiae from each region until one of the subsets succeeds [245]; and considering only the query minutiae inside a ring of particular radius around the core [244].

The next stage in generating D from Q' is to align Q' with the template minutiae, T'. Alignment is necessary since multiple impressions of the same fingerprint are likely to be rotated and translated versions of each other; therefore, before two fingerprints can be fairly compared, they must be aligned. While alignment is usually a non-trivial procedure in traditional fingerprint recognition systems, it is even more difficult in the fuzzy vault construction because we do not actually have access to the original template minutiae, T'.

In the fingerprint fuzzy vault literature, alignment is generally performed in one of four ways. The first and most trivial approach is to assume that pre-alignment of all fingerprint images in the database has been executed prior to constructing any fuzzy vaults. In this case, finding correspondences between a query fingerprint and the vault points becomes a simple point pattern matching problem. Implementations of the fingerprint fuzzy vault that assume pre-alignment include: [156, 232, 233, 235]. While pre-alignment is acceptable in research as it allows one to compare the performance of subsequent implementation stages more fairly, this approach is evidently not practical in a real-life application. This is because we would only have access to the secured fingerprint data (i.e., embedded in the fuzzy vault), not the original T', so alignment would have to be performed in the fuzzy vault domain. Having said

that, [235] suggests a strategy whereby each fingerprint is pre-aligned independently of any other fingerprints - such an approach could be employed in practice.

The second way of performing alignment in the fingerprint fuzzy vault is to store some additional information (often referred to as "helper data") about the template fingerprint, extract the same helper data from the query fingerprint, then use the helper data to align *Q'* with *T'*. The most popular alignment technique in this category was initially proposed in [163] and improved in [157]. This method calculates the maximum curvature points of the underlying flow of fingerprint ridges for both the template and query fingerprints, and then uses the iterative closest point (ICP) algorithm to align them during authentication. The improved alignment method in [157] has subsequently been adopted by [2, 174, 238]. Other notable alignment techniques in this category involve the use of helper data consisting of: minutiae structures [160, 167, 244]; and topological structures based on the core point and underlying fingerprint orientation field [237, 248].

A disadvantage of storing additional data is that it could potentially leak information about the original fingerprint. For example, [246] suggests that storing high curvature points for alignment purposes may leak some information that would help an attacker distinguish between genuine and chaff points in the vault. Specifically, they note that if chaff points with random orientations are added near the high curvature points, they can be easily identified since their orientations would be different from the orientation at the high curvature points. For this reason, several researchers have adopted a third way of alignment in their fingerprint fuzzy vault implementations, which does not rely on any information other than the features used to lock the vault. This technique attempts to represent the minutiae as translation and rotation invariant features, in order to eliminate the need for alignment altogether. Most publications in this area are related to the geometric hash table proposed in [162], where each point in the fuzzy vault is expressed relative to every other point (minutiae and chaff points). This technique, or variants of it, have been adopted in [239-242], mostly by the authors of [162]. Other translation and rotation invariant features proposed for alignment-free fingerprint fuzzy vaults are generally based on nearest-neighbour minutiae structures, and examples include: [166, 245, 246].

A disadvantage of most of the rotation and translation invariant minutiae representations mentioned above is that they require a much larger amount of storage space than the typical stand-alone minutiae point representations. In light of this limitation, as well as the problems associated with the pre-alignment and helper data based alignment techniques discussed earlier, a fourth alignment method has been adopted by some researchers in this field. This technique mimics the traditional manner in which fingerprint images are usually compared,

whereby an exhaustive search is used to align the fingerprints in the fuzzy vault domain [234, 235, 243]. In this approach, a number of different translations and rotations are applied to the query minutiae in Q' and a match score is calculated at each iteration, where the match score is a reflection of the number of correspondences established between the query minutiae in Q_A' and the x-coordinates of the vault points. The translation and rotation parameters that result in the highest match score are used to align the query fingerprint with the template fingerprint. An obvious disadvantage of this method is its high computational intensity, which depends on the number of translation and rotation steps at which the match score is calculated, as well as on the matching function used.

After alignment, the resulting Q_A' is either immediately encoded as elements of $\mathbb{F}$ to generate D, or else Q_A' is left in the minutiae-domain and D is generated only for those elements of Q_A' that are selected for the unlocking set, U. The choice in this matter depends on the matching method used in the establishment of the unlocking set, U, which is discussed in Section B.3.3.2.

B.3.3.2 Establishing the Unlocking Set, U

The elements in the aligned query minutiae set, Q_A', are next compared to the x-coordinate of each point in the vault in order to establish the unlocking set, U. The aim of the matching process is to filter out the chaff points. Matching is performed either in $\mathbb{F}$ or in the minutiae domain, with most researchers leaning towards the latter approach. The best example of an implementation using the former matching technique appears in [232], where the process begins by quantizing the attributes of the aligned minutiae in Q_A' and then encoding them into elements of $\mathbb{F}$ in order to generate the decryption key, D (this is analogous to the generation of E in Section B.3.2.1). Next, each element in D is compared to every vault x-coordinate, and if any vault x-coordinate is the same as an element in D, then that vault x-coordinate and its corresponding y-coordinate are added to the unlocking set, U. Examples of publications that have opted for matching in $\mathbb{F}$, include: [163, 166]. The problem with this matching technique is its intolerance to minutiae perturbations, which is a common form of intra-class variation in fingerprint samples acquired at different times. While some tolerance is provided via the quantization operation prior to encoding Q_A' into D, this is not a particularly robust way to deal with non-linear distortions in fingerprint images. A better alternative is to perform matching in the minutiae domain, which was first suggested in [157] and subsequently adopted by the majority of researchers in this field. The basic idea, as outlined clearly in [157], is to first decode the x-coordinate of each vault point in order to obtain its original minutiae attributes; e.g., in [157], each 16-bit vault x-coordinate is split up into 3 smaller

strings, which represent, respectively, $\{x, y, \theta\}$ in the minutiae domain. Note that the resulting $\{x, y, \theta\}$ values may either belong to a genuine minutia (i.e., a minutia from *T'*), or else a chaff point for which these attributes were randomly generated during vault locking. Matching is then carried out in the minutiae domain, where correspondences between query minutiae in Q_A' and vault points are established via some sort of distance measure or other similarity measure. The *x*-coordinates in $\mathbb{F}$ of all vault points that have a matching minutia from Q_A' are added to the unlocking set, *U*, along with their corresponding finite field *y*-coordinates. Examples of literature employing this general approach of performing matching in the minutiae domain, include: [2, 174, 233, 235, 237, 238, 240, 243, 246].

The reason that most researchers prefer to conduct matching in the minutiae domain is because this enables them to use traditional, well-established minutiae matchers, which have been specially designed to deal with the intra-class variation present between different samples of the same fingerprint. Recall that the "fuzziness" in a fingerprint fuzzy vault construction refers to its tolerance to *set differences* between *E* and *D* (or *T'* and *Q'*), which means its ability to deal with a certain number of missing or spurious minutiae in *Q'*. The fuzzy vault was *not* designed to deal with *variations in the value of a single element* in a set, which is why the establishment of *U* is better done in the minutiae domain using minutiae matchers, rather than in $\mathbb{F}$.

B.4 CONCLUSION

This appendix serves as a stepping stone towards a fuller appreciation of the fuzzy vault scheme amongst researchers. A discussion of the locking and unlocking processes in fingerprint-based fuzzy vault implementations in the literature has been provided. It has been established that minutiae are the most commonly used features in the construction of a fingerprint fuzzy vault, so only minutiae-based fuzzy vaults have been considered. For the locking process, the discussion was focused on the creation of an encryption key from minutiae, as well as on the constraints proposed for chaff point generation. For the unlocking process, the focus was on the generation of a decryption key from minutiae (including alignment techniques), as well as on the matching methods employed in the establishment of an unlocking set.

Note: *The focus of this thesis was not on the fuzzy vault scheme. Nevertheless, the initial stages of this research involved a detailed study of this fingerprint template protection strategy, and thus a contribution to this particular area has been made in the form of the*

publication in [25], which is replicated in this appendix. The purpose of this publication is to assist interested researchers in their own implementations of the fuzzy vault framework in the context of the fingerprint biometric. We believe that this is an important contribution, since researchers are often faced with the intimidating task of attempting to implement an accurate rendition of someone else's work. The most difficult part in this endeavour is knowing where to start. It is our hope that the material presented in this appendix and published in [25] will help lay the groundwork for this undertaking.

References

[1] D. Maltoni, *et al.*, "Fingerprint Analysis and Representation," in *Handbook of Fingerprint Recognition*, Second ed New York: Springer-Verlag, 2009.

[2] A. K. Jain, *et al.*, "Biometric template security," *EURASIP Journal on Advances in Signal Processing,* vol. 2008, pp. 1-17, 2008.

[3] D. Moon, *et al.*, "Improved cancelable fingerprint templates using minutiae-based functional transform," *Security and Communication Networks,* 2013.

[4] WinterGreen Research. (2013, 1 July, 2014). *Biometrics: Market Shares, Strategies, and Forecasts, Worldwide, 2013 to 2019* Available: http://www.reportsnreports.com/reports/270180-biometrics-market-shares-strategies-and-forecasts-worldwide-2013-to-2019.html

[5] D. Maltoni, *et al.*, *Handbook of Fingerprint Recognition*, First ed. New York: Springer-Verlag, 2003.

[6] National Science and Technology Council (NSTC), "Privacy & Biometrics: Building a Conceptual Foundation," in *Biometrics, Privacy, Progress and Government*, R. B. Jefferson, Ed., ed New York: Nova Science Publishers, Inc., 2010, pp. 119-150.

[7] C. J. Hill, "Risk of masquerade arising from the storage of biometrics," Bachelor of Science, Dept. of CS, Australian National University, Canberra, Australia, 2002.

[8] K. Saeed, "A Note on Problems with Biometrics Methodologies," in *International Conference on Biometrics and Kansei Engineering (ICBAKE)*, 2011, pp. 20-22.

[9] A. D. Smith, "Maintaining Secrecy when Information Leakage is Unavoidable," PhD, Massachusetts Institute of Technology, 2004.

[10] N. Owano. (2011, January 16, 2012). Engineers release car-seat identifier that reads your rear end. *PhysOrg.com*. Available: http://www.physorg.com/news/2011-12-unleash-car-seat-rear.html

[11] D. Maltoni, *et al.*, "Introduction," in *Handbook of Fingerprint Recognition*, Second ed New York: Springer-Verlag, 2009.

[12] The Wall Street Journal. (2014, 1 July, 2014). *The Global Government Biometric Systems Market 2014-2024*. Available: http://www.marketwatch.com/story/the-global-government-biometric-systems-market-2014-2024-2014-04-29

[13] D. Maltoni, *et al.*, "Fingerprint Classification and Indexing," in *Handbook of Fingerprint Recognition*, Second ed New York: Springer-Verlag, 2009.

[14] C. R. Kingston, "PROBABILISTIC ANALYSIS OF PARTIAL FINGERPRINT PATTERNS," D.Crim. 6503017, University of California, Berkeley, United States -- California, 1964.

[15] A. Ross, *et al.*, "Towards reconstructing fingerprints from minutiae points," in *Biometric Technology for Human Identification II*. vol. 5779, A. K. Jain and N. K. Ratha, Eds., ed Bellingham: Spie-Int Soc Optical Engineering, 2005, pp. 68-80.

[16] A. Adler, "Can images be regenerated from biometric templates?," presented at the The Biometric Consortium Conference, Hyatt Regency Crystal City, Arlington, VA, USA, 2003.

[17] R. Cappelli, *et al.*, "Fingerprint Image Reconstruction from Standard Templates," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 29, pp. 1489-1503, 2007.

[18] A. Ross, *et al.*, "From Template to Image: Reconstructing Fingerprints from Minutiae Points," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 29, pp. 544-560, 2007.

[19] K. Nandakumar, "Multibiometric systems: Fusion strategies and template security," PhD, Department of Computer Science and Engineering, Michigan State University, 2008.

[20] N. K. Ratha, *et al.*, "Enhancing security and privacy in biometrics-based authentication systems," *IBM Systems Journal,* vol. 40, pp. 614-634, 2001.

[21] V. Krivokuća, *et al.*, "A non-invertible cancellable fingerprint construct based on compact minutiae patterns," *International Journal of Biometrics,* vol. 6, pp. 125-142, 2014.

[22] V. Krivokuća, *et al.*, "Minutiae Persistence among Multiple Samples of the Same Person's Fingerprint in a Cooperative User Scenario," in *Proceedings of the 3rd International Conference on Pattern Recognition Applications and Methods*, ESEO, Angers, Loire Valley, France, 2014, pp. 76-86.

[23] V. Krivokuća and W. Abdulla, "Intra-Class Variance among Multiple Samples of the Same Person's Fingerprint in a Cooperative User Scenario," in *ICPRAM 2014 - Best Papers*, M. De Marsico, *et al.*, Eds., ed: Springer 2014.

[24] V. Krivokuća and W. Abdulla, "Fast fingerprint alignment method based on minutiae orientation histograms," in *Proceedings of the 27th Conference on Image and Vision Computing New Zealand*, Dunedin, New Zealand, 2012, pp. 486-491.

[25] V. Krivokuca, *et al.*, "A dissection of fingerprint fuzzy vault schemes," in *Proceedings of the 27th Conference on Image and Vision Computing New Zealand*, Dunedin, New Zealand, 2012, pp. 256-261.

[26] A. K. Jain and S. Pankanti, "A touch of money [biometric authentication systems]," *IEEE Spectrum,* vol. 43, pp. 22-27, 2006.

[27] M. Thieme, "Identifying and Reducing Privacy Risks in Biometric Systems," in *Presentation at 13th Annual Conference on Computers, Freedom & Privacy*, I. B. Group, Ed., ed. New York, 2003.
[28] A. K. Jain, *et al.*, "Biometrics: a tool for information security," *IEEE Transactions on Information Forensics and Security,* vol. 1, pp. 125-143, 2006.
[29] N. Ratha, *et al.*, "An Analysis of Minutiae Matching Strength," in *Audio- and Video-Based Biometric Person Authentication*. vol. 2091, J. Bigun and F. Smeraldi, Eds., ed: Springer Berlin / Heidelberg, 2001, pp. 223-228.
[30] B. Schneier, "Inside risks: the uses and abuses of biometrics," *Communications of the ACM,* vol. 42, p. 136, 1999.
[31] T. van der Putte, *et al.*, "Biometrical fingerprint recognition: don't get your fingers burned," in *IFIP TC8/WG8.8 Fourth Working Conference on Smart Card Research and Advanced Applications* 2000, pp. 289-303.
[32] T. Matsumoto, *et al.*, "Impact of artificial gummy fingers on fingerprint systems," in *Optical Security and Counterfeit Deterrence Techniques IV*, San Jose, CA, USA, 2002, pp. 275-289.
[33] T. Matsumoto, "Artificial Fingers and Irises: importance of Vulnerability Analysis," in *7th International Biometrics 2004 Conference and Exhibition*, London, UK, 2004.
[34] D. Willis and M. Lee, "Six Biometric Devices Point The Finger At Security," *Network Computing,* pp. 84-96, June 1, 1998.
[35] L. Thalheim, *et al.*, "Body check: biometric access protection devices and their programs put to the test," *c't,* vol. 11, p. 114ff, 2002.
[36] S. T. V. Parthasaradhi, "Comparison of classification methods for perspiration-based liveness algorithm," M.S.E.E. 1422091, West Virginia University, United States -- West Virginia, 2003.
[37] B. Toth, "Biometric liveness detection," *Information Security Bulletin,* vol. 10, pp. 291-297, 2005.
[38] R. Derakhshani, *et al.*, "Determination of vitality from a non-invasive biomedical measurement for use in fingerprint scanners," *Pattern Recognition,* vol. 36, pp. 383-396, 2003.
[39] R. Rowe, *et al.*, "Multispectral fingerprint image acquisition," *Advances in Biometrics,* pp. 3-23, 2008.
[40] A. Ross and A. Jain, "Information fusion in biometrics," *Pattern Recognition Letters,* vol. 24, pp. 2115-2125, 2003.
[41] A. Lumini and L. Nanni, "When fingerprints are combined with Iris-a case study: FVC2004 and CASIA," *International Journal of Network Security,* vol. 4, pp. 27–34, 2007.
[42] L. Hyeon Chang, *et al.*, "A New Mobile Multimodal Biometric Device Integrating Finger Vein and Fingerprint Recognition," in *Proceedings of the 4th International Conference on Ubiquitous Information Technologies & Applications (ICUT '09)* 2009, pp. 1-4.
[43] L. Hong and A. Jain, "Integrating faces and fingerprints for personal identification," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 20, pp. 1295-1307, 1998.
[44] F. Yang and B. Ma, "A New Mixed-Mode Biometrics Information Fusion Based-on Fingerprint, Hand-geometry and Palm-print," in *Fourth International Conference on Image and Graphics (ICIG 2007)* 2007, pp. 689-693.
[45] J. Fierrez-Aguilar, *et al.*, "Combining Multiple Matchers for Fingerprint Verification: A Case Study in FVC2004," in *Image Analysis and Processing – ICIAP 2005*. vol. 3617, F. Roli and S. Vitulano, Eds., ed: Springer Berlin / Heidelberg, 2005, pp. 1035-1042.
[46] U. Uludag and A. K. Jain, "Attacks on biometric systems: a case study in fingerprints," in *Security, Steganography, and Watermarking of Multimedia Contents VI*, San Jose, CA, USA, 2004, pp. 622-633.
[47] B. Gunsel, *et al.*, "Robust watermarking of fingerprint images," *Pattern Recognition,* vol. 35, pp. 2739-2747, 2002.
[48] M. M. Yeung and S. Pankanti, "Verification watermarks on fingerprint recognition and retrieval," *Journal of Electronic Imaging,* vol. 9, pp. 468-476, 2000.
[49] N. K. Ratha, *et al.*, "Secure data hiding in wavelet compressed fingerprint images," presented at the Proceedings of the 2000 ACM workshops on Multimedia, Los Angeles, California, United States, 2000.
[50] S. Jain, "Digital watermarking techniques: A case study in fingerprints and faces," in *Proceedings of the Indian conference on Computer Vision, Graphics and Image Processing (ICVGIP)*, Bangalore, India, 2000, pp. 139-144.
[51] U. Uludag, *et al.*, "A Spatial Method for Watermarking of Fingerprint Images," in *PRIS '01 Proceedings of the 1st International Workshop on Pattern Recognition in Information Systems: In conjunction with ICEIS 2001*, Setubal, Portugal, 2001, pp. 26–33.
[52] A. K. Jain, *et al.*, "Hiding a face in a fingerprint image," in *Proceedings of the 16th International Conference on Pattern Recognition*, 2002, pp. 756-759.
[53] A. K. Jain and U. Uludag, "Hiding biometric data," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 25, pp. 1494-1498, 2003.
[54] U. Uludag, "Secure biometric systems," Ph.D. 3248624, Michigan State University, United States -- Michigan, 2006.
[55] C. Soutar, "Biometric system security," *White Paper, Bioscrypt, http://www.bioscrypt.com*.

[56] A. Adler, "Images can be regenerated from quantized biometric match score data," in *Canadian Conference on Electrical and Computer Engineering*, 2004, pp. 469-472.
[57] R. Cappelli*, et al.*, "Synthetic fingerprint-image generation," in *Proceedings of the 15th International Conference on Pattern Recognition*, 2000, pp. 471-474.
[58] J. L. Araque*, et al.*, "Synthesis of fingerprint images," in *Proceedings of the 16th International Conference on Pattern Recognition*, 2002, pp. 422-425.
[59] E. Mordini and S. Massari, "Body, biometrics and identity," *Bioethics,* vol. 22, pp. 488-498, 2008.
[60] H. Chen and C. Zapata, *Medical Genetics Handbook*: WH Green, 1988.
[61] J. D. Woodward, "Biometrics: privacy's foe or privacy's friend?," *Proceedings of the IEEE,* vol. 85, pp. 1480-1492, 1997.
[62] A. Kong, "Palmprint identification based on generalization of IrisCode," Ph.D. NR34516, University of Waterloo (Canada), Canada, 2007.
[63] J. D. Woodward, "Biometrics: Identifying Law & Policy Concerns," in *Biometrics*, A. K. Jain*, et al.*, Eds., ed: Springer US, 2002, pp. 385-405.
[64] A. B. J. Teoh and D. C. L. Ngo, "Cancellable biometerics featuring with tokenised random number," *Pattern Recognition Letters,* vol. 26, pp. 1454-1460, Jul 2005.
[65] N. Ratha*, et al.*, "Cancelable Biometrics: A Case Study in Fingerprints," in *ICPR 2006: 18th International Conference on Pattern Recognition*, 2006, pp. 370-373.
[66] S. Yang, "Design and implementation for secure embedded biometric authentication systems," Ph.D. Doctoral Thesis, University of California, United States -- California, 2007.
[67] Q. Gao, "Secure biometrics," Ph.D. 3303793, City University of New York, United States -- New York, 2008.
[68] T. Boult and R. Woodworth, "Privacy and Security Enhancements in Biometrics," *Advances in Biometrics,* pp. 423-445, 2008.
[69] R. M. Bolle*, et al.*, "Biometric perils and patches," *Pattern Recognition,* vol. 35, pp. 2727-2738, 2002.
[70] D. Maltoni*, et al.*, "Securing Fingerprint Systems," in *Handbook of Fingerprint Recognition*, Second ed New York: Springer-Verlag, 2009.
[71] J. H. Burrows, "Secure Hash Standard," ed: DEPARTMENT OF COMMERCE WASHINGTON DC, 1995.
[72] L. Nanni and A. Lumini, "Cancellable Biometrics: Problems and Solutions for Improving Accuracy," in *Biometrics: Methods, Applications and Analysis*, H. Schuster and W. Metzger, Eds., ed New York: Nova Science Publishers, Inc., 2010, pp. 153-166.
[73] Y. Sutcu*, et al.*, "How to protect biometric templates," in *Security, Steganography, and Watermarking of Multimedia Contents IX*, San Jose, CA, USA, 2007.
[74] A. T. B. Jin*, et al.*, "Biohashing: two factor authentication featuring fingerprint data and tokenised random number," *Pattern Recognition,* vol. 37, pp. 2245-2255, 2004.
[75] R. Belguechi*, et al.*, "Biohashing for Securing Minutiae Template," in *20th International Conference on Pattern Recognition (ICPR)*, 2010, pp. 1168-1171.
[76] A. B. J. Teoh*, et al.*, "An Integrated Dual Factor Authenticator Based on the Face Data and Tokenised Random Number," in *Biometric Authentication*. vol. 3072, D. Zhang and A. K. Jain, Eds., ed: Springer Berlin / Heidelberg, 2004, pp. 1-19.
[77] A. B. J. Teoh*, et al.*, "Random Multispace Quantization as an Analytic Mechanism for BioHashing of Biometric and Random Identity Inputs," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 28, pp. 1892-1901, 2006.
[78] N. Saini and A. Sinha, "Optics based Biohashing using joint transform correlator," *Optics Communications,* vol. 283, pp. 894-902, 2010.
[79] T. Connie*, et al.*, "PalmHashing: a novel approach for cancelable biometrics," *Information Processing Letters,* vol. 93, pp. 1-5, 2005.
[80] Y. Pang*, et al.*, "Palmprint based cancelable biometric authentication system," *International Journal of Signal Processing,* vol. 1, pp. 98-104, 2004.
[81] T. Connie*, et al.*, "PalmHashing: a novel approach for dual-factor authentication," *Pattern Analysis & Applications,* vol. 7, pp. 255-268, 2004.
[82] C. Siew Chin*, et al.*, "High security Iris verification system based on random secret integration," *Computer Vision and Image Understanding,* vol. 102, pp. 169-177, 2006.
[83] A. B. J. Teoh*, et al.*, "Personalised cryptographic key generation based on FaceHashing," *Computers & Security,* vol. 23, pp. 606-614, 2004.
[84] D. Maio and L. Nanni, "Multihashing, human authentication featuring biometrics data and tokenized random number: A case study FVC2004," *Neurocomputing,* vol. 69, pp. 242-249, 2005.
[85] A. Teoh*, et al.*, "Remarks on BioHash and its mathematical foundation," *Information Processing Letters,* vol. 100, pp. 145-150, 2006.
[86] A. T. B. Jin and T. Connie, "Remarks on BioHashing based cancelable biometrics in verification system," *Neurocomputing,* vol. 69, pp. 2461-2464, 2006.

[87] A. Lumini and L. Nanni, "An improved BioHashing for human authentication," *Pattern Recognition,* vol. 40, pp. 1057-1065, 2007.
[88] A. B. J. Teoh*, et al.*, "Cancellable biometrics and annotations on BioHash," *Pattern Recognition,* vol. 41, pp. 2034-2044, Jun 2008.
[89] K. H. Cheung*, et al.*, "An analysis on invertibility of cancelable biometrics based on BioHashing," *CISST 2005,* pp. 40-45, 2005.
[90] A. Kong*, et al.*, "An analysis of BioHashing and its variants," *Pattern Recognition,* vol. 39, pp. 1359-1368, 2006.
[91] K.-H. Cheung*, et al.*, "Revealing the Secret of FaceHashing," in *Advances in Biometrics*. vol. 3832, D. Zhang and A. Jain, Eds., ed: Springer Berlin / Heidelberg, 2005, pp. 106-112.
[92] L. Nanni and A. Lumini, "A multi-modal method based on the competitors of FVC2004 and on palm data combined with tokenised random numbers," *Pattern Recognition Letters,* vol. 29, pp. 1344-1350, 2008.
[93] L. Nanni and A. Lumini, "Random subspace for an improved BioHashing for face authentication," *Pattern Recognition Letters,* vol. 29, pp. 295-300, 2008.
[94] H. Al-Assam*, et al.*, "Multi-factor biometrics for authentication: a false sense of security," presented at the Proceedings of the 12th ACM workshop on Multimedia and security, Roma, Italy, 2010.
[95] A. Nagar*, et al.*, "Biometric Template Transformation: A Security Analysis," in *Media Forensics and Security II*, San Jose, CA, USA, 2010.
[96] L. Yongjin*, et al.*, "Inverse operation and preimage attack on BioHashing," in *IEEE Workshop on Computational Intelligence in Biometrics: Theory, Algorithms, and Applications (CIB 2009)* 2009, pp. 92-97.
[97] W. Yongjin and K. N. Plataniotis, "Face Based Biometric Authentication with Changeable and Privacy Preservable Templates," in *Biometrics Symposium, 2007*, 2007, pp. 1-6.
[98] X. Zhou and T. Kalker, "On the security of biohashing," in *Media Forensics and Security II*, San Jose, CA, USA, 2010, p. 75410Q.
[99] A. Teoh*, et al.*, "Cancellable biometrics and user-dependent multi-state discretization in BioHash," *Pattern Analysis & Applications,* vol. 13, pp. 301-307, 2010.
[100] M. Khan*, et al.*, "Privacy-preserving and tokenless chaotic revocable face authentication scheme," *Telecommunication Systems,* pp. 1-8, 2010.
[101] A. T. B. Jin, "Cancellable Biometrics and Multispace Random Projections," in *Conference on Computer Vision and Pattern Recognition Workshop (CVPRW '06)*, 2006, pp. 164-164.
[102] A. Beng Jin Teoh and Y. Chong Tze, "Cancelable Biometrics Realization With Multispace Random Projections," *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on,* vol. 37, pp. 1096-1106, 2007.
[103] M. Savvides*, et al.*, "Cancelable biometric filters for face recognition," in *Proceedings of the 17th International Conference on Pattern Recognition (ICPR 2004)* 2004, pp. 922-925.
[104] S. Hirata and K. Takahashi, "Cancelable Biometrics with Perfect Secrecy for Correlation-Based Matching," in *Advances in Biometrics*. vol. 5558, M. Tistarelli and M. Nixon, Eds., ed: Springer Berlin / Heidelberg, 2009, pp. 868-878.
[105] Z. Jinyu*, et al.*, "Cancelable iris biometric," in *19th International Conference on Pattern Recognition (ICPR 2008)*, 2008, pp. 1-4.
[106] Q. Feng*, et al.*, "Cracking Cancelable Fingerprint Template of Ratha," in *International Symposium on Computer Science and Computational Technology (ISCSCT '08)*, 2008, pp. 572-575.
[107] S. Shin*, et al.*, "Dictionary attack on functional transform-based cancelable fingerprint templates," *ETRI journal,* vol. 31, pp. 628-630, 2009.
[108] R. Ang*, et al.*, "Cancelable Key-Based Fingerprint Templates," in *Information Security and Privacy*. vol. 3574, C. Boyd and J. M. González Nieto, Eds., ed: Springer Berlin / Heidelberg, 2005, pp. 242-252.
[109] L. Chulhan*, et al.*, "Alignment-Free Cancelable Fingerprint Templates Based on Local Minutiae Information," *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics,* vol. 37, pp. 980-992, 2007.
[110] S. Chikkerur*, et al.*, "Generating Registration-free Cancelable Fingerprint Templates," in *2nd IEEE International Conference on Biometrics: Theory, Applications and Systems (BTAS 2008)*, 2008, pp. 1-6.
[111] Y. Bian and C. Busch, "Parameterized geometric alignment for minutiae-based fingerprint template protection," in *BTAS 2009: IEEE 3rd International Conference on Biometrics: Theory, Applications, and Systems*, 2009, pp. 1-6.
[112] B. Yang*, et al.*, "Geometric-Aligned Cancelable Fingerprint Templates," in *Image Analysis and Processing – ICIAP 2009*. vol. 5716, P. Foggia*, et al.*, Eds., ed: Springer Berlin / Heidelberg, 2009, pp. 490-499.
[113] Y. Sutcu*, et al.*, "A secure biometric authentication scheme based on robust hashing," presented at the Proceedings of the 7th workshop on Multimedia and security, New York, NY, USA, 2005.

[114] S. Tulyakov, *et al.*, "Symmetric Hash Functions for Fingerprint Minutiae," in *Pattern Recognition and Image Analysis*. vol. 3687, S. Singh, *et al.*, Eds., ed: Springer Berlin / Heidelberg, 2005, pp. 30-38.
[115] F. Farooq, *et al.*, "Anonymous and Revocable Fingerprint Recognition," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR '07)*, 2007, pp. 1-7.
[116] A. B. J. Teoh and D. C. L. Ngo, "Biophasor: Token Supplemented Cancellable Biometrics," in *9th International Conference on Control, Automation, Robotics and Vision (ICARCV '06)* 2006, pp. 1-5.
[117] A. Teoh, *et al.*, "2^N Discretisation of BioPhasor in Cancellable Biometrics," in *Advances in Biometrics*. vol. 4642, S.-W. Lee and S. Li, Eds., ed: Springer Berlin / Heidelberg, 2007, pp. 435-444.
[118] J. MinYi, *et al.*, "Changeable Biometrics for Appearance Based Face Recognition," in *2006 Biometrics Symposium: Special Session on Research at the Biometric Consortium Conference*, 2006, pp. 1-5.
[119] K. Youngsung and T. Kar-Ann, "A Method to Enhance Face Biometric Security," in *First IEEE International Conference on Biometrics: Theory, Applications, and Systems (BTAS 2007)*, 2007, pp. 1-6.
[120] K. Youngsung and T. Kar-Ann, "Sparse random projection for efficient cancelable face feature extraction," in *3rd IEEE Conference on Industrial Electronics and Applications (ICIEA 2008)*, 2008, pp. 2139-2144.
[121] Y. Kim, *et al.*, "A performance driven methodology for cancelable face templates generation," *Pattern Recognition,* vol. 43, pp. 2544-2559, 2010.
[122] M. A. Dabbah, *et al.*, "Appearance-Based Biometric Recognition: Secure Authentication and Cancellability," in *15th International Conference on Digital Signal Processing*, 2007, pp. 479-482.
[123] M. A. Dabbah, *et al.*, "Secure Authentication for Face Recognition," in *IEEE Symposium on Computational Intelligence in Image and Signal Processing (CIISP 2007)*, 2007, pp. 121-126.
[124] M. A. Dabbah, *et al.*, "Image-based facial recognition in the domain of high-order polynomial one-way mapping," *Image Processing, IET,* vol. 2, pp. 139-149, 2008.
[125] M. A. Dabbah, *et al.*, "One-way Randomised Radon mapping for appearance-based biometric authentication," in *5th International Conference on Visual Information Engineering (VIE 2008)* 2008, pp. 276-281.
[126] M. A. Dabbah, *et al.*, "Randomized Radon Signature for face biometric verification," in *15th IEEE International Conference on Image Processing (ICIP 2008)*, 2008, pp. 273-276.
[127] M. A. Dabbah, *et al.*, "PCA Authentication of Facial Biometric in the Secure Randomized Mapping Domain," in *3rd International Conference on Information and Communication Technologies: From Theory to Applications (ICTTA 2008)*, 2008, pp. 1-5.
[128] E. Maiorana, *et al.*, "Template protection for HMM-based on-line signature authentication," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops (CVPRW '08)*, 2008, pp. 1-6.
[129] E. Maiorana, *et al.*, "Cancelable Biometrics for HMM-based Signature Recognition," in *2nd IEEE International Conference on Biometrics: Theory, Applications and Systems (BTAS 2008)*, 2008, pp. 1-6.
[130] E. Maiorana, *et al.*, "Cancelable Templates for Sequence-Based Biometrics with Application to On-line Signature Recognition," *IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans,* vol. 40, pp. 525-538, 2010.
[131] J. Bringer, *et al.*, "Anonymous identification with cancelable biometrics," in *Proceedings of 6th International Symposium on Image and Signal Processing and Analysis (ISPA 2009)* 2009, pp. 494-499.
[132] N. K. Ratha, *et al.*, "Generating cancelable fingerprint templates," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 29, pp. 561-572, 2007.
[133] T. Ahmad and H. Fengling, "Cartesian and polar transformation-based cancelable fingerprint template," in *IECON 2011: 37th Annual Conference on IEEE Industrial Electronics Society*, 2011, pp. 373-378.
[134] T. Ahmad, *et al.*, "Pair-polar coordinate-based cancelable fingerprint templates," *Pattern Recognition,* vol. 44, pp. 2555-2564, 2011.
[135] B. Yang, *et al.*, "Robust minutiae hash for fingerprint template protection," in *Proceedings of SPIE*, 2010, pp. 75410R1-75410R9.
[136] Y. Bian, *et al.*, "Non-invertible geometrical transformation for fingerprint minutiae template protection," in *IWSCN 2009: Proceedings of the 1st International Workshop on Security and Communication Networks*, 2009, pp. 1-7.
[137] S. Jinyang, *et al.*, "Biomapping: Privacy trustworthy biometrics using noninvertible and discriminable constructions," in *ICPR 2008: 19th International Conference on Pattern Recognition*, 2008, pp. 1-4.
[138] Z. Jin, *et al.*, "Secure Minutiae-Based Fingerprint Templates Using Random Triangle Hashing," in *Visual Informatics: Bridging Research and Practice*. vol. 5857, H. Badioze Zaman, *et al.*, Eds., ed: Springer Berlin Heidelberg, 2009, pp. 521-531.
[139] Z. Jin, *et al.*, "Fingerprint template protection with minutiae-based bit-string for security and privacy preserving," *Expert Systems with Applications,* vol. 39, pp. 6157-6167, 2012.

[140] Z. Jin, *et al.*, "Generating revocable fingerprint template using polar grid based 3-tuple quantization technique," in *IEEE 54th International Midwest Symposium on Circuits and Systems (MWSCAS)*, 2011, pp. 1-4.
[141] C. Lee and J. Kim, "Cancelable fingerprint templates using minutiae-based bit-strings," *Journal of Network and Computer Applications,* vol. 33, pp. 236-246, 2010.
[142] W. Wong, *et al.*, "Multi-line code: A low complexity revocable fingerprint template for cancelable biometrics," *Journal of Central South University,* vol. 20, pp. 1292-1297, 2013.
[143] Y. Sutcu, *et al.*, "A Geometric Transformation to Protect Minutiae-Based Fingerprint Templates," in *Biometric Technology for Human Identification IV: Proceedings of SPIE Defense, Security, and Sensing*, Orlando, FL, USA, 2007, pp. 65390E-1.
[144] Y. Huijuan, *et al.*, "Generating secure cancelable fingerprint templates using local and global features," in *ICCSIT 2009: 2nd IEEE International Conference on Computer Science and Information Technology*, 2009, pp. 645-649.
[145] T. Ahmad and J. Hu, "Generating Cancelable Biometric Templates Using a Projection Line," presented at the 11th International Conference on Control, Automation, Robotics and Vision, Singapore, 2010.
[146] T. Ahmad, *et al.*, "String-based cancelable fingerprint templates," in *6th IEEE Conference on Industrial Electronics and Applications (ICIEA)*, 2011, pp. 1028-1033.
[147] Z. Jin, *et al.*, "A Revocable Fingerprint Template for Security and Privacy Preserving," *KSII Transactions on Internet and Information Systems,* vol. 4, pp. 1327-1342, 2010.
[148] J. Zhe, *et al.*, "Generating revocable fingerprint template using minutiae pair representation," in *2nd International Conference on Education Technology and Computer (ICETC)*, 2010, pp. V5-251-V5-255.
[149] S. Wang and J. Hu, "Design of alignment-free cancelable fingerprint templates via curtailed circular convolution," *Pattern Recognition,* vol. 47, pp. 1321-1329, 2014.
[150] M. Ferrara, *et al.*, "Noninvertible Minutia Cylinder-Code Representation," *IEEE Transactions on Information Forensics and Security,* vol. 7, pp. 1727-1737, 2012.
[151] D. Ahn, *et al.*, "Matching with Secure Fingerprint Templates Using Non-invertible Transform," in *CISP 2008: Congress on Image and Signal Processing*, 2008, pp. 29-33.
[152] U. Uludag, *et al.*, "Biometric cryptosystems: issues and challenges," *Proceedings of the IEEE,* vol. 92, pp. 948-960, 2004.
[153] A. Cavoukian and A. Stoianov, "Biometric Encryption: A Positive-Sum Technology That Achieves Strong Authentication, Security *and* Privacy," Office of the Information and Privacy Commissioner, Ontario, Toronto March 2007.
[154] A. Juels and M. Sudan, "A fuzzy vault scheme," in *Proceedings of the IEEE International Symposium on Information Theory (ISIT 2002)*, 2002, p. 408.
[155] A. Juels and M. Wattenberg, "A fuzzy commitment scheme," in *Proceedings of the 6th ACM conference on Computer and communications security*, 1999, pp. 28-36.
[156] T. C. Clancy, *et al.*, "Secure Smartcard-Based Fingerprint Authentication," in *WBMA '03*, Berkley, CA, 2003, pp. 45-52.
[157] K. Nandakumar, *et al.*, "Fingerprint-Based Fuzzy Vault: Implementation and Performance," *IEEE Transactions on Information Forensics and Security,* vol. 2, pp. 744-757, 2007.
[158] A. Nagar, *et al.*, "Securing fingerprint template: Fuzzy vault with minutiae descriptors," in *19th International Conference on Pattern Recognition (ICPR 2008)*, 2008, pp. 1-4.
[159] U. Uludag and A. K. Jain, "Fuzzy Fingerprint Vault," in *BCTP '04*, Cambridge, UK, 2004, pp. 13–16.
[160] S. Yang and I. M. Verbauwhede, "Secure fuzzy vault based fingerprint verification system," in *Conference Record of the 38th Asilomar Conference on Signals, Systems and Computers*, Monterey, CA, 2004, pp. 577-581
[161] Y. Shenglin and I. Verbauwhede, "Automatic secure fingerprint verification system based on fuzzy vault scheme," in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP '05)*, 2005, pp. 609-612.
[162] Y. Chung, *et al.*, "Automatic Alignment of Fingerprint Features for Fuzzy Fingerprint Vault," in *CISC '05*, Beijing, China, 2005, pp. 358-369.
[163] U. Uludag and J. Anil, "Securing Fingerprint Template: Fuzzy Vault with Helper Data," in *CVPRW '06*, New York City, NY, 2006, pp. 163-163.
[164] A. Nagar and S. Chaudhury, "Biometrics based Asymmetric Cryptosystem Design Using Modified Fuzzy Vault Scheme," in *ICPR '06*, Hong Kong, 2006, pp. 537-540.
[165] U. Uludag and A. K. Jain, "Fingerprint-based Fuzzy Vault (online presentation)," ed, 2005.
[166] J. Jeffers and A. Arakala, "Minutiae-Based Structures for A Fuzzy Vault," presented at the Biometrics Symposium, BCC '06, Baltimore, MD, 2006.
[167] J. Jeffers and A. Arakala, "Fingerprint Alignment for A Minutiae-Based Fuzzy Vault," presented at the Biometrics Symposium, BCC '07, Baltimore, MD, 2007.
[168] Y. Lee, *et al.*, "Biometric Key Binding: Fuzzy Vault Based on Iris Images," in *Advances in Biometrics*. vol. 4642, S.-W. Lee and S. Li, Eds., ed: Springer Berlin / Heidelberg, 2007, pp. 800-808.

[169] F. Yi Cheng and P. C. Yuen, "Protecting Face Biometric Data on Smartcard with Reed-Solomon Code," in *Conference on Computer Vision and Pattern Recognition Workshop (CVPRW '06)*, 2006, pp. 29-29.
[170] M. Freire-Santos, *et al.*, "Cryptographic key generation using handwritten signature," in *Proceedings of SPIE: Biometric Technologies for Human Identification III*, Orlando (Kissimmee), FL, USA, 2006, pp. 62020N-7.
[171] K. Nandakumar and A. K. Jain, "Multibiometric Template Security Using Fuzzy Vault," in *2nd IEEE International Conference on Biometrics: Theory, Applications and Systems (BTAS 2008)*, 2008, pp. 1-6.
[172] E.-C. Chang, *et al.*, "Finding the original point set hidden among chaff," in *ASIACCS '06*, Taipei, Taiwan, 2006, pp. 182-188.
[173] W. J. Scheirer and T. E. Boult, "Cracking Fuzzy Vaults and Biometric Encryption," in *Biometrics Symposium, 2007*, 2007, pp. 1-6.
[174] K. Nandakumar, *et al.*, "Hardening Fingerprint Fuzzy Vault Using Password," in *ICB '07*, Seoul, Korea, 2007, pp. 927-937.
[175] A. Kholmatov and B. Yanikoglu, "Realization of Correlation Attack Against the Fuzzy Vault Scheme," in *Security, Forensics, Steganography, and Watermarking of Multimedia Contents X*, San Jose, CA, USA, 2008.
[176] G. I. Davida, *et al.*, "On enabling secure applications through off-line biometric identification," in *Proceedings of the 1998 IEEE Symposium on Security and Privacy*, 1998, pp. 148-157.
[177] F. Monrose, *et al.*, "Password hardening based on keystroke dynamics," in *Proceedings of the 6th ACM conference on Computer and communications security*, Kent Ridge Digital Labs, Singapore, 1999, pp. 73-82.
[178] F. Monrose, *et al.*, "Cryptographic key generation from voice," in *Proceedings of the 2001 IEEE Symposium on Security and Privacy*, 2001, pp. 202-213.
[179] F. Hao, *et al.*, "Combining Crypto with Biometrics Effectively," *IEEE Transactions on Computers,* vol. 55, pp. 1081-1088, 2006.
[180] E. Kelkboom, *et al.*, "“3D Face”: Biometric Template Protection for 3D Face Recognition," in *Advances in Biometrics*. vol. 4642, S.-W. Lee and S. Li, Eds., ed: Springer Berlin / Heidelberg, 2007, pp. 566-573.
[181] J.-P. Linnartz and P. Tuyls, "New Shielding Functions to Enhance Privacy and Prevent Misuse of Biometric Templates," in *Audio- and Video-Based Biometric Person Authentication*. vol. 2688, J. Kittler and M. Nixon, Eds., ed: Springer Berlin / Heidelberg, 2003, pp. 1059-1059.
[182] P. Tuyls, *et al.*, "Practical Biometric Authentication with Template Protection," in *Audio- and Video-Based Biometric Person Authentication*. vol. 3546, T. Kanade, *et al.*, Eds., ed: Springer Berlin / Heidelberg, 2005, pp. 436-446.
[183] Y. Dodis, *et al.*, "Fuzzy Extractors: How to Generate Strong Keys from Biometrics and Other Noisy Data," in *Advances in Cryptology - EUROCRYPT 2004*, 2004, pp. 523-540.
[184] Y. Dodis, *et al.*, "Fuzzy Extractors: How to Generate Strong Keys from Biometrics and Other Noisy Data," *SIAM Journal on Computing,* vol. 38, pp. 97-139, 2008.
[185] C. Vielhauer, *et al.*, "Biometric hash based on statistical features of online signatures," in *Proceedings of the 16th International Conference on Pattern Recognition*, 2002, pp. 123-126 vol.1.
[186] C. Yao-Jen, *et al.*, "Biometrics-based cryptographic key generation," in *2004 IEEE International Conference on Multimedia and Expo (ICME '04)*, 2004, pp. 2203-2206 Vol.3.
[187] E.-C. Chang and S. Roy, "Robust Extraction of Secret Bits from Minutiae," in *Advances in Biometrics*. vol. 4642, S.-W. Lee and S. Li, Eds., ed: Springer Berlin / Heidelberg, 2007, pp. 750-759.
[188] A. Arakala, *et al.*, "Fuzzy Extractors for Minutiae-Based Fingerprint Authentication," in *Advances in Biometrics*. vol. 4642, S.-W. Lee and S. Li, Eds., ed: Springer Berlin / Heidelberg, 2007, pp. 760-769.
[189] Y. Sutcu, *et al.*, "Secure Biometric Templates from Fingerprint-Face Features," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR '07)*, 2007, pp. 1-6.
[190] Y. Sutcu, *et al.*, "Protecting Biometric Templates With Sketch: Theory and Practice," *IEEE Transactions on Information Forensics and Security,* vol. 2, pp. 503-512, 2007.
[191] X. Zhou, "Template Protection and its Implementation in 3D Face Recognition Systems," in *Biometric Technology for Human Identification IV*, Orlando, FL, USA, 2007, pp. 65390L-8.
[192] P. T. Tuyls, *et al.*, "Privacy-protected biometric templates: acoustic ear identification," in *Biometric Technology for Human Identification*, Orlando, FL, USA, 2004, pp. 176-182.
[193] X. Boyen, "Reusable cryptographic fuzzy extractors," in *Proceedings of the 11th ACM Conference on Computer and Communications Security*, Washington DC, USA, 2004, pp. 82-91.
[194] K. Simoens, *et al.*, "Privacy Weaknesses in Biometric Sketches," in *30th IEEE Symposium on Security and Privacy*, 2009, pp. 188-203.
[195] O. Song, *et al.*, "Application-specific key release scheme from biometrics," *International Journal of Network Security,* vol. 6, pp. 127-133, 2008.
[196] Y. C. Feng, *et al.*, "A hybrid approach for face template protection," in *Biometric Technology for Human Identification V*, Orlando, FL, USA, 2008, pp. 694408-11.

[197] J. Bringer, *et al.*, "The best of both worlds: Applying secure sketches to cancelable biometrics," *Science of Computer Programming,* vol. 74, pp. 43-51, 2008.
[198] N. Lalithamani and K. Soman, "An Effective Scheme for Generating Irrevocable Cryptographic Key from Cancelable Fingerprint Templates," *International Journal of Computer Science and Network Security,* vol. 9, p. 183, 2009.
[199] N. Lalithamani and K. P. Soman, "An Efficient Approach for Non-Invertible Cryptographic Key Generation from Cancelable Fingerprint Biometrics," in *International Conference on Advances in Recent Technologies in Communication and Computing (ARTCom '09)*, 2009, pp. 47-52.
[200] N. Lalithamani and K. P. Soman, "Irrevocable cryptographic key generation from cancelable fingerprint templates: An enhanced and effective scheme," *European Journal of Scientific Research,* vol. 31, pp. 372-387, 2009.
[201] T. E. Boult, *et al.*, "Revocable Fingerprint Biotokens: Accuracy and Security Analysis," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR '07)*, 2007, pp. 1-8.
[202] N. Ratha, *et al.*, "Privacy Enhancements for Inexact Biometric Templates," in *Security with Noisy Data*, P. Tuyls, *et al.*, Eds., ed: Springer London, 2007, pp. 153-168.
[203] Y. Bian, *et al.*, "Dynamic random projection for biometric template protection," in *Fourth IEEE International Conference on Biometrics: Theory Applications and Systems (BTAS)*, 2010, pp. 1-7.
[204] B. Yang and C. Busch, "Keyed Scalable Minutiae Coding," in *ICHB 2011: International Conference on Hand-Based Biometrics* 2011, pp. 1-5.
[205] A. M. P. Canuto, *et al.*, "Enhancing Performance of Cancellable Fingerprint Biometrics Using Classifier Ensembles," in *Eleventh Brazilian Symposium on Neural Networks (SBRN)*, 2010, pp. 55-60.
[206] Y. Bian, *et al.*, "Renewable Minutiae Templates with Tunable Size and Security," in *20th International Conference on Pattern Recognition (ICPR)*, 2010, pp. 878-881.
[207] R. Cappelli, *et al.*, "Minutia Cylinder-Code: A New Representation and Matching Technique for Fingerprint Recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 32, pp. 2128-2141, 2010.
[208] S. Wang and J. Hu, "Alignment-free cancelable fingerprint template design: A densely infinite-to-one mapping (DITOM) approach," *Pattern Recognition,* 2012.
[209] C. Li and J. Hu, "Attacks via record multiplicity on cancelable biometrics templates," *Concurrency and Computation: Practice and Experience,* 2013.
[210] Futronic. (2013, 2 October 2013). *FS88 FIPS201/PIV Compliant USB2.0 Fingerprint Scanner*. Available: http://www.futronic-tech.com/product_fs88.html
[211] Neurotechnology. (2011, 18 April 2012). *VeriFinger SDK*. Available: http://www.neurotechnology.com/verifinger.html
[212] D. Maltoni, *et al.*, "Fingerprint Matching," in *Handbook of Fingerprint Recognition*, Second ed New York: Springer-Verlag, 2009.
[213] Biometric System Laboratory. (2013, 7 October 2013). *Databases*. Available: http://biolab.csr.unibo.it/databasesoftware.asp
[214] D. Maio, *et al.*, "FVC2002: Second Fingerprint Verification Competition," in *Proceedings of the 16th International Conference on Pattern Recognition*, 2002, pp. 811-814.
[215] Biometric System Laboratory. (2006, 2 December 2013). *FVC 2006 Fingerprint Verification Competition*. Available: http://bias.csr.unibo.it/fvc2006/databases.asp
[216] A. K. Jain, *et al.*, "Fingerprint Recognition," in *Introduction to Biometrics*, ed: Springer US, 2011, pp. 51-96.
[217] Neurotechnology. (2013, 25 October 2013). *VeriFinger SDK*. Available: http://www.neurotechnology.com/verifinger.html
[218] N. K. Ratha, *et al.*, "A real-time matching system for large fingerprint databases," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 18, pp. 799-813, 1996.
[219] BioLab - University of Bologna. (2002, 8 August 2014). *FVC2002 Fingerprint Verification Competition - Download Page*. Available: http://bias.csr.unibo.it/fvc2002/download.asp
[220] BioLab - University of Bologna. (2002, 8 August 2014). *FVC2002 Fingerprint Verification Competition - Databases*. Available: http://bias.csr.unibo.it/fvc2002/databases.asp
[221] S. Wang and J. Hu, "Alignment-free cancelable fingerprint template design: A densely infinite-to-one mapping (DITOM) approach," *Pattern Recognition,* vol. 45, pp. 4129-4137, 2012.
[222] Markets and Markets. (2011, 10 February 2011). Global Biometrics Technology Market worth $11,229.3 million in 2015. Available: http://www.marketsandmarkets.com/Market-Reports/biometric-market-278.html
[223] J. Liu, *et al.*, "Fingerprint comparison. II: On the development of a single fingerprint filing and searching system," *Journal of Forensic Sciences,* vol. 27, pp. 305-317, 1982.
[224] C.-H. Park, *et al.*, "Fingerprint Matching Using the Distribution of the Pairwise Distances Between Minutiae," in *AVBPA '05*, Hilton Rye Town, NY, 2005, pp. 693-701.

[225] M. Tico and P. Kuosmanen, "Fingerprint matching using an orientation-based minutia descriptor," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 25, pp. 1009-1014, 2003.
[226] A. Wahab, *et al.*, "Novel approach to automated fingerprint recognition," *IEE Proceedings - Vision Image and Signal Processing,* vol. 145, pp. 160-166, 1998.
[227] J. Xudong and Y. Wei-Yun, "Fingerprint minutiae matching based on the local and global structures," in *ICPR 2000: Proceedings of the 15th International Conference on Pattern Recognition*, Barcelona, Spain, 2000, pp. 1038-1041.
[228] J. Qi, *et al.*, "Fingerprint matching combining the global orientation field with minutia," *Pattern Recognition Letters,* vol. 26, pp. 2424-2430, 2005.
[229] A. Jain, *et al.*, "On-line fingerprint verification," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 19, pp. 302-314, 1997.
[230] C. Fanglin, *et al.*, "Reconstructing Orientation Field From Fingerprint Minutiae to Improve Minutiae-Matching Accuracy," *IEEE Transactions on Image Processing,* vol. 18, pp. 1665-1670, 2009.
[231] G. Jinwei, *et al.*, "Fingerprint recognition by combining global structure and local cues," *IEEE Transactions on Image Processing,* vol. 15, pp. 1952-1964, 2006.
[232] U. Uludag, *et al.*, "Fuzzy Vault for Fingerprints," in *AVBPA '05*, Hilton Rye Town, NY, 2005, pp. 310-319.
[233] Q. Feng, *et al.*, "Fingerprint-based key binding/recovering scheme based on fuzzy vault," *Journal of Electronics (China),* vol. 25, pp. 415-421, May 2008 2008.
[234] C. Örencik, *et al.*, "Securing fuzzy vault schemes through biometric hashing," *Turk J Electr Eng Co,* vol. 18, pp. 515-539, Jul. 2010 2010.
[235] J. Merkle, *et al.*, "Performance of the Fuzzy Vault for Multiple Fingerprints (Extended Version)," *Cornwell University Library* 2011.
[236] J. Merkle, *et al.*, "Security Capacity of the Fuzzy Fingerprint Vault," *International Journal On Advances in Security,* vol. 3, pp. 146-168, 2010.
[237] P. Li, *et al.*, "Security-Enhanced Fuzzy Fingerprint Vault Based on Minutiae's Local Ridge Information," in *ICB '09*, Alghero, Italy, 2009, pp. 930-939.
[238] A. Nagar, *et al.*, "A hybrid biometric cryptosystem for securing fingerprint minutiae templates," *Pattern Recognition Letters,* vol. 31, pp. 733-741, 2010.
[239] S. Lee, *et al.*, "Protecting Secret Keys with Fuzzy Fingerprint Vault Based on a 3D Geometric Hash Table," in *ICANNGA '07*, Warsaw, Poland, 2007, pp. 432-439.
[240] D. Moon, *et al.*, "Fingerprint Template Protection Using Fuzzy Vault," in *ICCSA '07*, Kuala Lumpur, Malaysia, 2007, pp. 1141-1151.
[241] L. Sungju, *et al.*, "Memory-Efficient Fuzzy Fingerprint Vault based on the Geometric Hashing," in *ISA '08*, Busan, Korea, 2008, pp. 312-315.
[242] S. Lee, *et al.*, "Secure fuzzy fingerprint vault against correlation attack," *IEICE Electronics Express,* vol. 6, pp. 1368-1373, Sep. 2009 2009.
[243] D. Moon, *et al.*, "A practical implementation of fuzzy fingerprint vault for smart cards," *Journal of Intelligent Manufacturing,* pp. 1-10, May 2012 2012.
[244] A. A. Nasiri and M. Fathy, "Alignment-Free Fingerprint Cryptosystem Based On Multiple Fuzzy Vault and Minutia Local Structures," presented at the 5th SASTech, Mashhad, Iran, 2011.
[245] X. Kai and H. Jiankun, "Biometric Mobile Template Protection: A Composite Feature Based Fingerprint Fuzzy Vault," in *ICC '09*, Dresden, Germany, 2009, pp. 1-5.
[246] P. Li, *et al.*, "An alignment-free fingerprint cryptosystem based on fuzzy vault scheme," *Journal of Network and Computer Applications,* vol. 33, pp. 207-220, 2010.
[247] C. Örencik, *et al.*, "Improved fuzzy vault scheme for fingerprint verification," presented at the SECRYPT '08, Porto, Portugal, 2008.
[248] L. Jianjie, *et al.*, "Topological structure-based alignment for fingerprint Fuzzy Vault," in *ICPR '08*, Tampa, FL, 2008, pp. 1-4.