UNIVERSITY OF THE WESTERN CAPE

Independent hand-tracking from a single two-dimensional view and its application to South African sign language recognition

by

Imran Achmed

A thesis submitted in conformity with the requirements for the degree of Doctor of Philosophy

in the
Faculty of Science
Department of Computer Science

February 2014

Declaration of Authorship

I declare that *Independent hand-tracking from a single two-dimensional view and its application to South African sign language recognition* is my own work, that it has not been submitted for any degree or examination in any other university, and that all the sources I have used or quoted have been indicated and acknowledged by complete references.

Signed: ____

Date: ____

UNIVERSITY OF THE WESTERN CAPE

Abstract

Faculty of Science
Department of Computer Science

Doctor of Philosophy

by Imran Achmed
Supervisor: I. M. Venter
Co-supervisor: P. Eisert

Hand motion provides a natural way of interaction that allows humans to interact not only with the environment, but also with each other. The effectiveness and accuracy of hand-tracking is fundamental to the recognition of sign language. Any inconsistencies in hand-tracking result in a breakdown in sign language communication. Hands are articulated objects, which complicates the tracking thereof. In sign language communication the tracking of hands is often challenged by the occlusion of the other hand, other body parts and the environment in which they are being tracked. The thesis investigates whether a single framework can be developed to track the hands independently of an individual from a single 2D camera in constrained and unconstrained environments without the need for any special device. The framework consists of a three-phase strategy, namely, detection, tracking and learning phases. The detection phase validates whether the object being tracked is a hand, using extended local binary patterns and random forests. The tracking phase tracks the hands independently by extending a novel data-association technique. The learning phase exploits contextual features, using the scale-invariant features transform (SIFT) algorithm and the fast library for approximate nearest neighbours (FLANN) algorithm to assist tracking and the recovering of hands from any form of tracking failure. The framework was evaluated on South African sign language phrases that use a single hand, both hands without occlusion, and both hands with occlusion. These phrases were performed by 20 individuals in constrained and unconstrained environments. The experiments revealed that integrating all three phases to form a single framework is suitable for tracking hands in both constrained and unconstrained environments, where a high average accuracy of 82,08% and 79,83% was achieved respectively.

Acknowledgements

One of the joys of completing my thesis is looking back at the long journey that has passed, the uphill battles, the endless late nights, the frustration, the hard work and remembering all of those who supported and helped me along this long but fulfilling road.

First and foremost, I thank the Almighty God for blessing me throughout my studies and granting me the knowledge, wisdom and guidance to reach this level of success.

My heartfelt thanks and appreciations are conveyed to my supervisors, Prof. Isabella M. Venter and Prof. Peter Eisert. I feel fortunate and priviledged to have been guided by these remarkable academics. I value their insights, unparalleled knowledge, and the valuable time they spent guiding me in my research. Without their constructive critiques and advice, this thesis would not have been the same.

Throughout my research, Mr. Dodds contributed much of his time proof-reading my articles and providing me with sound advice and constructive comments that was of tremendous benefit. An enormous thanks goes to him for his generous assistance.

I am sincerely grateful and would like to thank Prof. Richard Madsen, Professor in Statistics, for his willingness to assist and provide insightful guidance on my statistical analysis from the research planning stages until the very end. His statistical expertise, especially in this thesis, is greatly appreciated.

I would also like to thank Prof. Alex Potter, Professor in English, for professionally editing the thesis and for his thoughtful comments and suggestions.

Many thanks to the Computer Science Department staff, especially, Ms. Abbott, Mrs. Jacobs-Samsodien and Mr. Leenderts for their selfless assistance in seeing to my (at times urgent) needs.

A word of thanks to my sponsor, Telkom, for their financial support throughout the entire duration of my studies. Without their support, it would not have been possible to do this research. A further thanks is extended to all the individuals who volunteered and offered their time to participate in this research.

To my lab colleagues, Pei-Li, Dane, Diego, Warren, Ibraheem, Kenzo and Roland, thank you for providing a stimulating environment to work in. I had a great time working alongside you all, sharing ideas and the many laughters we had. It was truly enjoyable.

I am forever indebted to my parents who instilled in me the values of hard work and dedication. Even though they did not complete their school education, their enormous support in the furthering of my studies has pushed me to excel. It was always difficult for them to understand why I wanted to pursue my Ph.D., but my passion for research and seeing the research transform into a thesis has made them realise its worth.

A special mention is given to my friends and family who have supported me through my studies; Arnold, Andrew, Ashley, Fozia, Chiara, Riedewaan, Mrs. Yvonne Meyer (Ma), Toshca, Jody, Latasha, Isaac, Lindy and Jason, thank you for your genuine concern in my well-being and always motivating me to succeed.

Above all, I would like to convey a special thanks to the most important person in my life, my fiancee, and soon to be wife, Tamlynne Meyer. Her love, support, patience and understanding were just a few of her qualities that kept me sane throughout my studies. There were times when I was ready to throw in the towel, when no-one understood what I was going through but she was always beside me, listening to me and encouraging me. She kept me motivated and gave me the momentum to see through each year. No-one will understand the extent to which she has supported me, not only through my Ph.D., but throughout my studies since undergraduate level. To her I owe everything. There are no words to convey how much I love her. She is my source of inspiration.

Tamlynne, to you I dedicate this thesis.

Contents

Declaration of Authorship i

Abstract ii

Acknowledgements iii

List of Figures ix

List of Tables xiii

Glossary xv

1 Introduction 1
- 1.1 Background and motivation . . . 1
- 1.2 Research problem . . . 2
- 1.3 Research questions . . . 5
- 1.4 Research objectives . . . 5
- 1.5 Premises . . . 6
- 1.6 Contributions . . . 7
- 1.7 Thesis outline . . . 7

2 Related Work 9
- 2.1 Hand detection . . . 10
  - 2.1.1 Three-dimensional hand detection . . . 10
  - 2.1.2 Two-dimensional hand detection . . . 11
- 2.2 Context-based object tracking . . . 14
  - 2.2.1 Knowledge-based context . . . 14
  - 2.2.2 Temporal context . . . 15
  - 2.2.3 Spatial context . . . 16
- 2.3 Hand-tracking . . . 17
  - 2.3.1 Auxiliary approaches . . . 17
    - 2.3.1.1 Two-dimensional input . . . 18
    - 2.3.1.2 Three-dimensional input . . . 19
  - 2.3.2 Passive approaches . . . 21
    - 2.3.2.1 Model-based methods . . . 21
    - 2.3.2.2 Appearance-based methods . . . 25

2.3.2.2.1 Three-dimensional input . . . 26
2.3.2.2.2 Two-dimensional input . . . 27
2.4 Discussion and summary . . . 32

3 Design Science Research **35**
3.1 Research philosophy . . . 35
3.2 Epistemology . . . 37
3.3 Theoretical perspective . . . 38
3.4 Methodology . . . 39
3.4.1 Design science research . . . 39
3.5 Methods . . . 42
3.6 Summary . . . 43

4 Detection-Tracking-Learning **44**
4.1 Detection Phase . . . 44
4.1.1 Local binary patterns . . . 44
4.1.1.1 Uniform LBP . . . 47
4.1.1.2 Rotational-invariant LBP . . . 48
4.1.1.3 Global versus local LBP histograms . . . 48
4.1.2 Random forests . . . 49
4.1.2.1 Error, strength and correlation using out-of-bag-error . . . 52
4.1.2.2 Variable importance . . . 53
4.1.2.3 Testing . . . 53
4.1.2.4 Parameter selection . . . 55
4.1.2.5 Strengths and weaknesses of random forests . . . 55
4.1.3 Support vector machines . . . 56
4.1.3.1 Kernel functions . . . 61
4.2 Tracking phase . . . 62
4.2.1 Skin segmentation . . . 63
4.2.1.1 Face detection . . . 64
4.2.1.2 Does a suitable colour space exist? . . . 65
4.2.1.3 Dynamic skin model . . . 66
4.2.2 Connected-components labelling . . . 67
4.2.3 Independent hand-tracking . . . 68
4.2.3.1 Generating object hypotheses . . . 69
4.2.3.2 Tracking-object hypotheses . . . 71
4.2.3.3 Predicting of object hypotheses . . . 72
4.3 Learning phase . . . 73
4.3.1 Feature detectors . . . 74
4.3.1.1 Scale-invariant feature transform . . . 74
4.3.2 Feature descriptors . . . 75
4.3.2.1 Scale-invariant feature transform descriptor . . . 75
4.3.3 Image matching . . . 76
4.3.3.1 FLANN-based matcher . . . 76
4.3.4 Comparison of feature detectors and descriptors . . . 76
4.4 Summary . . . 78

5 Hand-Tracking Framework **80**
5.1 Prototype T . . . 80
5.2 Prototype DT . . . 86
5.3 Prototype DTL . . . 90
5.4 Summary . . . 95

6 Experimental Results and Analysis **96**
6.1 Introduction . . . 96
6.2 Dataset generation . . . 97
6.2.1 Sign language data . . . 98
6.2.2 SASL dataset . . . 98
6.2.3 Hand dataset . . . 99
6.3 Experimental setup . . . 102
6.4 Ground-truth annotation . . . 102
6.5 Performance evaluation metrics . . . 104
6.5.1 Euclidean distance . . . 104
6.5.2 Tracking ratio . . . 106
6.5.3 Accuracy, precision and recall . . . 106
6.6 Results and discussion . . . 107
6.6.1 Tracking evaluation . . . 108
6.6.1.1 Prototype T . . . 108
6.6.2 Detection and tracking evaluation . . . 112
6.6.2.1 Hand detection . . . 112
6.6.2.1.1 Random forests . . . 113
6.6.2.1.2 Support vector machines . . . 116
6.6.2.1.3 Comparison between hand-detection systems . . 119
6.6.2.2 Prototype DT . . . 122
6.6.3 Detection, tracking and learning evaluation . . . 126
6.6.3.1 Prototype DTL . . . 126
6.6.4 Comparing prototypes T, DT and DTL . . . 130
6.6.4.1 Statistical analysis of the comparison between prototypes T, DT and DTL . . . 133
6.7 Summary . . . 140

7 Conclusion and Directions for Future Research **141**
7.1 Contributions . . . 142
7.1.1 Articulated hand detection using extended local binary patterns with random forests . . . 143
7.1.2 Pixel-level data association for independent hand-tracking in unconstrained environments . . . 144
7.1.3 Exploiting contextual features towards assisting hand-tracking . . 145
7.1.4 A framework for independent hand-tracking . . . 146
7.2 Limitations of the research study . . . 146
7.3 Future work . . . 147
7.3.1 Arm-based hand-tracking . . . 148
7.3.2 Enhancing skin-colour segmentation . . . 148
7.3.3 Improved hand detection . . . 149

7.3.4 Optimised hand-tracking framework 150
7.4 Reflecting on the study 150
7.4.1 Current interest of the study 151
7.4.2 Significance of the study 151
7.4.3 Relation to previous studies 151

A State-of-the-art feature detectors and descriptors 153
A.1 Feature detectors 153
A.1.0.1 Harris 153
A.1.0.2 Good-features-to-track 154
A.1.0.3 Feature from accelerated segment test 154
A.1.0.4 Speeded up robust feature 155
A.1.0.5 STAR 156
A.1.0.6 Maximally stable extremal regions 156
A.2 Feature descriptors 157
A.2.0.7 Speeded up robust feature descriptor 157
A.2.0.8 Binary robust independent elementary features descriptor 157
A.2.0.9 Oriented FAST and rotated BRIEF descriptor 158
A.2.0.10 Binary robust scalable keypoints descriptor 158
A.2.0.11 Fast retina keypoint descriptor 158
A.3 Image matching 159
A.3.0.12 Brute-force matcher 159

B Consent form 160

C Dataset compilation 161

D Additional results 169
D.1 Tracking 170
D.2 Detection 173
D.3 Learning 178

E Publications 190

Bibliography 192

List of Figures

1.1 The South African sign language translation system. 3
1.2 The human-computer interaction gesture taxonomy [33]. 3

2.1 Using depth data (bottom left and bottom middle) and skin-colour information (top middle and top right) to find a hand (bottom right) [156] (p. 68). 11
2.2 Skin-coloured distribution and geometric constraints were used to determine the position of a hand [125] (p. 112). 11
2.3 The red box (top left of image) represents the ROI for a HOG patch and each circle represents the center of the ROI for each HOG patch [151] (p. 92). 12
2.4 Hand detection applied to three simple hand postures [166] (p. 286). . . . 12
2.5 (a) The original image; (b) the selected ROI used for training; (c) the classification of AdaBoost using a single positive sample only; (d) the output of AdaBoost using positive and negative samples [116]. 13
2.6 Tracking multiple targets that exhibit a common motion pattern [89] (p. 812). 15
2.7 A global temporal context constraint applied to tracking [163] (p. 719). . 15
2.8 Each feature point is classified as either object, background and companion model [25] (p. 2127). 16
2.9 The yellow box represents the target. The green, blue and red dots represents the features that belong to the object itself, the uncorrelated features that are discarded and the features that are used to vote for the position of the object, respectively [64] (p. 1286). 17
2.10 The gloves were specially designed using colour patches to track the fingers and hand while dealing with occlusion [161] (p. 63). 18
2.11 The Kinect sensor is used to track the main points on hands [55] (p. 319). 19
2.12 The images correspond to the (a) original image; (b) depth image; (c) segmented area; and (d) the output image [167] (p. 151). 19
2.13 The output after using the depth threshold to find the hand and the Kalman filter to track the hand [124]. 20
2.14 The depth data are used to track the hands, where the grey and white clusters represent the right and left hand, respectively [27]. 20
2.15 A two-step minimisation algorithm was applied to match the model to a hand [70]. 22
2.16 Hierarchical detection was employed to find the hand and a dynamic model guided the search while approximating the optimal filtering equations [146] (p. 1381). 23

2.17 The rows correspond to the original image, the final synthetic image, the final residual image and the side view of the synthetic image [42]. 24
2.18 The left and right camera views of the Bumblebee stereo camera system were used to determine the depth correspondences [49] (p. 26). 26
2.19 The output of the tracking system using skin-colour detection only [128] (p. 31). 28
2.20 The modified flocks-of-features algorithm still fails due to rapid movements [54]. 29
2.21 A sample output of the improved flocks-of-features algorithm [97] (p. 466). 29
2.22 An example of the output obtained from framework used by Spruyt et al. [144] (p. 3120). 30
2.23 The tracking of both hands, but not distinguishing between them using the framework of Ongkittikul et al. [120] (p. 205). 31
2.24 The output of the framework introduced by Argyros and Lourakis [10] (p. 1190). 32

3.1 The decision-making process as four elements that inform one another . . 36
3.2 An illustration of the link between epistemology and the research process. 37
3.3 The iterative process of a general DSR cycle consists of six distinctive stages [159]. 40
3.4 The DSR cycle applied to each of the phases. 41
3.5 A taxonomy of objective evaluation methods for tracking algorithms. . . . 42

4.1 The first DSR cycle consisting of the detection phase. 45
4.2 An example of the LBP process that determines the decimal-label value. . 45
4.3 Optional caption for list of figures 46
4.4 Examples of the different micro-feature information that are extracted using LBPs. 47
4.5 An example of the concatenation of a spatially enhanced feature histogram. 49
4.6 Optional caption for list of figures 56
4.7 Linear classification of a hyperplane [170]. 57
4.8 The second DSR cycle, consisting of the tracking phase. 62
4.9 The probability map where lighter pixels have higher probabilities and darker pixels have lower probabilities. 67
4.10 Optional caption for list of figures 72
4.11 The third DSR cycle, consisting of the learning phase. 74

5.1 An overview of the systematic design of prototype T. 82
5.2 Optional caption for list of figures 83
5.3 Optional caption for list of figures 83
5.4 Optional caption for list of figures 84
5.5 Association of the right and left hand in the initial frames. 84
5.6 An example of the updating process of the foreground-skin image. 85
5.7 An example of the convex hull of a hand. 86
5.8 An overview of the systematic design of prototype DT. 87
5.9 Optional caption for list of figures 88
5.10 An example of the global representation of a LBP image in the form of a single histogram. 89

5.11 An example of a feature vector directly formed from the labelled pixel values of a LBP image. 89
5.12 An overview of the systematic design of prototype DTL. 92
5.13 The bounding region of the hand used to extract contextual feature information. 93
5.14 The features for each hand are stored in two separate databases. 93
5.15 The voting strategy used in the learning phase, where the cluster with the highest number of votes is selected. 94

6.1 The South African sign language translation system. 97
6.2 Optional caption for list of figures 99
6.3 Optional caption for list of figures 100
6.4 Optional caption for list of figures 101
6.5 Optional caption for list of figures 101
6.6 A frame in which the right hand is occluded by the participant's body. . . 103
6.7 A frame in which the ground-truth coordinates have been verified where the blue circle represents the right hand and the red circle represents the left hand. 104
6.8 An illustration of a radius of 10 pixels (red circle), 15 pixels (green circle) and 20 pixels (orange pixels) from the center of the hand. 105
6.9 The average hand-tracking success rates of prototype T for each signed gesture in both environments. 109
6.10 A comparison of the average hand-tracking success rates of prototype T for each signed gesture in constrained and unconstrained environments. . 111
6.11 Evaluating different random forest parameters. 114
6.12 Evaluating different LBP algorithms on the original images in the dataset (experiment 1) and compared to the same images that were filtered based on skin-coloured pixels only (experiment 2), using a random forest model. 115
6.13 Evaluating and comparing different SVM kernels. 117
6.14 Evaluating different LBP algorithms on the original images in the dataset (experiment 3) and compared to the same images that were filtered based on skin-coloured pixels only (experiment 4), using an SVM model. 118
6.15 The average hand-tracking success rates of prototype DT for each signed gesture in both environments. 123
6.16 A comparison of the average hand-tracking success rates of prototype DT for each signed gesture in constrained and unconstrained environments. . 125
6.17 The average hand-tracking success rates of prototype DTL for each signed gesture in both environments. 127
6.18 A comparison of the average hand-tracking success rates of prototype DTL for each signed gesture in constrained and unconstrained environments.129
6.19 A comparison between the hand-tracking accuracies of each prototype. . . 131

A.1 Optional caption for list of figures 156

D.1 The average hand-tracking success rates of prototype T for each participant in both environments. 170
D.2 A comparison of the average hand-tracking success rates of prototype T for each participant in constrained and unconstrained environments. . . . 170

D.3 The average hand-tracking success rates of prototype DT for each participant in both environments. . . . 174
D.4 A comparison of the average hand-tracking success rates of prototype DT for each participant in constrained and unconstrained environments. . . . 175
D.5 The average hand-tracking success rates of prototype DTL for each participant in both environments. . . . 178
D.6 A comparison of the average hand-tracking success rates of prototype DTL for each participant in constrained and unconstrained environments. 178
D.7 A comparison between the hand-tracking accuracies of each prototype in the constrained environments. . . . 181
D.8 A comparison between the hand-tracking accuracies of each prototype in the unconstrained environments. . . . 182

List of Tables

6.1 The hand dataset setup . . . 102
6.2 The confusion matrix of a binary classification problem . . . 107
6.3 Summary of the average hand-tracking success rates of prototype T for each signed gesture in both environments. . . . 108
6.4 Comparison between different hand-detection methods considering all four factors. . . . 120
6.5 Summary of the average hand-tracking success rates of prototype DT for each signed gesture in both environments. . . . 124
6.6 Summary of the average hand-tracking success rates of prototype DTL for each signed gesture in both environments. . . . 128
6.7 Analysis on sign 17 representing an interaction between the prototype and the environment. . . . 134
6.8 Analysis between prototypes in which the results were statistically significant with an adjusted p-value less than 0.05. . . . 135
6.9 Evaluation to determine if the results are statistically significant between the environment and the sign being performed. . . . 138
6.10 Analysis to determine if there are any significant differences between the signs being performed on prototype DTL in a constrained environment. . 139

C.1 Group 1: These sign gestures contain single hand movements only. . . . 161
C.1 Group 1: These sign gestures contain single hand movements only. . . . 162
C.2 Group 2: These sign gestures contain the movement of both hands without any interaction or occlusion. . . . 163
C.2 Group 2: These sign gestures contain the movement of both hands without any interaction or occlusion. . . . 164
C.2 Group 2: These sign gestures contain the movement of both hands without any interaction or occlusion. . . . 165
C.3 Group 3: These sign gestures contain the movement of both hands that includes the interaction and occlusion between hands. . . . 165
C.3 Group 3: These sign gestures contain the movement of both hands that includes the interaction and occlusion between hands. . . . 166
C.3 Group 3: These sign gestures contain the movement of both hands that includes the interaction and occlusion between hands. . . . 167
C.4 The selected groups of signs used to evaluate the prototypes. . . . 168

D.1 Evaluation results using a threshold of 10, 15 and 20 pixels from the center of the hand. . . . 169
D.2 Compare sign gestures using prototype T in constrained and unconstrained environments. . . . 171

D.3 Compare participants using prototype T in constrained and unconstrained environments. 172
D.4 Results on diffeernt random forest parameters using the original LBP with global histogram features. 173
D.5 Results on random forests using different LBP algorithms on pixels with and without filtering. 173
D.6 Results on different SVM kernel parameters using the original LBP with global histogram features. 174
D.7 Results on SVMs using different LBP algorithms on pixels with and without filtering. 174
D.8 Compare sign gestures using prototype DT in constrained and unconstrained environments. 176
D.9 Compare participants using prototype DT in constrained and unconstrained environments. 177
D.10 Compare sign gestures using prototype DTL in constrained and unconstrained environments. 179
D.11 Compare participants using prototype DTL in constrained and unconstrained environments. 180
D.12 A list of the non-significant differences between the results of the different hand detection methods. The difference between the results for the methods not listed in the table are all significant (This list was excluded as it a much longer list). In the table, the combination and _combination column values can be referred to in Table 6.4. 183

Glossary

2D	-	**Two-dimensional space** - A space in the physical world, *e.g.* an image, that has two dimensions (width and height).
3D	-	**Three-dimensional space** - A space in the physical world that has three dimensions (width, height and depth).
ASL	-	**American Sign Language** - The form of sign language developed by the Deaf and hard-of-hearing individuals in the United States of America.
DSR	-	**Design science research** - A methodological framework that is used to develop a solution and provide general knowledge that can later be used to design solutions to specific problems by professionals in a respective field.
ETH	-	**ETH** - ETH Zurich, Department of Information Technology and Electrical Engineering video sequence.
ELBP	-	**Extended local binary pattern** - A generic form of LBP based on circular neighbourhoods and linearly interpolating the pixel values that is not restricted by the size of the neighbourhood or the number of sampling points.
FLANN	-	**Fast library for approximate nearest neighbours** - A library provides an efficient search method that is used to find correspondences between feature vector sets.
GPU	-	**Graphics processing unit** - An electronic circuit that is specially designed to accelerate computations for image processing.
Haar	-	**Haar features** - A set of features that originated from Haar wavelets.
HCI	-	**Human Computer Interaction** - It includes the planning, design and analysis of the interaction between people and computers
HMM	-	**Hidden Markov Model** - A statistical Markov model that assumes the system to be modelled to be a Markov process with hidden states.
HOG	-	**Histogram of Oriented Gradients** - Feature descriptors that count the occurrences of gradient orientations in local regions of an image.
HSV	-	**Hue, saturation and value** - An alternative colour space that is computed from the red-green-blue colour space and is based on the human colour perception.

LAB	-	**Luminance, red-green and blue-yellow colour space** - L represents luminance, A represents the red-green colour components and B represents the blue-yellow colour components. The LAB colour space is a colour model that is based on the nonlinearly compressed CIE-XYZ colour space coordinates.
LBP	-	**Local Binary Patterns** - Describes the neighbouring area of a pixel by generating a binary code from the binary derivatives of that pixel.
OOB	-	**Out-of-bag** - Bootstrap samples constructed for each tree by sub-sampling with replacement from the original training set.
OOBE	-	**Out-of-bag-error** - Error estimates of the tree and the forest that is based on the performance of the split.
RGB	-	**Red, Green and Blue** - The primary colour space, is defined by the three chromaticities of the red, green and blue primary colours.
SASL	-	**South African Sign Language** - The form of sign language developed by the Deaf and hard-of-hearing individuals in South Africa.
SIR	-	**Sequential Importance Resampling** - It is the recursive version of the importance sampling in particle filters
SiGML	-	**Signing Gesture Markup Language** - The Signing Gesture Markup Language is based on HamNoSys.
SURF	-	**Speeded Up Robust Features** - Is a performance scale-invariant and rotation-invariant interest point detector and descriptor.
SVM	-	**Support Vector Machine** - Is a set of supervised learning algorithms, used for classification and regression analysis, which analyses data and recognises patterns.
ToF	-	**Time-of-flight** - Represents the time for an acoustic wave to travel between two points from the camera to an object.
TSL	-	**Taiwanese Sign Language** - The form of sign language commonly used by the Deaf and hard-of-hearing individuals in Taiwan.

Chapter 1

Introduction

1.1 Background and motivation

The symbolic value of sign language remains at the heart of the identity of the hearing impaired. Sign language is used as a primary means of communication by Deaf[1] people and people that are hard of hearing all over the world, whereas oral communication is inherently used by the hearing community. Over the past decade mobile communication has grown exponentially to allow millions of people to communicate daily [7], improving social interaction, educational services and socio-economic opportunities to a great extent. Sadly, the hearing impaired have not been able to fully reap the benefits of this rich form of information exchange and social communication, thus marginalising them from the broader society [2].

Despite this, the Constitution of the Republic of South Africa recognises South African Sign Language (SASL) as the official language for Deaf South African communities [43]. The communication problem experienced by the Deaf is partially alleviated by using skilled interpreters; however, their services need to be arranged ahead of time, they are expensive and there is unfortunately a lack of skilled interpreters to assist the hearing impaired in South Africa [61][63]. In cases where privacy is required, such as medical consultations, a Deaf person may not be comfortable having an interpreter present.

[1]Deaf with a capital D refers to people that use SASL as their primary language, while deaf with a lower case d refers to people that are hard of hearing.

To bridge this communication gap, an automated mobile translation system that translates from sign language to a spoken language and vice versa would benefit the Deaf community immensely. It would, in effect, improve communication between the hearing and Deaf communities. It would also be a solution to the shortage of skilled interpreters and the privacy issue. Such a system is complex and encompasses a multi-disciplinary research area that involves image processing, natural language processing, linguistics and artificial intelligence.

This research forms part of a broader research project at the University of the Western Cape that deals with the development of *an integrated sign and verbal communication mobile translation system*, depicted in Figure 1.1. One aspect of the translation system is concerned with recognising SASL, which is challenging due to the complexities involved in the visual interpretation of signed gestures. These gestures are collectively represented by facial expressions, hand trajectory, hand location and hand shapes, each of which forms a different component. This research focuses on two of these components, namely hand trajectory and hand location, which is a subdivision of hand-tracking. Fundamental to the recognition of sign language is the effectiveness and accuracy of hand-tracking. Any failure in hand-tracking would result in a breakdown in communication. This can be compared to voice communication where a breakdown results in the message not being correctly conveyed. According to the gesture taxonomy [33], depicted in Figure 1.2, hand movements form the basis of human–computer interaction. They provide a natural way of interaction that allows us to interact not only with the environment, but with each other. This illustrates the importance of tracking hands accurately.

1.2 Research problem

One of the principal aims of the SASL project is to implement the entire translation system as a service on a mobile phone. According to a study by Ghaziasgar [60], mobile devices are well suited for capturing audio and video; however, the video material captured by most mobile devices is two-dimensional (2D). The aim is to allow users to use the system freely in an unconstrained environment.

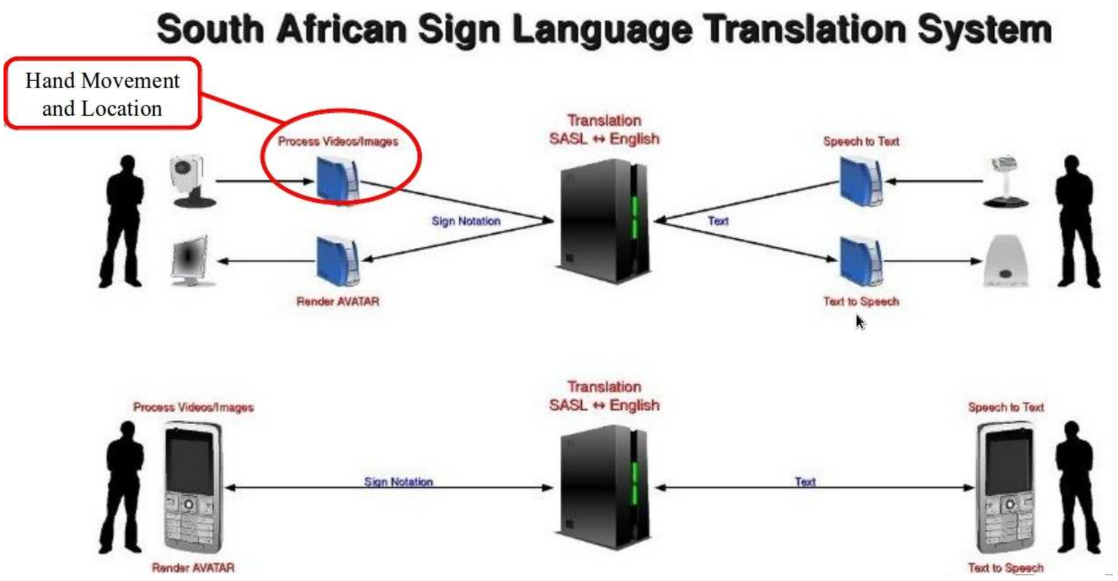


FIGURE 1.1: The South African sign language translation system.

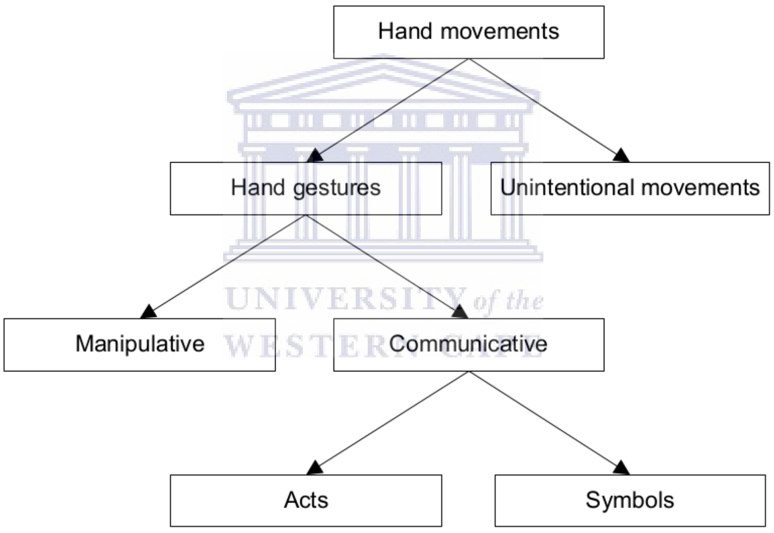


FIGURE 1.2: The human-computer interaction gesture taxonomy [33].

Two-dimensional video data captured in unconstrained environments contain much less information than three-dimensional (3D) video data [142]. This is a major drawback in extracting information accurately for sign language translation, since depth cannot be used to distinguish between the hands or between the hands and the face. Naturally, the hands form a major part of transferring information among the hearing impaired in their everyday lives and should not be limited to a specific environment.

The problem of hand-tracking has been explored for more than a decade [103, 107, 173] and continues to attract research interest, not only for sign language applications, but also for human–computer interfaces, human–robot interactions, and a number of applications in the recent and developing advances in gaming, such as the Sony Move, Nintendo Wii and Microsoft Kinect, each with their individual set of assumptions [111].

To date, hand-tracking from a single 2D view is still a largely unsolved problem [23], especially in unconstrained environments, and therefore an active research area [111]. Hand-tracking is a broad term that is used when determining the hand trajectory and hand location. Before higher-level information can be extracted and used for further tasks, the hands need to be localised in an environment. The position of the hands throughout an image sequence should therefore be found accurately and reliably.

The importance of hand-tracking and finding the location of the hands is demonstrated in other components within the translation system, such as hand-shape recognition, which includes tracking the individual fingers. Hand-tracking therefore serves as a first step in hand-shape recognition for SASL, where detailed information on the hand configuration can be extracted. In addition, it serves to indicate when the face is occluded by the hand(s), a scenario that requires information from hand location, hand-shape recognition and facial expression recognition.

The difficulties of hand-tracking arise primarily from the high dimensionality of the hand configuration space, high appearance variation, self-occlusions of the body parts, motion blur caused by the speed of signing, camera characteristics and complex backgrounds. Hands are articulated objects, which therefore complicates tracking as a result of the high dimensionality of possible space configurations. With respect to the appearance, the variation between one person and another may be significant, e.g. the skin may vary from a very light to a very dark skin-colour tone, and the geometry of the hands may vary in terms of thickness, length and width.

In sign language gestures, as well as other forms of hand motions, the hands are often occluded by a different body part or the opposite hand. In addition, the speed at which the sign or gesture is performed is often accompanied by motion blur and is different for each person. Some may move their hands slowly, while others move them faster.

One of the main challenges confronting hand-tracking and the tracking of other objects is the environment in which it is being tracked. An environment that contains too much or too little light, too many objects (especially objects that are similar), and too much background movement may negatively affect the tracking capabilities. While many researchers rule out these challenges by making several assumptions, this research directly addresses such challenges in a single framework.

1.3 Research questions

The hypothesis driving this research is: *A single framework can be developed to track the hands of an individual independently from a single 2D camera in constrained and unconstrained environments.* Thus, the question is: How should a framework be developed to detect, learn and track the hands? This research question can be translated into three phases:

1. How should hands be detected in an image sequence?
2. How should hands be tracked independently while effectively handling occlusion?
3. How should features be learnt to assist hand-tracking and avoid tracking failure?

1.4 Research objectives

These questions will be addressed by means of the development of a proof-of-concept framework. The framework will be developed around the same concept introduced by Kalal [81]. Kalal [81] introduced the tracking-learning-detection concept as a strategy to track single objects in any environment or condition. This concept is adapted in this research to track two similar objects, such as the hands, by applying the detection phase before the tracking phase in the following order detection-tracking-learning. Here *learning* is referred to as storing information to memory over time.

The same order is used to address the research questions as follows:

1. While tracking the hands, a detection phase should be implemented that validates that the object being tracked is a hand. To detect whether the object is a hand, a comparison should be made between different kind of local binary patterns (LBPs) to determine the best method for extracting hand features, as well as a comparison between them using different support vector kernels and random forests with different parameters to determine the best way to model these features.

2. In the tracking phase, the hands should be localised in the form of skin clusters and identified by applying the connected-components-labelling algorithm to a skin-detected image. This image should be generated dynamically using an individual's facial information to determine his or her skin colour and labelling an image according to skin pixels and non-skin pixels. To track the hands independently, a unique algorithm that uses data association should be applied to distinguish the right hand from the left and vice versa in an image sequence while dealing with occlusion.

3. The tracking of any object at a particular point in time is subjected to tracking failure. To recover from tracking failure, a learning phase should be implemented. This phase should continuously extract additional features from objects around the hands that could be used to recover the tracking process if tracking failure should occur. These features will be extracted using the scale-invariant features transform (SIFT) algorithm, then stored in a database and matched using the fast library for approximate nearest neighbours (FLANN) algorithm.

4. To effectively and accurately track the hands independently in constrained and unconstrained environments, the detection, tracking and learning phase should be integrated in a single framework.

1.5 Premises

Sign language communication does not involve the entire body and requires only the upper body to convey the message [51][135]. Therefore, only information from the upper body is used for SASL recognition. Furthermore, the focus is on sign language gestures using long-sleeved clothing, because this is more challenging. Recognising gestures with short-sleeved clothing is a simple task, because the skin of the arms can be used to find and distinguish between the hands.

In addition, the focus will exclude hand-shape recognition, which forms another component in the SASL translation system. The lack of hand configuration information imposes additional barriers on hand-tracking and distinguishing between hands.

1.6 Contributions

In addition to sign language recognition, the proposed contributions in this thesis have the potential to be applied to different problems such as human–computer and human–robotic interaction. The main contribution of this thesis is a novel solution to independently track the right and left hand of an individual in any type of environment from a single 2D camera within a framework.

Additional contributions of this framework include:

1. An algorithm that determines whether a region of interest is a hand or not using global local binary patterns to extract features from a region of interest that has been filtered using a novel skin detection algorithm and predicts the region using a random forest model;
2. A unique algorithm that distinctively applies data association to distinguish and track the right hand from the left and vice versa in both constrained and unconstrained environments from a single 2D camera. The algorithm is able to effectively handle occlusion in situations where the hands are stationary or moving; and
3. An algorithm that exploits contextual information from strong and reliable features on the hand and on objects in close proximity to the hand using SIFT. These features are used to recover from tracking failure and assist in distinguishing hands using a novel voting strategy.

1.7 Thesis outline

The thesis is laid out as follows:

Chapter 2: *Related work:* Existing literature that involve the hand detection, context-based tracking and hand-tracking algorithms will be reviewed. The algorithms of the various approaches that have significantly impacted the development of these research areas will be compared and their benefits and trade-offs highlighted.

Chapter 3: *Design science research:* The philosophical grounding that underpins the research will be discussed to ensure the consistency of the study. The methodology that is informed by the philosophical stance will be described and the methods selected to perform the data analysis will be motivated.

Chapter 4: *Detection-tracking-learning:* The chapter will explore the components that form part of the detection, tracking and learning phases of the proposed framework. The various algorithms included in the components of each phase will be compared and their applicability to addressing the research problems will be explained.

Chapter 5: *Hand-tracking framework:* The chapter will describe the systematic design and procedures that were implemented to investigate the research questions. It will explain the way in which the various algorithms were linked and integrated in a particular phase and collectively used to form the final framework.

Chapter 6: *Experimental results and analysis:* In this chapter the experimental setup used to evaluate the proposed framework will be discussed and its effectiveness in terms of hand-tracking in SASL will be analysed. The chapter will present the results of the experiments as well as the analysis on the effectiveness of the skin detection algorithm, the accuracy of hand detection in images and the accuracy of tracking both hands independently throughout an image sequence.

Chapter 7: *Conclusion and future work:* The final chapter will highlight the main contributions of the research, recommend directions for future work and reflect back on the study.

Chapter 2

Related Work

Traditionally, most hand-tracking systems depended on complex devices or markers that were pre-attached to an individual's body. These systems have several disadvantages: they are expensive, conspicuous and impractical in a day-to-day sign language translation application. Hand-tracking systems relying on image-processing and machine-learning techniques to provide markerless solutions and have increased as a focus of attention in the past decade. With the advances in image processing and machine learning, new efforts are constantly being made to create novel systems or improve on existing ones.

In this chapter, existing literature that relates to hand-tracking is discussed. It provides an overview of the approaches and aspects others have used in this field. Many contributions have been made, but only the most recent and interesting studies directly related to this research are discussed. These studies not only presents the latest developments, but also the current interest and relevance of this research area. Furthermore, focusing on the most recent studies allows one to identify existing research groups and potential contributions that can be made in the field.

To give an overview of these studies, the chapter is divided into three parts, relating to the detection, learning and tracking phases. Section 2.1 reviews approaches for the detection of hand instances in images. In Section 2.2, studies that have used context-based tracking to improve object-tracking methods are discussed. Section 2.3 gives an overview of the different hand-tracking techniques that exist, ranging from those using an auxiliary means to those using a purely passive means. The chapter concludes with a summary of each section, the challenges that still remain and suggestions as to what future directions could be investigated as contributions to a hand-tracking framework.

2.1 Hand detection

Hand detection is the process of determining the presence of a hand in an image. Several algorithms have been proposed, employing cues such as shape, colour, texture, context and depth. These cues have been successful in face-detection algorithms, since the appearance of faces has less variation, e.g. the nose and mouth can always be found below the eyes. However, hands are articulated objects and have a greater degree of freedom, resulting in a larger appearance variation. Identifying hands in images, despite the large variation, is the foremost challenge in hand-detection or -tracking tasks. This is further complicated by occlusion, complex environments and different illumination conditions. In this section algorithms proposed for hand detection in images have been grouped into 3D and 2D methods.

2.1.1 Three-dimensional hand detection

In 3D views the hand detection problem is simplified by the assumption that the hands will always be in front of the body. Van den Bergh and Van Gool [156] proposed to use depth from a time-of-flight (ToF) camera and colour from a red-green-blue (RGB) camera to detect the hands. Firstly, they calibrated the cameras by mapping the depth data to the colour data. They then used the distance of the face and skin colour to determine which objects are likely to be the hands. They therefore assumed that any skin-coloured object that has a shorter distance to the camera will be a hand, as shown in Figure 2.1. They evaluated their system on 362 frames and obtained a correct detection rate of 99.3%.

The same segmentation approach was taken by Ren et al. [130] and Hadfield and Bowden [68], who used the MS Kinect camera and stereo cameras, respectively, to extract depth. They followed the same assumption that any skin-coloured object that it is closer to the camera than the face can be assumed to be a hand.

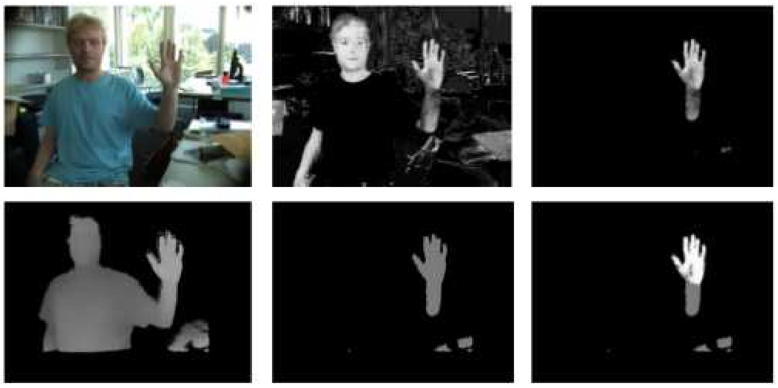


FIGURE 2.1: Using depth data (bottom left and bottom middle) and skin-colour information (top middle and top right) to find a hand (bottom right) [156] (p. 68).

2.1.2 Two-dimensional hand detection

In 2D views, the hand detection problem is categorised as the detection of articulated objects due to the large degrees of freedom of the hands. The most common cue used to detect hands are colour-based features.

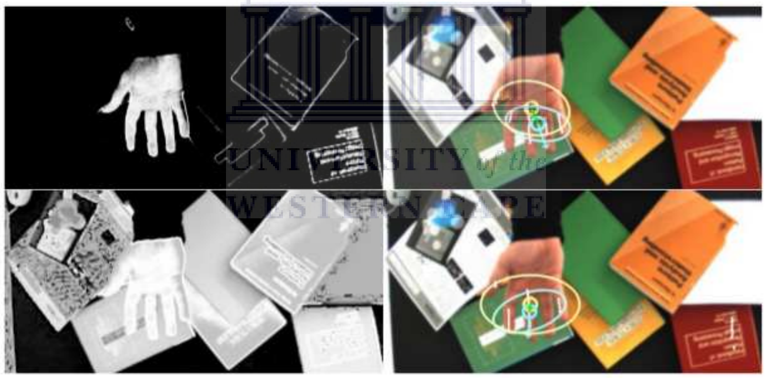


FIGURE 2.2: Skin-coloured distribution and geometric constraints were used to determine the position of a hand [125] (p. 112).

As an alternative to colour-based features, Petersen and Stricker [125] suggested that geometric posture-invariant local constraints found on finger appearances could be used to detect hands in images, as shown in Figure 2.2. They detected fingers by applying another constraint where the maximum angle between an outstretched finger and the principle hand axis is constrained by an upper-bound limit. They subjectively evaluated their method on a sequence of 140 images and showed a good detection rate for an open-hand sequence. Their method, however, is limited to open-hand postures where the fingers are visible.

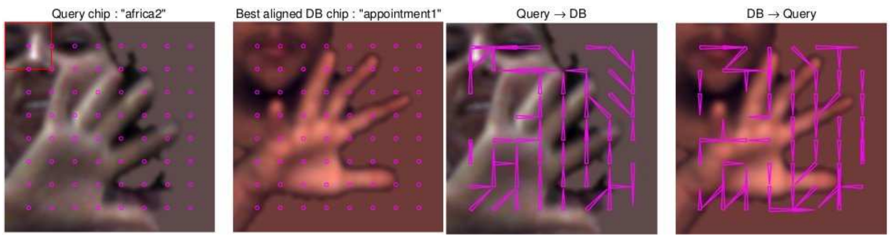


FIGURE 2.3: The red box (top left of image) represents the ROI for a HOG patch and each circle represents the center of the ROI for each HOG patch [151] (p. 92).

In Thangali and Sclaroff [151], a histogram of oriented gradients (HOG) was used. They further proposed an alignment step to allow for non-rigid deformations between image-region pairs. This step measured an aligned distance between image regions computed using the best-matched HOG feature vector in the local neighbourhood for each HOG feature location, as depicted in Figure 2.3. The aligned distance was used with a support vector machine to classify hand-image regions. Their training and testing set consisted of 4000 hand and 8000 non-hand image regions extracted from a set of American Sign Language videos. They showed that improved results are obtained using their approach compared to the rigid matching in the traditional HOG and the vocabulary-guided pyramid-matching kernel.

Similarly, Zondag et al. [175] proposed an algorithm employing HOG, but combined it with stump-and-tree weak classifiers and two variations of the AdaBoost algorithm, namely, Discrete AdaBoost and Gentle AdaBoost. They provided a comparison between using HOG features and Haar-like features for the hand-detection task. They evaluated their algorithm on a dataset of 40 000 positive and 100 000 negative samples, where two-thirds of the samples were used for training and one-third for testing. Their results indicated that although Haar-like features performed faster than HOG features, the HOG features obtained better false-positive-average rates.

FIGURE 2.4: Hand detection applied to three simple hand postures [166] (p. 286).

The same comparison between AdaBoost algorithms was performed by Xiao et al. [166].

However, they applied an LBP feature-extraction method. Their comparison was performed between the Discrete, Real and Gentle AdaBoost algorithms. They trained their classifier on a set of 2 577 positive and 2 000 negative samples containing only three hand postures, shown in Figure 2.4. Their test set consisted of 482 images. Their results indicated an average accuracy of 92.7%, 89.8% and 84.5% for the Gentle, Discrete and Real AdaBoost respectively.

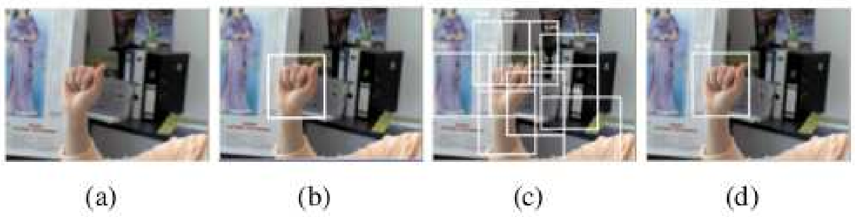


FIGURE 2.5: (a) The original image; (b) the selected ROI used for training; (c) the classification of AdaBoost using a single positive sample only; (d) the output of AdaBoost using positive and negative samples [116].

A framework using an on-line version of AdaBoost was proposed by Nguyen et al. [116]. In their framework, three features were combined and used, namely, local orientation histograms, Haar wavelets and a simplified version of the LBP that only uses a four-neighbourhood. The motivation behind their framework was to update the classifier during the training process when new samples were provided; however, the initial samples still needed to be supervised, as illustrated in Figure 2.5. They evaluated their framework on two datasets, with the first containing three videos and the second containing a total of 673 frames. Their results indicated an average of 99.9% on the two small datasets used.

Multiple features with a single classifier were similarly used by Mittal et al. [109]. They proposed a two-stage approach for detecting hands in images. In the first stage, their approach operates by identifying hand hypotheses from three complementary methods, namely, a part-based deformable model based on HOG features, a skin-colour detector and a context-based detector. In the second stage, a support vector machine (SVM) classifier was trained to verify the hand hypotheses based on the confidence scores obtained from the first stage. They further introduced a publicly available dataset consisting of 5 628 images. Their results indicated a recall rate of 85.3% when using all three methods, compared to a recall rate of 74.1% when employing the part-based detector only. They further evaluated their method on two external datasets and achieved a good result.

Mattheij and Postma [106] followed the same two-stage approach that included feature extraction and classification. Features were extracted using integral images and Haar-like features, and were classified using random forests. They evaluated their detector using 10 fold cross-validation where 2 880 training and 320 testing samples were used for each fold. Their results indicated an accuracy, recall and precision rate of 69.5%, 78.9% and 66.6%, respectively.

2.2 Context-based object tracking

In general, context-based object tracking is the process of tracking an object using contextual information about the object [160]. In temporal context tracking, temporal information is used to track an object, such as employing the Kalman filter [160]. Contextual information has also been used, where knowledge about the tracked object and their mutual relationships is utilised [114]. Spatial-context tracking is defined as using a set of objects or features within a local region surrounding the object being tracked [160].

2.2.1 Knowledge-based context

Multiple-object tracking have been proposed by Duan et al. [46] by modelling mutual context objects. They introduced a framework that combined individual-object models with mutual-relational models. The mutual-relational models consisted of three components, namely, a relational graph that indicates objects that are related, mutual-relation vectors calculated within the relational graph to measure the impact one object has on the other, and relational weights that balance all interactions using an on-line latent SVM. They evaluated their framework on ten-object tracking sequences that included occlusion, appearance changes, rapid motions and varying illumination. They compared their results to several other object-tracking methods and showed an improved detection rate. However, their method ultimately depends on the tracking of the individual object models in order to build relatively strong connections.

Similarly, in Lao and Zheng [89], multi-target tracking was applied to a group of objects that exhibit a common motion pattern with individual variation, as seen in Figure 2.6. Their aim was to learn motion correlation from the trajectories of multiple targets

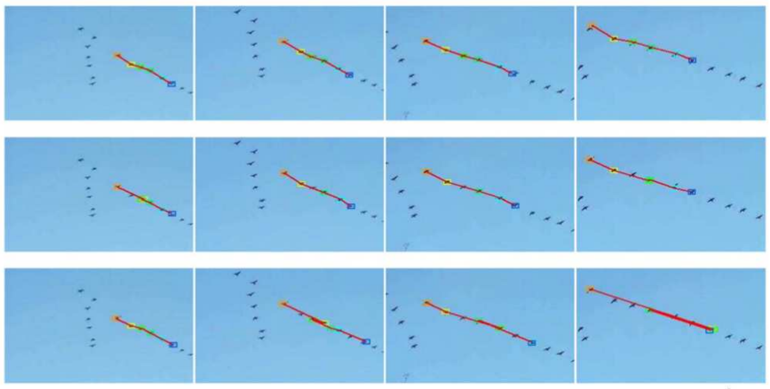


FIGURE 2.6: Tracking multiple targets that exhibit a common motion pattern [89] (p. 812).

while integrating the learned correlation into a sampling process to improve the tracking efficiency. They proposed a statistical framework that embedded the learned correlation among targets and the most recent observations into a proposal distribution based on Markov Chain Monte Carlo particle filters. They suggested that the position of a target can therefore be predicted based on its own local observation, its previous motion and the correlated observations of other targets. They compared their framework with multi-target tracking and showed a lower tracking-error rate.

2.2.2 Temporal context

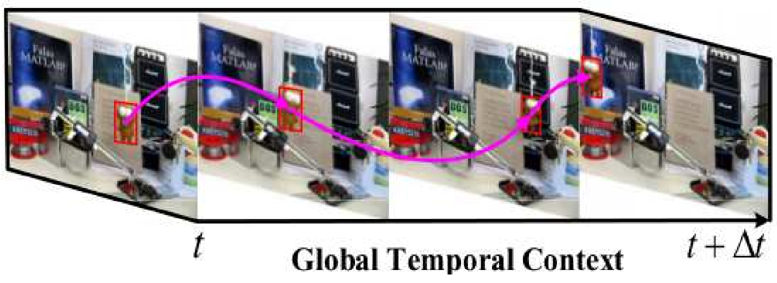


FIGURE 2.7: A global temporal context constraint applied to tracking [163] (p. 719).

A single framework that combined spatial and temporal contexts to predict the location of a target was proposed by Wen et al. [163]. Their temporal context model was constructed by updating the linear subspace method with continuous positive samples and considering the correlation between them, as illustrated in Figure 2.7. The spatial-context model was constructed from contributors around the target that have the same region size and consistent motion correlation with the target. Furthermore, weak contextual contributors were selected by a boosting method to form a strong supporting field. They evaluated their system on ten sequences and showed improved accuracies compared to several object-tracking methods.

2.2.3 Spatial context

In Cerman and Hlaváč [25] it was shown that a Markov random field can be used with contextual information for object tracking. They proposed a two-step process for object tracking, where the first step involved representing an image by a set of feature points tracked by a standard tracker, as depicted in Figure 2.8. In the second step a semi-supervised learning algorithm was proposed to label the feature points as an object, background or companion model. The companion model represented the spatial context of the object and contained non-object feature points that have a similar motion to the target object. Furthermore, instead of using a voting scheme, they proposed to use a 3D graph of tracked feature points and automatically labelled the points such that the energy of the Markov random field on the graph was minimised. They evaluated their system on a few video sequences and showed a positive result for tracking objects with a short occlusional state.

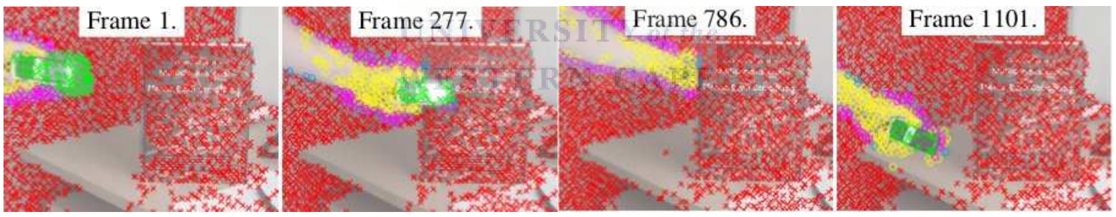


FIGURE 2.8: Each feature point is classified as either object, background and companion model [25] (p. 2127).

A layer-based approach was proposed by Cerman et al. [26] where the foreground layer included the target object and other image regions (the companion model) that move coherently with the object whether it is temporary or permanent. The background layer consisted of all other objects not connected to the object. While tracking, the companion model was adopted on-line and gradually extended by neighbouring image regions in the foreground layer. They demonstrated their tracking system on several videos and showed a better result than tracking without context. Their method, however, is unable to reinitialise after failure and is dependent on a static background assumption.

A similar approach was used by Grabner et al. [64]. They exploited a model that learns features in an image and used them to predict the position of a target. To associate

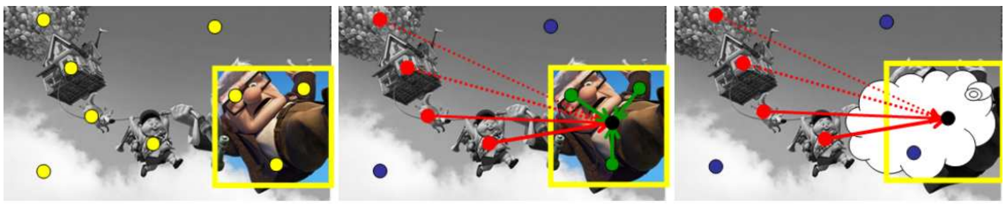


FIGURE 2.9: The yellow box represents the target. The green, blue and red dots represents the features that belong to the object itself, the uncorrelated features that are discarded and the features that are used to vote for the position of the object, respectively [64] (p. 1286).

and disassociate features with the tracked object, they proposed a method based on the Generalised Hough Transform. In contrast to the companion model of Cerman et al. [26], multiple features could be associated at the same time and are not confined to regions surrounding the target. They only focused on features that were coupled with motion (red and green dots) and disregarded those that were stationary (blue dots), as illustrated in Figure 2.9. Furthermore, they adopted a voting strategy on the features where the highest votes were given to features lying on the object. They evaluated their approach on the ETH-Cup[1] sequence and obtained a precision and recall rate of 97% and 89%, respectively. Furthermore, their subjective evaluation showed positive results. Their system, however, is limited by rapid motion that abruptly changes the features.

2.3 Hand-tracking

Hand-tracking is the process that continuously estimates the hand movement and spatial locations throughout an image sequence [34]. A number of hand-tracking approaches have been proposed and vary from those using an auxiliary approach to those using a passive approach. In the following subsections these approaches are discussed.

2.3.1 Auxiliary approaches

The use of auxiliary approaches in hand-tracking employs intricate devices such as data gloves, data suits, or position markers to measure the spatial positions and joint angles of the hands [90]. Although these devices usually offer near to real-time performance and more accurate information, they are an inconvenient and impractical solution because they may hinder the naturalness and ease of signing. Furthermore, these devices require

[1]ETH Zurich - Department of Information Technology and Electrical Engineering video sequence

calibration according to an individual's needs. Recent studies following an auxiliary approach have rather focused on tracking finger movements using these special devices.

2.3.1.1 Two-dimensional input

Pamplona et al. [122] proposed the use of markers displaying QR codes on the fingertips. To estimate the hand and finger movements, they attached a camera to the wrist to recognise the codes and described their prototype as an image-based data glove.

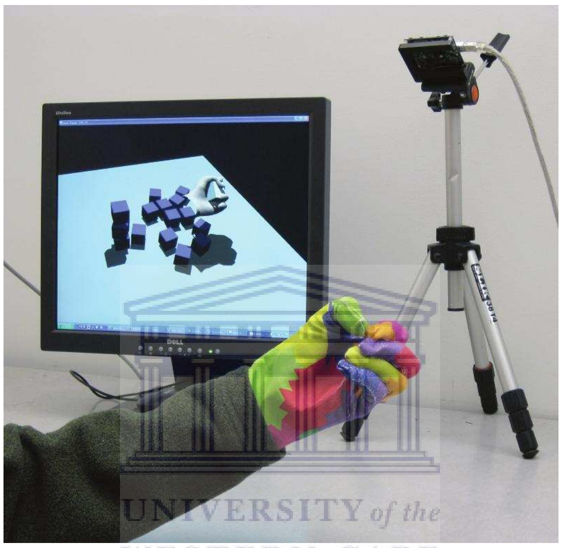


FIGURE 2.10: The gloves were specially designed using colour patches to track the fingers and hand while dealing with occlusion [161] (p. 63).

More recently, coloured glove-based methods have also been used. Wang and Popović [161] used a multi-coloured glove imprinted with a custom pattern with a nearest-neighbour approach to track the individual articulation of the fingers and the global hand pose, illustrated in Figure 2.10.

Auxiliary methods have also been combined with passive methods such as in the work of Prisacariu and Reid [127]. They proposed a 3D hand-pose tracker that uses an accelerometer to deal with ambiguities from hand silhouettes. In addition, they employed a region-based energy function that uses the contour of a 3D hand model by embedding it inside a level-set function where the region statistics were represented by a variable bin size, colour histograms and on-line adaptation.

2.3.1.2 Three-dimensional input

Various devices have been used to extract depth from image sequences. In 2009, the ZCam (a depth camera based on the ToF principle) was sold to Microsoft (MS), where it was improved and reintroduced commercially as the MS Kinect Camera in 2010. The intended primary use of the MS Kinect Camera is an accessory to the Microsoft XBOX 360 gaming device. Unlike other ToF cameras, MS Kinect instead uses a projector, an RGB colour-space camera and an infrared light.

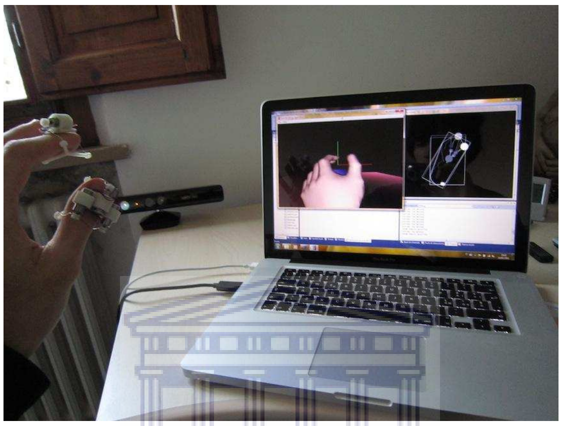


FIGURE 2.11: The Kinect sensor is used to track the main points on hands [55] (p. 319).

In Frati and Prattichizzo [55], wearable haptics devices were used with a Kinect to provide feedback on finger movements, as shown in Figure 2.11. In Ma et al. [102] magnetic signals were provided by small magnets fixed to fingernails and linked to magnetic sensors on a wristband to track the movement of fingers. Similarly, Aristidou and Lasenby [11] placed markers on the fingertips and wrist, and an inverse kinematics solver was used to fit the joint movements to a hand model.

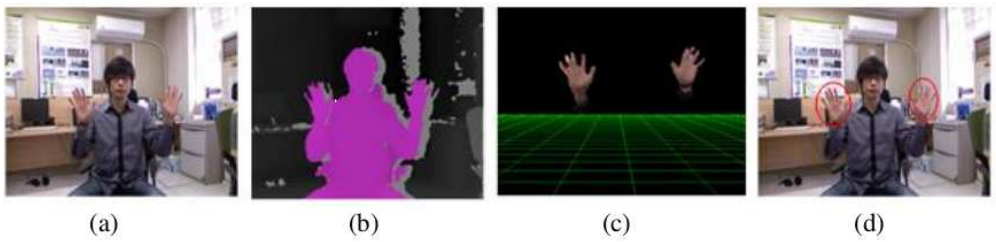


FIGURE 2.12: The images correspond to the (a) original image; (b) depth image; (c) segmented area; and (d) the output image [167] (p. 151).

In Xu and Lee [167] an improved CAMShift algorithm combined with depth information was proposed for tracking hand motion using the MS Kinect. They showed that by using this device, the hands can be isolated from the background, as illustrated in Figure 2.12. Their results showed an 88.75% accuracy on eight very basic gestures.

Similarly, Chai et al. [28] proposed a system that combined skin-colour information and depth information extracted using the MS Kinect. To address the hand-tracking problem, they proposed an energy-function-optimisation strategy by integrating the appearance, location and depth cues. They found that synchronisation between the colour and depth data was inconsistent. Therefore, their main challenge was to determine when the two streams of data were synchronised in their tracking algorithm. They evaluated their algorithm on 300 sign language videos and showed a correct detection rate of 91%.

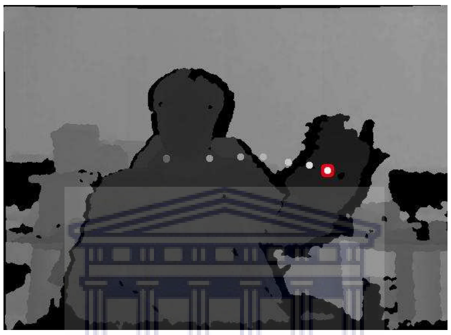


FIGURE 2.13: The output after using the depth threshold to find the hand and the Kalman filter to track the hand [124].

A different type of depth sensor, known as the PrimeSensor, was used by Park et al. [124]. They proposed a hand-tracking system using depth information and the Kalman filter. They generated a motion image from the depth image and detected the hands using spacial filtering, morphological processing and motion clustering. To track the hands, they used a simple Kalman filter, as shown in Figure 2.13. They evaluated their algorithm on six gestures and showed a positive result.

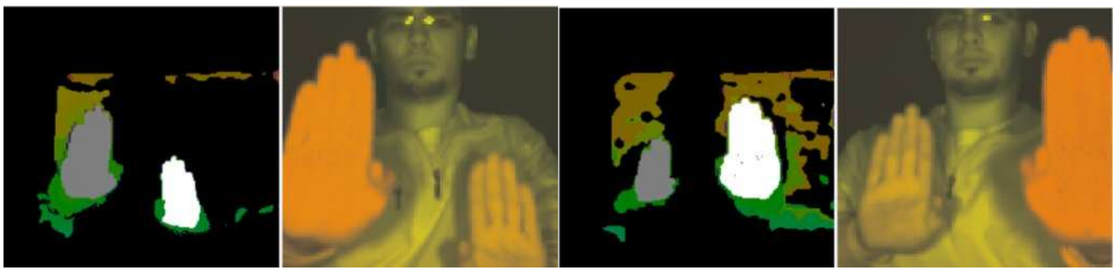


FIGURE 2.14: The depth data are used to track the hands, where the grey and white clusters represent the right and left hand, respectively [27].

ToF range cameras have also been used to infer depth from the scenes being viewed. These cameras determine depth data by measuring the required time for light to travel to an object and back to the camera. In Cespi et al. [27] the intensity and depth data from a ToF camera were used to segment and track the hands, as illustrated in Figure 2.14. The hands were segmented using a hierarchical clustering technique implemented as a parallel merging algorithm on the graphics processing unit (GPU). The hands were then tracked based on the distance between the two hand clusters. Their results showed that although depth can successfully be used to track both hands, when both hands have the same depth measurement while crossing, it leads to tracking failure, since the hands merge into a single-hand cluster.

Suau et al. [147] also used a ToF range camera to resolve ambiguities and overlaps between the head and hands while tracking. Using the same hierarchical clustering technique used by Cespi et al. [27], Suau et al. [147] used depth data to select a maximum of two clusters representing each hand. They tracked the hands by measuring the Hausdorff distance between the current and previous hand position. However, they assumed that a cluster detected further right or left would correspond to the right or left hand, respectively. When the hands crossed each other they considered it as a single hand and could not distinguish between the two. Furthermore, when the hands had the same depth, it led to tracking failure.

2.3.2 Passive approaches

Hand-tracking using passive approaches is able to determine the spatial locations of the hands by using various image processing algorithms in non-invasive ways. These methods offer more practical solutions and are capable of achieving near to real-time performance. Studies following a purely passive means can be further grouped into model-based and appearance-based methods.

2.3.2.1 Model-based methods

In model-based approaches the current state of the hand is estimated by matching a 3D hand model to the observed image features. The approach can therefore be described as a search problem in a high-dimensional space [44]. Recent work in model-based approaches uses multiple constraints such as object textures, shading, edges or lighting that are incorporated into a hand model, along with a variational framework [42] or a probabilistic graphical model [100] that is adapted to the task of hand-tracking. These methods are more popular with single hand-tracking due to the complexity involved in minimising the error cost between the image features and the model projection.

A 3D hand model constructed with truncated quadrics for the recovery of 3D hand motion was used by Kerdvibulvech and Saito [83] They extracted corresponding features between the 3D hand model and the input image, which consisted of edges and a silhouette. The edges were extracted using the Canny edge detection method, and the edge likelihood was determined using the Chamfer distance function. The silhouette was determined using a Bayesian classifier and an outline adaptation of skin-colour probabilities. To predict the next state of the 3D hand model, particle filters were used to track the hands. They further proposed the use of particle filters to automatically recover from tracking failures. In their application's context this was successful, but in general particle filters alone cannot be used for the automatic tracking recovery of hands. Their results were subjectively evaluated and showed satisfactory results.

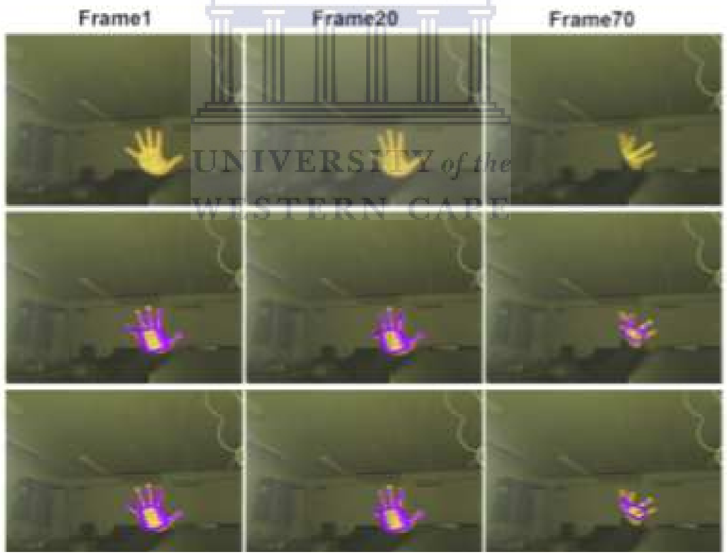


FIGURE 2.15: A two-step minimisation algorithm was applied to match the model to a hand [70].

In contrast to the work of Kerdvibulvech and Saito [83], Henia et al. [70] proposed to combine a non-overlapping surface function with the directed Chamfer distance function to form a new dissimilarity function. They showed that their new dissimilarity function provided better results compared to the directed Chamfer and Hausdorff distance functions. Their algorithm operates in two steps: the first step provides the position and orientation of the palms, which are the global parameters of the hands, and the second step provides the finger-joint angles, which are the local parameters of the hands. This two-step algorithm is more robust to local minima than a one-step minimisation algorithm and also reduces the complexity of the minimisation problem. Their results were subjectively evaluated and showed a satisfactory result, as seen in Figure 2.15. Although the minimisation problem was improved, they failed to handle rotations and self-occlusions.

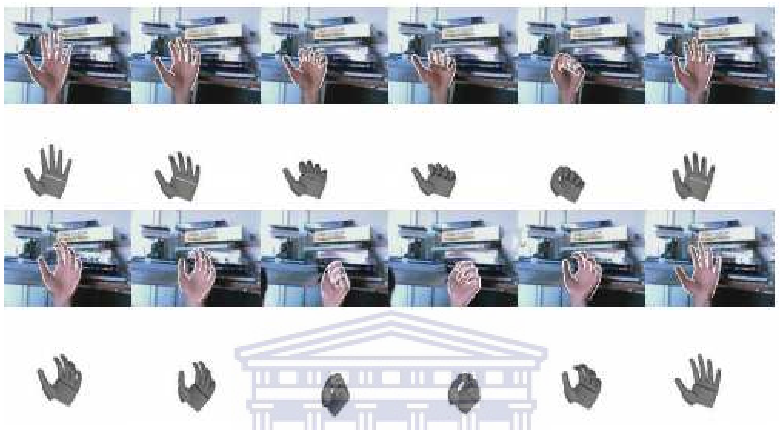


FIGURE 2.16: Hierarchical detection was employed to find the hand and a dynamic model guided the search while approximating the optimal filtering equations [146] (p. 1381).

To overcome the high computational cost of most model-based hand-tracking approaches, researchers have opted to use hierarchical data structures within the framework, as shown by the work of Stenger et al. [146]. They combined hierarchical detection and probabilistic tracking in a single framework. They generated a large number of templates from a 3D hand model and constructed a hierarchy of the templates off-line by partitioning the parameter space. The posterior distribution of the state parameters for each time instant was estimated over these partitions. At initialisation, hierarchical detection was employed to find the hand. In frames that followed, a dynamic model guided the search while approximating the optimal filtering equations. This model was given transition probabilities between parameter space regions and was trained on data that was captured from articulated motion. They have shown that their framework was able to recover 3D motion even when there was self-occlusion; however, this was only based on a single hand. Their results were subjectively evaluated and showed a positive result, as shown in Figure 2.16.

Hierarchical data structures were also used in a multi-camera model-based hand-tracking approach used by Mohr and Zachmann [110]. In their approach they generated a large number of templates using a 3D hand model. They generated a confidence map by applying continuous-edge gradient detection to an input image and the set of templates. This confidence map was combined with a confidence map produced by their skin segmentation and silhouette-area comparison technique. They finally performed hierarchical template matching to handle the large computational cost of their framework.

In Lu et al. [100], a dynamic hand formulation was proposed that integrated multiple cues, such as the gradient-based optical flow, edges and shading within a deformable model framework to track articulated hand motions. The integration of multiple cues allowed the hand model to be refined during tracking where the error in the generalised optical flow function was used to compute the generalised 3D forces that corrected the model shape at each time step. Their framework showed a positive result for different single-hand motions with shading changes, self-occlusions of parts of the hand and rotations.

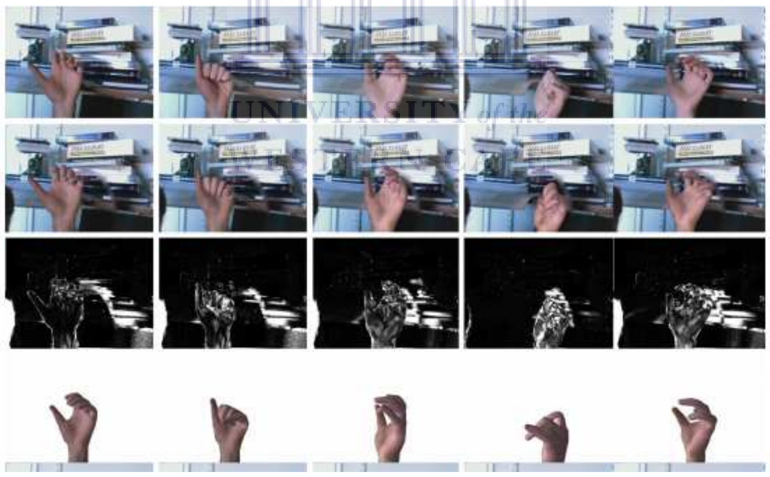


FIGURE 2.17: The rows correspond to the original image, the final synthetic image, the final residual image and the side view of the synthetic image [42].

Similar to Lu et al. [100], de La Gorce et al. [42] introduced a variational framework integrating the geometry of a 3D hand model with scene and object texture, shading, lighting and handling of self-occlusions of the fingers. During each frame in the tracking process they determined the hand posture and illumination parameters of a single hand by minimising the objective function and updating the texture model that was used for registration in the next frame. In contrast to the optical flow method in Lu et al. [100], the similarity measurement used by de La Gorce et al. [42] did not assume limits on the range of displacements, but rather allowed large displacements and discontinuities. Using this framework, they were able to determine the optimal configuration of the hand model through a quasi-Newton descent using the exact gradient of their objective function. However, their method required rough initialisation and therefore assumed the hand to be parallel to the image plane at initialisation. This allowed for a texture estimate to be obtained in the first frame. Their results were subjectively evaluated and produced overall good results, as shown in Figure 2.17. However, their approach failed to recover the precise location of the fingers when the palm was occluded.

In Sudderth et al. [148] it was shown that a non-parametric belief propagation algorithm can be used to track a 3D geometric model of the hand, where a redundant local representation was considered to describe a hand's 3D position and orientation. They showed that the model's kinematic constraints take a simple form in this local representation. Furthermore, the representation of the model allowed colour and edge-based similarity measures, such as the Chamfer distance, to be used in situations where significant self-occlusion did not take place. Similar to particle filters, the non-parametric belief propagation approximates the hand configuration. However, compared to particle filters, the non-parametric belief propagation greatly reduces the dimensionality of the distributions using the graphical structure. To improve the non-parametric belief propagation's computational efficiency, several methods including a novel k-dimensional tree-based method for efficient Chamfer distance computations were used. Their experiments indicated that non-parametric belief propagation can refine initialisation in single frames that contain noise as well as tracking the hand over two extended sequences. They suggested that local hand feature detectors would improve their method's robustness.

2.3.2.2 Appearance-based methods

In appearance-based methods, image features are analysed in order to determine the position of the hand [44]. These approaches can be classified into 3D-based and 2D-based methods [147].

2.3.2.2.1 Three-dimensional input

Hand-tracking using a 3D-based method allows for depth information to be exploited. This information can be used to help distinguish between the right and left hand when there is a difference in depth between the two. In addition, it can be used to distinguish between the hands when occlusion occurs. However, when the depth information is the same for both hands, the problem relates to a 2D hand-tracking problem.

Binocular cues provide a rich source of information about the structure of 3D objects and often involve two or more views. One of the most important sources is obtained from binocular disparity in humans, where each eye receives a slightly different view that is combined with the other eye's view to perceive the depth of objects. This process is referred to as stereopsis. When using a pair of stereo cameras, an object is projected onto different locations depending on the distance of the object [136]. The disparity from stereo cameras varies with the object's distance and therefore is inversely proportionate to the distance of the object. Thus, it is not effective for obtaining small depth differences for objects at large distances [136].

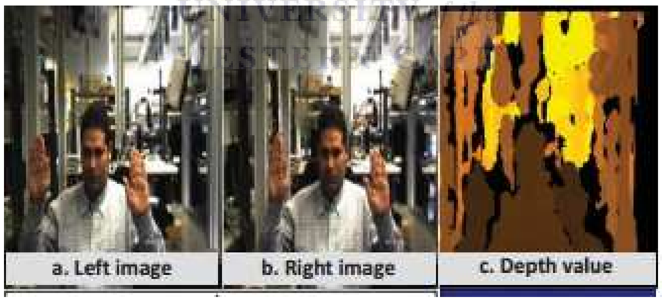


FIGURE 2.18: The left and right camera views of the Bumblebee stereo camera system were used to determine the depth correspondences [49] (p. 26).

In Elmezain et al. [49] mean-shift analysis and the Kalman filter were used with stereo cameras to track the hands in complex environments. Based on cross-correlation and the known calibration data of the cameras, the depth information (as seen in Figure 2.18) was retrieved with skin-colour information and used to segment the hands. After segmenting the hands, the mean-shift analysis was applied and the gradient of the Bhattacharyya coefficient was used as a similarity function to track a given hand between frames and find the centroid for each hand target. The Kalman filter was then used to estimate the position of the hand in the next frame. They evaluated their method on some video samples and showed a positive result.

A similar framework to extract depth information from stereo cameras was used by Park and Lee [123]. They proposed 3D particle filters for the hand-tracking task in a 3D pointing-gesture-recognition system. They used depth information to find a human in the scene and applied skin detection to segment the face and hands. Since they suggested that 2D particle filters often fail, they employed 3D particle filters to track the hands that were based on 2D information and depth information. As part of their framework they used a cascade of two hidden Markov models (HMMs) to estimate the pointing direction. The first-stage HMM mapped the hand position and the second-stage HMM involved gesture spotting. They evaluated their system on 12 individuals for pointing gestures, moving gestures and non-gestures. Their experimental results showed an accuracy of 89% and 99% for gesture recognition and selection rates, respectively.

In Xu et al. [168] colour and depth maps using stereo cameras were employed in a multi-cue-based tracking system for tracking the face and hands. They combined colour histogram-based particle filtering with the mean-shift algorithm to handle rapid and complex motions of the hands by increasing the hit ratio of samples of the particle-filtering method. The position of the hands was then used to estimate the other upper body parts using human kinematics. Their system was evaluated on a music-playing human–computer interaction scenario and achieved an accuracy of 98.95% and 90.12% for the left and right hand, respectively.

2.3.2.2.2 Two-dimensional input

Tracking hands from a 2D view is a more challenging task, since depth cannot be used to deal with occlusion. It is also challenging to segment the hands in complex environments and track them over long image sequences.

The simplest means to track a hand from a 2D view is to employ colour-based methods, because skin has a distinctive colour compared to many objects. In Rautaray and Agrawal [128] a hand-tracking algorithm was proposed based on skin-colour detection using randomised lists and the LAB colour space. They subjectively evaluated their algorithm and demonstrated it using a single hand, as illustrated in Figure 2.19. Methods

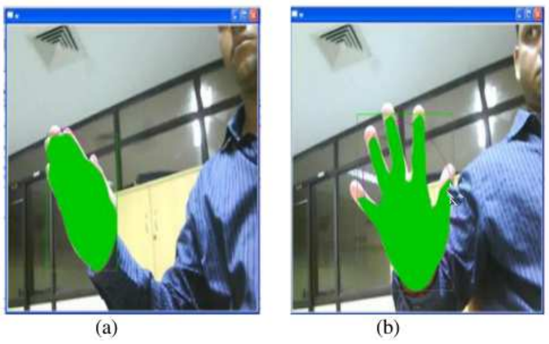


FIGURE 2.19: The output of the tracking system using skin-colour detection only [128] (p. 31).

employing a single cue to track hands from a 2D view have a very high probability of failing in unconstrained environments and when occlusion occurs near the face or opposite hand.

Instead of using a single feature, Coogan et al. [36] proposed to use three features, namely motion, colour and position. They also proposed to use a Kalman filter to detect, but not handle occlusion between the face or hands and to reduce the search space.

In Wen and Zhan [162] the hands were adaptively tracked based on skin colour using the CAMShift algorithm. They proposed a two-step algorithm where the first step consisted of detecting the hands using a contour-based method to extract the five fingertips for each hand. In the second step an improved Grey model that used historical data to make a tracking prediction was combined with the CAMShift algorithm. They suggested that the combination would address the problem of distractors and occlusion handling. They subjectively evaluated their results on a limited set of four videos in a simple environment and showed a positive result. However, their method operates in a similar way to the combination of CAMShift algorithm and Kalman filter as in Luo et al. [101]. When the hands overlap each other and are stationary at the same time, this method would fail and would be unable to distinguish and track the hands correctly. This problem was addressed by Elmezain et al. [48] using a Kalman filter and CAMShift combination with depth information from stereo cameras.

To track a single hand continuously during various pose variations and movements, Kölsch and Turk [86] introduced the flocks-of-features algorithm. This algorithm was inspired by the way birds flock together during flight, exhibiting variability and local

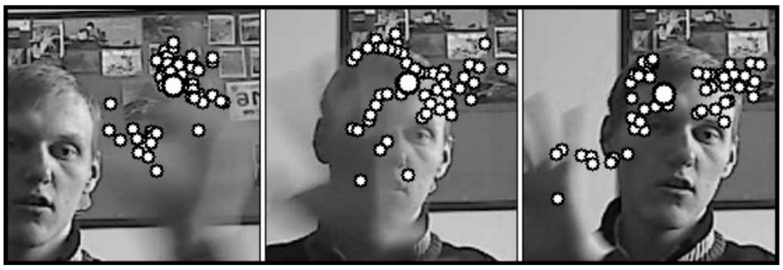


FIGURE 2.20: The modified flocks-of-features algorithm still fails due to rapid movements [54].

individualism while maintaining their clustered form. Fogelton [54] proposed to modify this algorithm by mainly applying the algorithm on a histogram back-projected image based on skin colour, instead of a gray-scaled image as in the original algorithm. An evaluation of several videos showed that this modified algorithm performed twice as well as the original algorithm. Similar to the original algorithm, the modified algorithm still failed to deal with fast movements in front of other skin-coloured objects, as seen in Figure 2.20.

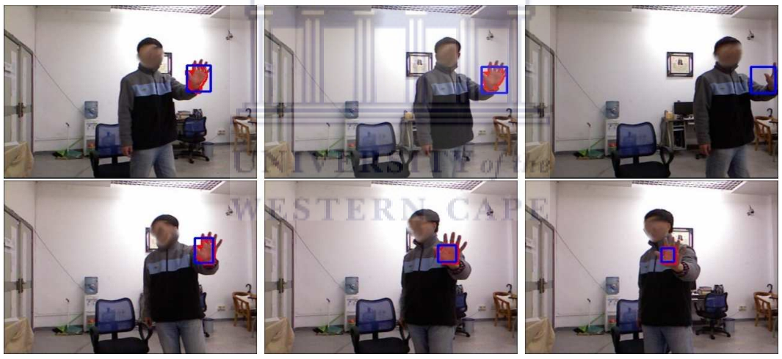


FIGURE 2.21: A sample output of the improved flocks-of-features algorithm [97] (p. 466).

To improve the flocks-of-features algorithm further against distractors and background cluster, Liu et al. [97] proposed a multi-cue flocks-of-features algorithm based on the Boids algorithm [131], which was used to simulate flock behaviour in computer graphics. They furthermore integrated the algorithm with an on-line Hough forest framework that combines Hough voting with the random forest. This integration allowed for the flocks-of-features tracker to refine the tracking and influence the updating of the Hough forest tracker. They evaluated their framework on six videos and compared it to the CAMShift, original flocks-of-features, and Hough tracker on the second video only. They showed that their framework yields a more accurate result than the compared trackers, but still fails when the tracked hand is moved in front of the face. Furthermore, their algorithm was able to track a single hand only, as seen in Figure 2.21.

In Liu and Zhang [98] LBPs and colour cues were combined in a particle-filter framework and used to track the hands. The similarity measurement used in the particle-filter framework was based on the Bhattacharyya distance. They evaluated their results subjectively and showed that by combining LBPs with skin-colour information, an improved hand-tracking method can be achieved than by using either cue alone.

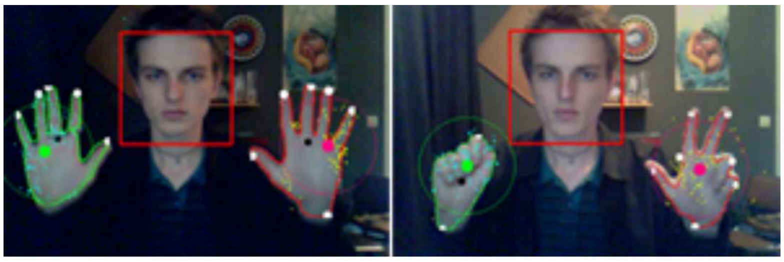


FIGURE 2.22: An example of the output obtained from framework used by Spruyt et al. [144] (p. 3120).

In addition, a particle-filter framework was proposed by Spruyt et al. [144] to track the hands; however, they combined it with motion and colour cues. They suggested that combining motion detection, edge detection, skin-colour detection and colour clustering increases the reliability of their framework against illumination invariance. They also suggested that by combining these cues in their framework, their system automatically recovers from failure and does not need an initialisation phase. Their results were presented visually and showed a satisfactory result, as illustrated in Figure 2.22.

In Ongkittikul et al. [120] only the skin-colour cue was used in their particle-filter framework. However, they used two particle filters to track the two hands individually with the likelihood function based on skin-colour classification. While tracking, they employed the reliability measurement from the particle distribution to adaptively weigh the colour classification. They also embedded the K-means algorithm in the particle-filter framework to discriminate the particle-filter weights when the two hands were close to or moving away from each other. The system was evaluated on 45 videos and compared it using the mean-shift algorithm. The results were subjectively evaluated and showed that the framework yielded better results compared to the mean-shift algorithm, as depicted in Figure 2.23. Although two particle filters were use, their method was unable to distinguish between the left and right hand.

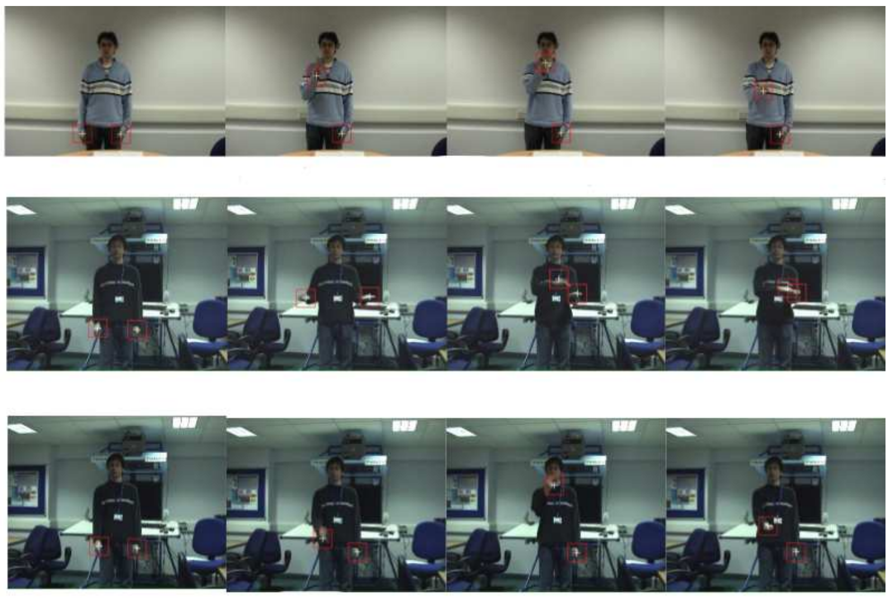


FIGURE 2.23: The tracking of both hands, but not distinguishing between them using the framework of Ongkittikul et al. [120] (p. 205).

Multiple cues that include motion-history images, selected colour-weighted images and gradient orientation templates integrated in a sequential importance resampling (SIR) particle filter was proposed by Chen et al. [32]. The SIR particle filter was used to track the head and hands when they were apart. However, when the head and hands were close to or overlapping one another, they employed the multiple-importance-sampling particle filter with depth-order reasoning to generate multiple tracking targets for the merged hypothesis. They separated the merged hypothesis into several targets based on the hand-shape orientation, motion continuity and occluded face-template cues. They evaluated their framework on only two videos and demonstrated the tracking of both hands.

To track both hands and handle occlusion, many studies suggest using multiple cues or one of the algorithms belonging to the set of sequential Monte Carlo methods such as particle filters or Kalman filters. Some studies have also used data-association strategies to address the tracking problem while handling occlusion. Data association involves linking the detected object or features over time and using the detections to estimate its trajectory [104].

Argyros and Lourakis [10] implemented an approach that applies a non-parametric method for skin-colour detection and tracking in a non-Bayesian framework. They employed the data association strategy to track multiple skin-coloured objects over time.

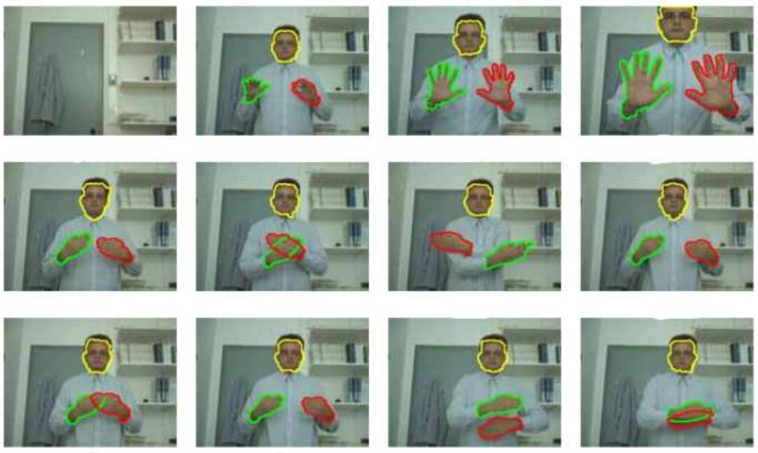


FIGURE 2.24: The output of the framework introduced by Argyros and Lourakis [10] (p. 1190).

Their algorithm operates by intelligently associating each new skin-coloured cluster that has entered the scene with an object hypothesis and assigning it a unique label, thereby treating each object as a separate entity. They evaluated their system on video containing different scenarios, including moving the camera while capturing, and it showed a positive result when tracking the hands and face, as shown in Figure 2.24. The limitations of their method are that it is restricted to constrained environments, unable to discriminate between the different skin areas and unable to recover from tracking failure. However, their method is a good starting point for developing an independent hand-tracking framework in a sign-language-recognition application.

2.4 Discussion and summary

In the preceding sections, approaches on hand detection, context-based tracking, and hand-tracking were described and analysed. Because each of the approaches has its relative strengths and weaknesses, many challenges remain.

In hand-detection approaches using 3D views, it is assumed that any skin-coloured object that is closer to the camera than the body is the hand. These methods would therefore fail when the body and hands have the same depth. Furthermore, obtaining 3D views from specialised cameras is not always a straightforward task and many challenges remain, such as synchronisation, bright light and calibration. On the other hand, several hand-detection algorithms from 2D views have been proposed. The difficulties of detecting hands from 2D views arise from the high configuration space and the geometry of hands. Many algorithms based on 2D input, such as HOG, Haar and LBPs have been inspired by its success in face detection and recognition. Although some algorithms have shown potential, they do not provide a complete solution to hand detection, due to the articulated nature of hands. Recent advances in face-detection applications have also demonstrated the use of the powerful texture-based features of LBPs. LBPs have not been extensively used for hand detection. Nguyen et al. [116] only used a simplified version of the LBP and Xiao et al. [166] used the original LBP, but with the focus more on the comparison between different AdaBoost algorithms. These methods often extract local features from the entire image region, thereby extracting many irrelevant features. The current study therefore investigates whether an algorithm that filters the image region to only extract features from the hands would improve hand detection using LBPs. This approach would effectively deal with the space configuration or geometry of hands.

Different types of context-based object tracking have successfully been applied to increase the accuracy of object-tracking methods. Although knowledge about objects in a scene has been applied to multiple-object tracking, it is more applicable to objects exhibiting a common motion pattern. On the other hand, spatial- and temporal-context information are more often used in single object tracking; however, context-based object tracking has not been applied to hand-tracking. Inspired by Wen et al. [163], Cerman et al. [26] and Grabner et al. [64], the current research investigates whether spatial and temporal context-based object tracking would improve the accuracy of hand-tracking.

In order to conform to the requirements of a mobile SASL translation system, this study is restricted to using 2D video input streams in both constrained and unconstrained environments, thus making it challenging to deal with occlusion and distinguish between the right and left hands. Various methods have been proposed for hand-tracking using 2D input; however, most methods experience difficulties in tracking both hands [36, 128, 162]. Those that do track both hands have difficulties in distinguishing between the left and right hand or dealing with occlusion. To allow for various pose variations and movements, flocks-of-features were introduced [86] and improved [54, 97]. However, the method is prone to distractions near other skin-coloured areas. To deal with such distractions, particle filters are often used. A number of algorithms employing particle filters combined with other cues have been proposed [32, 120, 144]. Although these algorithms show potential for tracking both hands, they are limited by the likelihood function they employ. As an alternative, data association strategies have been applied, as demonstrated by the work of Argyros and Lourakis [10]. However, their method is restricted to constrained environments, unable to discriminate between the objects being tracked and unable to recover from tracking failure. This research therefore investigates whether the algorithm of Argyros and Lourakis [10] can be extended and applied to deal with occlusion and distinguish between hands while tracking in constrained and unconstrained environments from 2D video data. With respect to people, the skin-colour appearance varies significantly. Successfully segmenting an individual's skin pixels would effectively provide a valuable grounding to apply this data-association method. In addition, this research investigates whether a dynamic skin segmentation would provide a suitable foundation on which to base the extended data-association method.

In the following chapter the development of a framework will be explained using the methodology adopted for this research.

Chapter 3

Design Science Research

In the previous chapter an overview was given of the approaches and aspects that researchers have considered when analysing hand detection, context-based object tracking and hand-tracking. In this chapter the philosophical grounding that underpins the research will be discussed to ensure the consistency of the study. The methodology that is informed by the philosophical stance will be described in Section 3.4. In Section 3.5 the methods selected to perform the data analysis will be discussed and motivated. The chapter will be concluded with a summary in Section 3.6.

3.1 Research philosophy

Research philosophy can be defined as a belief about the way in which information about a phenomenon should be collected, analysed and used. All research is based on latent assumptions about what constitutes valid research, whether it is quantitative, qualitative or both, and which research methodologies and methods are appropriate. It should be noted that some researchers use the terms 'methods' and 'methodology' interchangeably [35]. In this research methodology refers to the overall approach or design that lies behind the selection of specific methods, and that links the selection of methods to the desired end result [38]. The methods refer to the various techniques used to collect and analyse data with respect to a particular research question or hypothesis [38]. Therefore, the research approach, the data collection methods and the means of analysis all form part of the research process and are considered under the overall structure of methodology. The methodology, however, is underpinned by the theoretical perspective, which refers to the philosophical stance that informs the methodology and thereby provides a context for the research process and grounds its logic and criteria [38]. Embedded in the theoretical perspective is the epistemology, which refers to the theory of knowledge, i.e. a way of understanding and how knowledge is known.

To ensure the consistency of the study, it is necessary to consider the philosophical basis that informs the decision-making process. Crotty [38] structures this process as four elements that inform one another, as depicted in Figure 3.1

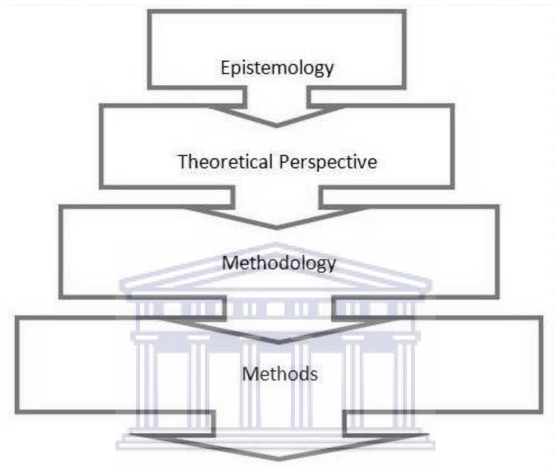


FIGURE 3.1: The decision-making process as four elements that inform one another

According to Crotty [38], the decision-making process in the study can be defined by posing four questions that relate to the four basic elements:

1. What methods will be used?
2. What methodology guides the choice and use of the proposed methods?
3. What theoretical perspective underpins the preferred methodology?
4. What epistemology informs the suggested theoretical perspective?

Postulating the research process in terms of these four elements ensures the soundness of the research and maintains consistency within the study. It also forms an incisive analysis of the process, points out the theoretical assumptions that underpin it and determines the quality of its findings. For each research question or research problem a relevant set of methods should be chosen and used [62]. Hughes [77] states that 'every research tool or procedure is inextricably embedded in commitments to particular versions of the world and to knowing that world'. This implies that the effectiveness of a method is fundamentally dependent on the justification of the epistemology, illustrated in Figure 3.2. In the following subsections these four elements will be discussed in more detail, beginning with a justification of the epistemology.

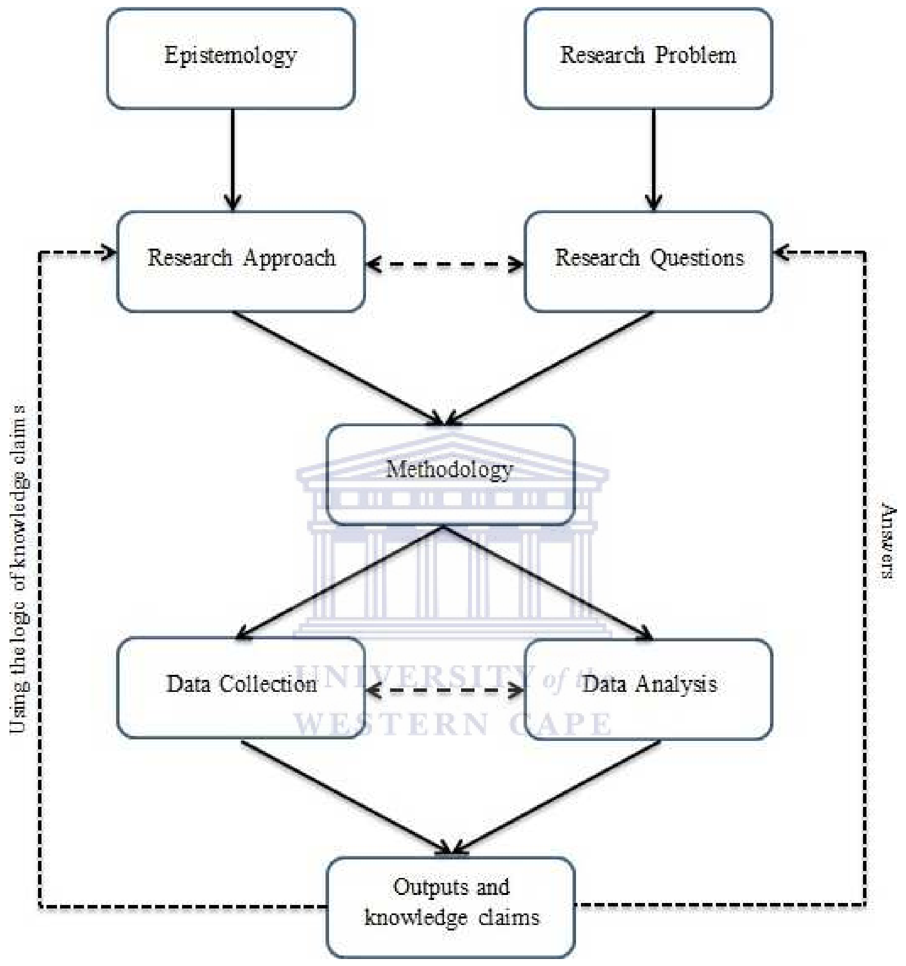


FIGURE 3.2: An illustration of the link between epistemology and the research process.

3.2 Epistemology

Epistemology deals with the theory or nature of knowledge. It is combined with ontology (what things exist) to form the branch of philosophy known as metaphysics. It provides a philosophical grounding for the decision regarding the kind of knowledge that is possible and ensures that it is both adequate and legitimate. Although a range of epistemologies exist, in general epistemological beliefs are seen as lying on a continuum ranging from objectivism to subjectivism. Objectivism holds that a meaningful reality exists independently of the mind and that it can be described by measurable properties that are independent of the observer and subject [38]. On the other hand, subjectivism holds that meaning is imposed by the subject's mind without the contribution of the world, where there is no meaning independent of the mind and where the observer is not independent of the subject [38].

By defining an objectivist epistemological stance, the perspective is that one's feelings about objective knowledge should be separated from it [78][41]. The objectivist perspective provides a theoretical basis for quantitative research, whereby researchers will often be more inclined to conduct quantitative, rather than qualitative, research [78].

There has been much debate on whether it is possible to combine objectivism and subjectivism — the two dichotomies of epistemology [52]. Some researchers believe that objectivism and subjectivism should be regarded as total opposites [117][18], while others believe they should be regarded as a continuum (where they exist on opposite sides of the continuum), thereby complementing one another.

This research takes on more of an objectivist stance and is thus inclined to quantitative research.

3.3 Theoretical perspective

A distinction can be made between two major research philosophies, namely positivism (also referred to as scientific) and interpretivism (also referred to as anti-positivism).

A central tenet of positivism is that one can objectify the phenomena under investigation and thus view reality from an objective viewpoint [132]. Furthermore, positivists observe social behaviour by taking a 'scientific' perspective with an objective analysis [153] and thus separate the object and the subject in the study. Positivists focus on the facts or the given and ignore everything else. They also argue that phenomena should be isolated and observations should be repeatable. However, Bryman and Bell [22] caution against assuming that the positivist philosophy and scientific approach are synonymous. They note that although there are some instances where an inductive strategy is employed in positivist research, generally such research tends to be based on deductive theorising [13].

Gall et al. [57] state that positivist research is essentially synonymous with quantitative research. They also state that researchers adopting a positivist stance develop knowledge by applying numerical analysis to numerical data collected from the observable behaviours of samples.

Interpretivism or anti-positivism denies any objective reality and assumes that any social reality can only be studied from the perspective of the individuals that are directly involved, through social constructions such as consciousness, language and shared meanings [132]. Interpretive research generally attempts to understand phenomena through the subjective meanings that individuals assign to them and contend that phenomena shoud be studied in their natural environment. Interpretivism therefore focuses on the domain of meaning and the methods of studying it [62]. Furthermore, interpretivists acknowledge that there may be many interpretations of reality; however, they hold that these interpretations from an anti-positivistic stance form part of the scientific knowledge they are pursuing.

The research considerations with regard to the current study followed a positivist theoretical perspective, since the study has a more objective approach. This particular paradigm offers a range of methodological choices where researchers can apply quantitative methods.

3.4 Methodology

The theoretical perspective that underpins this research is mostly of a positivist stance and thus the methodology best suited to manage it was design science research (DSR).

3.4.1 Design science research

The DSR methodology has historical origins in engineering and architecture where much of the literature is based on a positivistic epistemology. Gregory [65] defines DSR as a '*general research approach with a set of defining characteristics that can be used in combination with different research methods*'. In the current research, DSR will be used, thereby addressing the objective aspects of the stated research questions.

An integral part of the DSR framework is searching for and solving practically relevant real-world problems [65]. The iterative process of a general DSR cycle consists of six distinctive stages, displayed in Figure 3.3 [159].

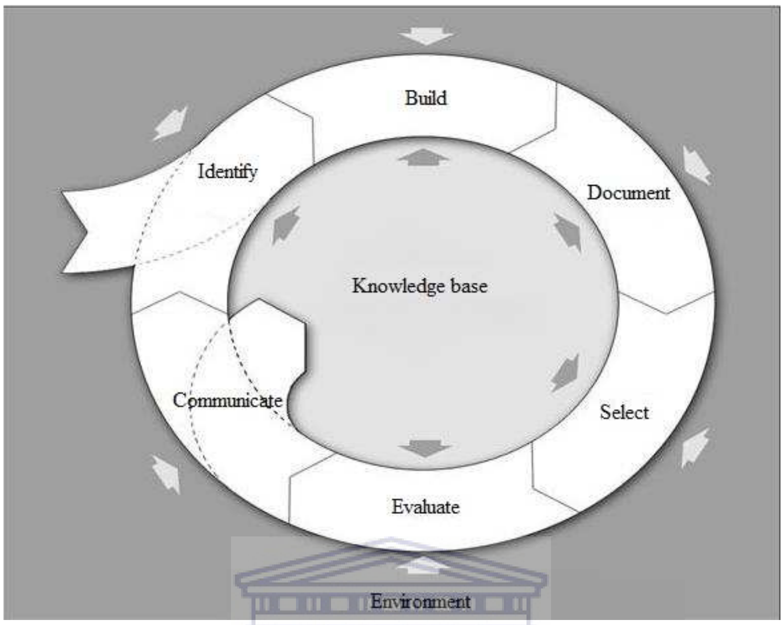


FIGURE 3.3: The iterative process of a general DSR cycle consists of six distinctive stages [159].

These six stages can be further elaborated as follows:

1. The needs and foundation of the requirements are identified in Chapter 1.
2. Using the requirements, an artifact or software system is developed that also delivers functionality, as discussed in Chapter 5.
3. The software system is represented and documented (discussed in Chapter 5), which includes the documentation of the limitations (discussed in Chapter 7).
4. The evaluation criteria and methods are selected, as discussed in Chapter 3 and 6.
5. The software system is evaluated using the selected evaluation criteria and methods, as discussed in Chapter 6.
6. Additional requirements are communicated for another iteration of the DSR cycle, as discussed in Chapter 6.

To ensure that the research addresses the key aspects of the DSR methodology, Hevner and Chatterjee [71] proposed a list of questions:

1. What is the research question (design requirements)?
2. What is the artifact? How is the artifact represented?
3. What design processes (search heuristics) will be used to build the artifact?
4. How are the artifact and the design processes grounded by the knowledge base? What, if any, theories support the artifact design and the design process?
5. What evaluations are performed during the internal design cycles? What design improvements are identified during each design cycle?
6. How is the artifact introduced into the application environment and how is it field tested? What metrics are used to demonstrate artifact utility and improvement over previous artifacts?
7. What new knowledge is added to the knowledge base and in what form (e.g. peer-reviewed literature, meta-artifacts, new theory, new method)?
8. Has the research question been satisfactorily addressed?

In addition, Hevner et al. [72] state that the list '*provides a structured path for doctoral students interested in using this methodology in their research, structuring and legitimising their research*'. In order to answer the research questions, the DSR cycle was applied to a detection, tracking and learning phase, as depicted in Figure 3.4.

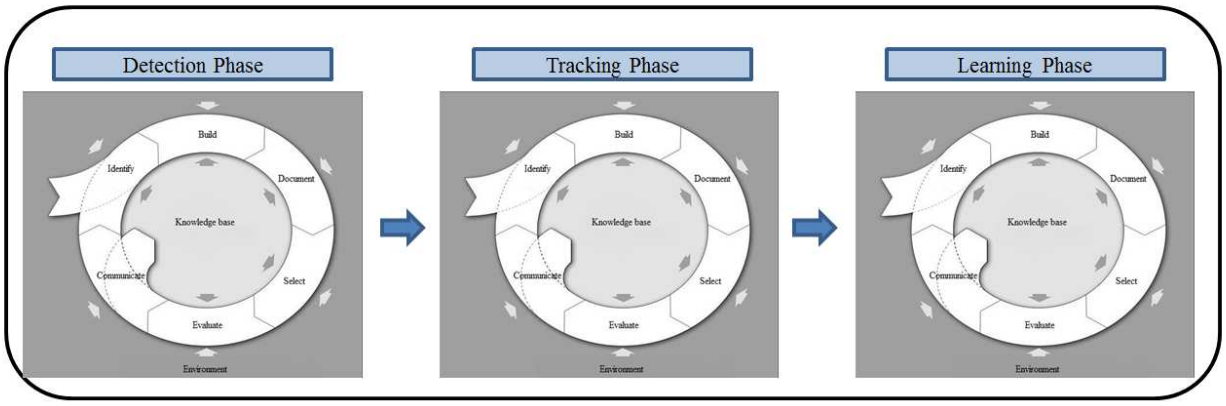


FIGURE 3.4: The DSR cycle applied to each of the phases.

3.5 Methods

In computer vision, the objective evaluation of tracking algorithms may either focus on analysing the tracking algorithm itself or analysing the results thereof. In the context of vision-based tracking algorithms, these approaches are clearly expressed by Maggio and Cavallaro [104] as either analytical or empirical methods.

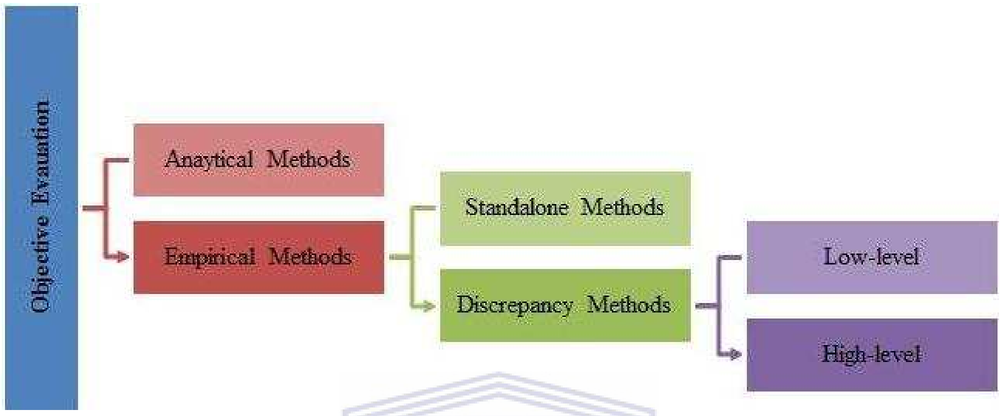


FIGURE 3.5: A taxonomy of objective evaluation methods for tracking algorithms.

Analytical methods evaluate tracking algorithms based on their principles, complexity and requirements. These methods have the benefit of not requiring the implementation of the tracking algorithm; however, since tracking algorithms may form complex systems consisting of several subsystems, the disadvantage is that not all the properties of the systems may be easily evaluated [104].

On the other hand, empirical methods evaluate tracking algorithms based on their results. These methods have the benefit of comparing several tracking algorithms that are applied to a dataset that is relevant to a given application. The empirical methods can further be divided into stand-alone methods and discrepancy methods [104]. The difference between the methods is that discrepancy methods require the use of ground-truth data, whereas stand-alone methods do not. Empirical stand-alone methods measure the 'goodness fit' of a tracking algorithm result that is based on some quality criteria such as the smoothness or consistency of a tracking trajectory. Empirical discrepancy methods measure the variation between the tracking algorithm result and the ground-truth result, and can further be divided into low-level and high-level methods [104].

Low-level methods evaluate the results of a tracking algorithm independently from an application, whereas high-level methods evaluate the results in the context of the application [104]. This research therefore adopts a high-level empirical evaluation protocol. A taxonomy of objective evaluation methods for tracking algorithms is shown in Figure 3.5.

3.6 Summary

The chapter described the philosophical grounding that underpins the research in order to ensure the consistency of the study. It described how the epistemology and theoretical perspectives inform the research process. It was concluded that in order to follow a more objective approach, a positivist theoretical perspective should be taken. It was also concluded that the DSR methodology is relevant to managing this philosophical stance. In addition, the six-stage cycle of the DSR framework that structures the research was discussed. Moreover, methods used to evaluate tracking algorithms objectively were described, where it was shown that a high-level empirical evaluation protocol would be applicable. In the following chapter the components that form part of the detection, tracking and learning phases will be explored.

Chapter 4

Detection-Tracking-Learning

In this chapter the components that form part of the detection, tracking and learning phases of the proposed framework will be explored. The different algorithms included in these components will be compared and their applicability to addressing the research problems will be explained. In Section 4.1 the components included in the detection phase will be discussed. These components consist of LBPs, random forests and SVMs. In Section 4.2 the tracking phase, which includes a unique skin-segmentation algorithm and a data-association method that allows for multiple skin-coloured object tracking in both constrained and unconstrained environments, will be described. In Section 4.3 the components used to form a context-based tracking approach to recover the hands from tracking failure will be explored.

4.1 Detection Phase

The detection phase was employed in the first DSR cycle (see Figure 4.1). The phase consists of LBPs, random forests and SVMs with the aim of developing a suitable hand-detection algorithm.

4.1.1 Local binary patterns

The LBP was introduced by Ojala et al. [119], who originally applied it to texture patterns. Since then it has become increasingly popular and has been widely used to

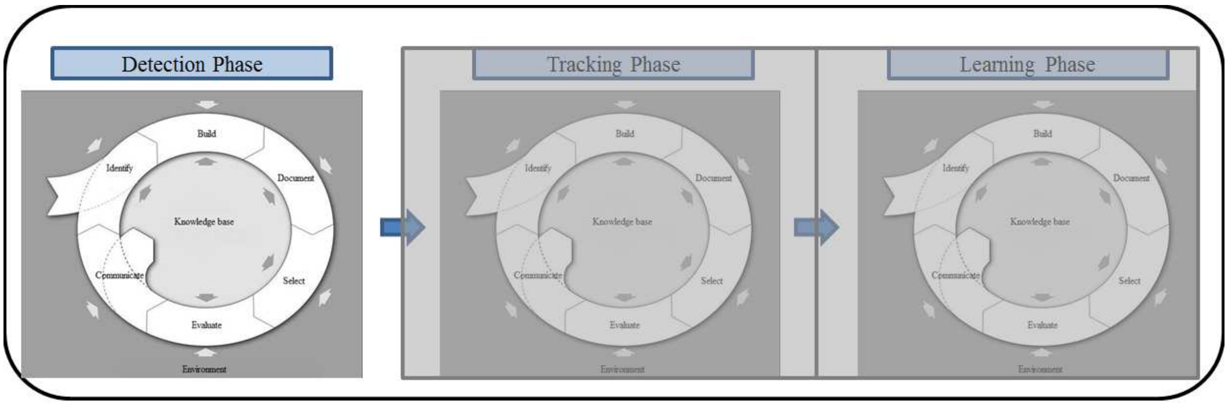


FIGURE 4.1: The first DSR cycle consisting of the detection phase.

encode various objects appearances such as people detection, food types, cars and faces [115]. It is often applied to greyscale images and the derivative of the intensities [94]. The success of the LBP operator is attributed to its invariance against illumination changes, its discriminative power and its computational simplicity. It describes the neighbourhood of a given pixel by generating a binary code from the binary derivatives of that pixel.

In its basic form, the LBP operates in a 3x3 neighbourhood of each pixel. The pixels in this neighbourhood are thresholded according to their centre pixel value. A binary code is obtained by generating a zero or a one binary value if the neighbouring pixel is smaller or larger than the centre pixel, respectively. Each binary value is multiplied by a power of two and then summed to obtain a label that is assigned to the centre pixel. In this neighbourhood, a total of 256 (2^8) different labels exist. In Figure 4.2, an example of the basic LBP operator is shown.

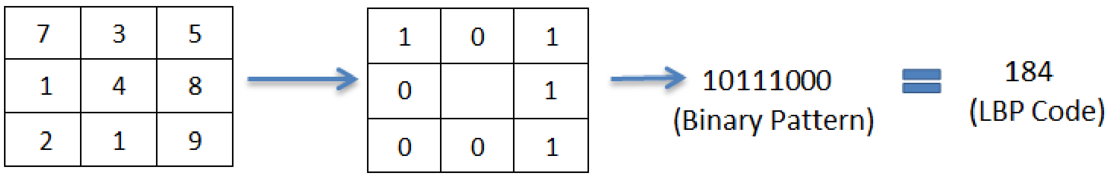


FIGURE 4.2: An example of the LBP process that determines the decimal-label value.

In contrast to the basic form, Ojala et al. [119] derived a generic form of LBP based on circular neighbourhoods and linearly interpolating the pixel values that is not restricted by the size of the neighbourhood or the number of sampling points. This form allows LBPs to be used on different scales and is referred to as the extended LBP (or the circular LBP).

In the extended LBP (ELBP) the decimal label for each centre pixel, c, in an image is given by [126]:

$$LBP_{P,R}(x_c, y_c) = \sum_{p=0}^{P-1} s(i_p - i_c)2^p \tag{4.1}$$

where P is the number of sample points or neighbours of c, R is the radius of the neighbourhood, i_p is the value of the neighbours, i_c is the value of c and the threshold function s is defined as:

$$s(x) = \begin{cases} 1 & \text{if} \quad x \geq 0 \\ 0 & \text{if} \quad x < 0 \end{cases} \tag{4.2}$$

In Figure 4.6, examples of LBPs with different sampling points and radii are shown:

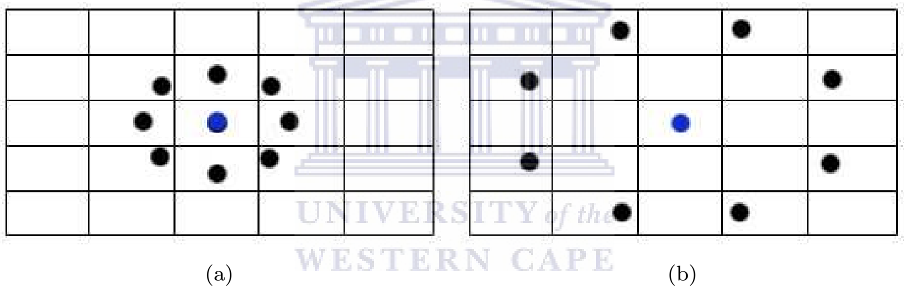


FIGURE 4.3: (a) An LBP with eight sample points and a radius of 1; (b) an LBP with eight sample points and a radius of 2.

Given an image or region containing LBP codes, a histogram can be used to describe the entire image or region, with the advantage of being invariant to image translation. Invariance to the image or region size is achieved by applying normalisation to the histogram. This histogram can be determined as [66]:

$$H(k) = \sum_{x,y} f(LBP_{P,R}(x, y) = k), \;\; k \;\epsilon\; [0, ..., 2^P - 1] \tag{4.3}$$

where the function f is defined as:

$$f(x, y) = \begin{cases} 1 & \text{if} \quad x = y \\ 0 & \text{if} \quad x \neq y \end{cases} \tag{4.4}$$

The histogram contains information about the distribution of primitive micro-features that include curves, edges, lines, spots, flat areas and corners, as shown in Figure 4.4.

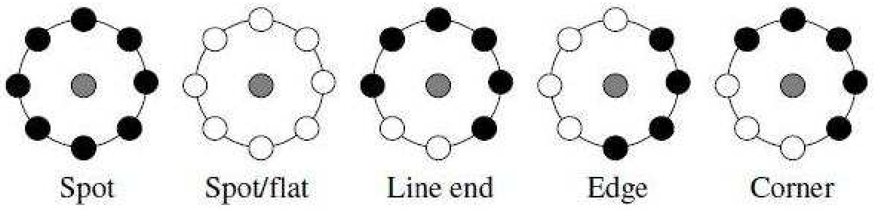


FIGURE 4.4: Examples of the different micro-feature information that are extracted using LBPs.

4.1.1.1 Uniform LBP

If P neighbouring pixels are considered, then from equation 4.3 it can be deduced that the number of histogram bins will be 2^P. To reduce the dimension of the histogram, another extension to the original LBP, called uniform LBP, was introduced. The uniform LBP is denoted as $LBP_{P,R}^{U2}$, where the superscript "$U2$" means that LBPs are considered to be uniform when at most two bitwise transitions (U) from zero to one and vice-versa are present in their circular binary representation [66]. For example, the LBP codes 11111111 ($U = 0$), 11110000 ($U = 1$) and 00111100 ($U = 2$) are uniform patterns, since there are at most two transitions from zero to one and/or one to zero. The LBP codes 00110001 ($U = 3$) and 00110110 ($U = 4$) are non-uniform patterns, since there are at more than two transitions from zero to one and one to zero. In mapping from $LBP_{P,R}$ to $LBP_{P,R}^{U2}$, a different label is assigned to each uniform LBP and all non-uniform LBPs are assigned the same label. Therefore there are $P*(P-1)+3$ different labels assigned to pixels when using uniform LBPs. For example, a neighbourhood of eight sampling points produces 58 uniform labels and 1 non-uniform label; thus a total of 59 bins will only be needed in the LBP histogram. This reduces feature vectors of an eight sampling-point neighbourhood by 76.95% [6]. The motivation behind uniform LBPs is two-fold. Firstly, Ojala et al. [119] pointed out in experiments with different-textured images that uniform LBPs account for approximately 90% of all LBPs in an (8,1) neighbourhood and 70% in a (16,2) neighbourhood. Furthermore, Ahonen et al. [6] observed that in experiments with facial images, uniform LBPs account for 90.6% and 85.2% of all LBPs in an (8,1) and (8,2) neighbourhood, respectively. Secondly, it is indicated that uniform LBPs are less prone to noise and that fewer samples offer a reliable estimation of the distribution, thereby producing better recognition results in many applications [126].

4.1.1.2 Rotational-invariant LBP

When rotating an image in plane, the neighbouring pixels around the centre pixel, c, will be rotated in the same direction. However, this rotation results in a different LBP value for the centre pixel. To minimise this rotational effect each LBP code is circularly rotated to obtain its minimum value [29]. This rotational-invariant LBP operator can therefore be defined as [29]:

$$LBP^{ri}_{P,R}(x,y) = min\{ROR(LBP_{P,R}(x,y),k)|k\epsilon[0,P-1]\} \tag{4.5}$$

where $ROR(z,k)$ denotes the circular bit-wise right-shift rotation of bit sequence z by k steps. For example, the binary patterns 01011000, 00010110 and 01100001 all map to the minimum pattern 00010110.

One disadvantage of this operation is that the histogram of the rotational-invariant LBPs are only invariant to image rotations by angles $(a\frac{360}{P})^{\circ}$, $a = 0, 1, ..., P-1$. Thus, for an 8-bit code, it would only be invariant to 45° image rotations. Another disadvantage is that it loses directional information, which would be crucial in some applications.

4.1.1.3 Global versus local LBP histograms

In texture analysis the LBP histogram is often used to represent the global texture information [66]. Face images, as well as hand images, consist of a composition of micro-patterns that can effectively be described by LBP histograms. When computing a LBP histogram over an entire face image, only the occurrences of the micro-patterns will be encoded, without any indication about their locations [139]. The histogram therefore only represents the global information. To consider the shape information and include the locations of the micro-patterns, Shan et al. [139] divided face images into several equal regions, $R_0, R_1, ..., R_m$, where R denotes the region and m is the total number of regions. For each region an LBP histogram is computed that represents the local information. The LBP histograms are then concatenated to form a single spatially enhanced feature histogram, as shown in Figure 4.5.

Shan et al. [139] and Yang and Chen [169] have shown that spatially enhanced histogram features are more effective than global histogram features in facial-expression recognition

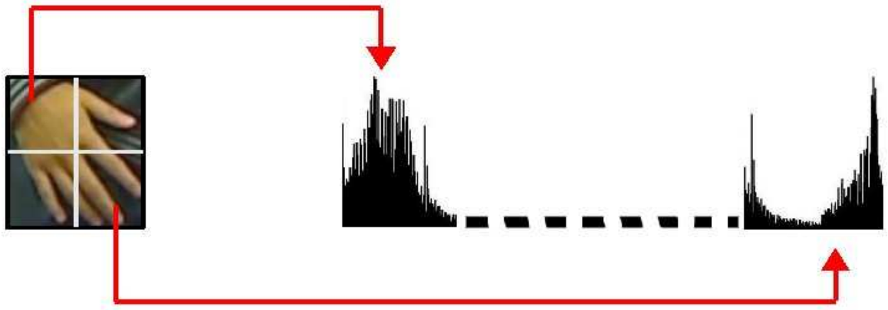


FIGURE 4.5: An example of the concatenation of a spatially enhanced feature histogram.

and facial recognition, respectively. Yang and Chen [169] further compared the LBP image (which consists of the LBP codes only) to the LBP histogram method and found the LBP images to be a good candidate for facial recognition with large variation.

Hrúz et al. [74] and Xiao et al. [166] applied spatially enhanced histogram features to the hand-shape recognition problem. However, they do not motivate why these features were used over the global histogram features and whether the LBP images could be used to address the problem. The current research therefore investigates whether LBP images can be used for hand detection in images. It also assesses how well spatially enhanced histogram features compare to global histogram features in terms of the hand-detection problem.

The success of LBP methods applied to various problems has inspired new research on several variants in the past decade. The local Gabor binary pattern was introduced by Zhang et al. [172]. Tan and Triggs [150] extended the LBP by allowing a three-valued quantisation of intensity difference and named it local ternary patterns. Fu and Wei [56] addressed the problem of noise by proposing the centralised LBP and Liao et al. [93] proposed the dominant LBP. Many more LBP variants have been proposed; however, the choice of a suitable variant depends on many factors such as the application, computational efficiency, discriminative power and robustness to illumination.

4.1.2 Random forests

Single decision trees have a high variance and low prediction accuracy. To maintain the advantages of decision trees while increasing the accuracy, random forests was proposed by Breiman [21] to decrease correlation among the trees by growing trees that employ a random-split selection on the input variables [91]. They consist of an ensemble of unpruned decision trees learned independently from a randomly selected subset of the training data [155].

By using an ensemble of decision trees, random forests have the ability to learn more than one class at a time by allowing each decision tree to vote on a class. In a classification approach the class receiving the maximum number of votes is determined to be the output class [20]. Regression in random forests is achieved by taking the average of the values across the leaves of all the trees. The random forest algorithm [21] is constructed as follows:

Random forests are formally defined as follows [21]:

$$F = \{f_1, f_2, ..., f_M\} \tag{4.6}$$

where M is the total number of trees in a forest and the m^{th} tree is denoted as:

$$f_m(x) = f(\mathbf{x}, \theta_m) : X \longrightarrow Y \tag{4.7}$$

where θ_m are independently distributed random samples and the input variable, $\mathbf{x}$, is the casted vote of a tree for the most popular class. While training the trees, random tests, $n \leq p$, are created for each decision node of a tree, where n is often set to the square root of p and p is the number of all input variables [91].

The best split is then selected for each node, j, of the tree, according to a certain quality measurement score, such as the entropy or Gini score that scores the potential information gain, $\triangle H$, as follows [91]:

$$\triangle H = \frac{|I_l|}{-|I_l| + |I_r|} H(I_l) - \frac{|I_r|}{|I_l| + |I_r|} H(I_r), \tag{4.8}$$

where $H(I)$ is the node score, I_r is the right and I_l is the left subset of the training data. Furthermore, the following measurement is used for the entropy for node j:

$$H(I) = -\sum_{i=1}^{k} p_i^j log(p_i^j) \tag{4.9}$$

```
Given:
  1. The training feature set $V = \{(\mathbf{x_1}, y_1), (\mathbf{x_2}, y_2), ..., (\mathbf{x_m}, y_m)\}$, where
     $y_i \epsilon Y = \{-1, 1\}$;
  2. The forest size, $M$;
  3. The maximum depth, $D$, of the decision trees.
Then:
for $m \leftarrow 1$ to $M$ do
    1. Construct a bootstrap sample with replacement training samples
       $\{(\mathbf{x_1}, y_1), (\mathbf{x_2}, y_2), ..., (\mathbf{x_n}, y_n)\}$ from the learning set $\mathbf{x}$;
    2. Grow a random forest tree $T_m$ on $\{(\mathbf{x_1}, y_1), (\mathbf{x_2}, y_2), ..., (\mathbf{x_n}, y_n)\}$ by
       recursively repeating the following steps and applying the modification at
       each node;
    while there exist non-pure nodes and depth $d < D$ do
        1. Randomly select $n$ variables from the input variables to use as possible
           split variables;
        2. Calculate potential information gain for each of the selected variables;
        3. Pick the best split position and split the node into two child nodes;
    end
end
Output the random forest, $F$.
$F$ is used to make a prediction as follows:
  1. The prediction of each tree $f_m$ is calculated.
  2. In a classification strategy, the output is the majority vote over all trees.
  3. In a regression strategy, the output is the average response value over all
     trees.
```

Algorithm 1: The algorithm for constructing a random forest

or the Gini score:

$$H(I) = -\sum_{i=1}^{k} p_i^j (1 - p_i^j) \tag{4.10}$$

where p_i^j is the label density of class i in node j [91]. The training continues recursively until no further information gain is obtained or a maximum depth is reached.

4.1.2.1 Error, strength and correlation using out-of-bag-error

During training, a new bootstrap sample is constructed for each tree by sub-sampling *with replacement*[1] from the original training set. These samples that are not included during the training of a tree is referred to as out-of-bag (OOB) samples of that tree [91]. In addition, the OOB samples are used to estimate the performance of the split and compute what is called the OOB error (OOBE) of the tree and the forest [91].

When computing the OOBE, bagging (bootstrap aggregation) is used in tandem with random feature selection [21]. Bagging provides two advantages: first, it improves accuracy when random features are used; second, it can be used to return ongoing estimates of the generalisation error of the random forest as well as estimates for the strength and correlation between trees [21]. Empirical estimates have shown that OOBE estimates have the same accuracy as using a test set that is the same size as the training set [21]. Furthermore, the estimates on strength and correlation are helpful in understanding the classification accuracy and ways to improve the random forest.

This error calculation is an unbiased estimate of the generalisation and therefore can be used to obtain better estimates of how well it will do on unseen data [20]. If the training and testing data have a similar distribution, then the performance of the OOB prediction can be very accurate [20]. Unlike n-fold cross validation, this operation is part of the learning strategy in random forests and therefore inherently provided [91].

[1]It means that some data points may be randomly repeated.

4.1.2.2 Variable importance

In random forest, all variables are ranked according to their importance, referred to as variable importance. The measures of variable importance are returned by the random forest and can used to perform variable selection. These measures are based on the misclassification rate when the variable values in the OOB samples are randomly permuted.

When using the Gini index to determine the node purity, the measure is known as the Gini importance. At every node split done on a variable, the Gini index for the two descendent nodes are decreased. The sum of all the Gini decreases for a given variable, which is normalised by the number of trees, provides a fast variable importance that is consistent with the permutation-based variable importance.

4.1.2.3 Testing

The estimated probability for predicting class c for a sample can be defined as:

$$p(c|\mathbf{x}) = \frac{1}{M}\sum_{m=1}^{M} p_m(c|\mathbf{x}) \tag{4.11}$$

where $p_m(c|\mathbf{x})$ is the estimated density of class labels of the leaf nodes of the M^{th} tree [91]. The final decision function of the forest is thus defined as:

$$D(\mathbf{x}) = arg_{c\epsilon y} max\ p(c|\mathbf{x}) \tag{4.12}$$

In addition, the classification margin of a labelled sample ($\mathbf{x}$,y) is defined as:

$$mg(\mathbf{x}, y) = p(y|\mathbf{x}) - max_{\substack{c\epsilon y \\ c\neq y}} p(c|\mathbf{x}) \tag{4.13}$$

The margin measures the extent to which the average vote for any other class is exceeded by the average number of votes at ($\mathbf{x}$,y) for the right class [21]. Thus, for a larger margin, the higher the confidence would be in the classification tasks. Derived from this, the generalisation error is given by [21]:

$$GE = p(\mathbf{x}, y)(mg(\mathbf{x}, y) < 0) \tag{4.14}$$

where ($\mathbf{x}$,y) indicates the probability is over the ($\mathbf{x}$,y) space. The error of a forest depends on the correlation between pairs of trees in the forest, $\bar{p}$, and the strength of the individual trees in the forest, s. Breiman [21] states that although random forests do not overfit when more trees are added, it produces a limit on the generalisation-error that is upper bounded by [21]:

$$GE \leq \bar{p}\frac{1 - s^2}{s^2} \tag{4.15}$$

where,

$$s = E_{(\mathbf{x},y)} mg(\mathbf{x}, y) \tag{4.16}$$

The strength, s, relates to the average accuracy of each tree and the correlation is the measurement of tree diversity [12]. Increasing the correlation among trees lowers their diversity and increases the forest error rate. However, increasing the strength of the individual trees decreases the forest error rate. Thus, the lower the error rate, the stronger the classifier would be [121].

An additional property concerning random forests is that it can be used to determine the probability between any two data points [20]. In this context, the probability refers to the similarity between the points. To obtain these probabilities, the random forest algorithm traverses the tree, counting the number of times the points end up at the same leaf node and divides this leaf count by the total number of trees [20]. This results in probability values between one and zero, where one represents exactly similar and zero represents very dissimilar. The probability is useful in identifying the outliers as well as to cluster points together.

4.1.2.4 Parameter selection

When constructing random forests, only two parameters are required, namely the number of random split variables, L, and the number of trees, M [12].

The random split variables should be chosen according to a purity measure, such as the deviance for classification. For smaller number of split variables, the diversity of trees will increase; however, when the number of split variables is too small, this may decrease the accuracy of a tree [12]. Breiman [21] therefore suggests that the value of L that is equal to the square root of the number of inputs should be used for classification tasks.

When increasing the number of trees, the generalisation error would approach a limit asymptotically [21]. Therefore, the number of trees should be selected and increased until the OOB error no longer improves significantly [12]. Oshiro et al. [121] showed that beyond this point only the computational cost would increase with no performance gain. They also suggest that the more features there are, the more trees would be needed.

4.1.2.5 Strengths and weaknesses of random forests

Random forests are powerful classifiers that inherit the many benefits of decision trees, such as the ability to handle missing data; the ability to handle categorical and numerical values without the need to normalise these values; and methods to find variables that are important for prediction. Random forests also overcome the limitations of decision trees such as: not overfitting the data; not needing to prune the trees in order to generalise; and an improved prediction accuracy compared to single-decision trees.

However, Genuer et al showed that the performance of random forests may be limited when a small number of random-split variables are used or a large number of noisy variables are present [59].

Overall, random forests have been used for a diverse range of applications and have displayed high levels of predictive accuracy compared to other classification techniques. Furthermore, it is fast to train, fast to predict and only require a few parameters to tune it [21].

4.1.3 Support vector machines

There has been growing interest in using SVMs for pattern-recognition problems. SVMs are derived from statistical learning theory and are applied as a machine learning tool that inherently classifies data into two classes. SVMs offer several advantages such as dealing with high-dimensional feature vectors without affecting the training time; being memory efficient by only using a subset of training points in the decision function; and using different kernel functions that offer both power and flexibility [164]. The default kernel is the linear kernel that can easily be substituted with the radial basis function, sigmoid, polynomial or other more recent kernels that allow features to be separated more clearly in a given classification problem. Alternative kernels allow SVMs to solve non-linear classification problems using linear classification techniques.

In principle, an SVM is a mathematical algorithm that aims to maximise a mathematical function given a set of data points [118]. Consider a set of data points that consists of two classes: the theory of SVMs suggests that it is possible to find a boundary that can separate the two classes. Furthermore, consider a training set of data points, N, that is represented by $S = \{(x_1, y_1), (x_2, y_2), ..., (x_N, y_N)\}$, where each x_i, with $i = 1, 2, ..., N$, is a data point in $\mathbb{R}^n$ and each $y_i \epsilon \{-1, 1\}$ is the corresponding classification label for x_i such that the data points are divided into positive and negative classes. Moreover, suppose that the positive class, $S^+ = \{x_i | y_i = 1\}$, and negative class, $S^- = \{x_i | y_i = -1\}$, are linearly separable in $\mathbb{R}^n$, such that at least one boundary can be formed between the two [118], then this boundary (see Figure 4.6(a)), also referred to as the decision boundary, is formed by training an SVM on S.

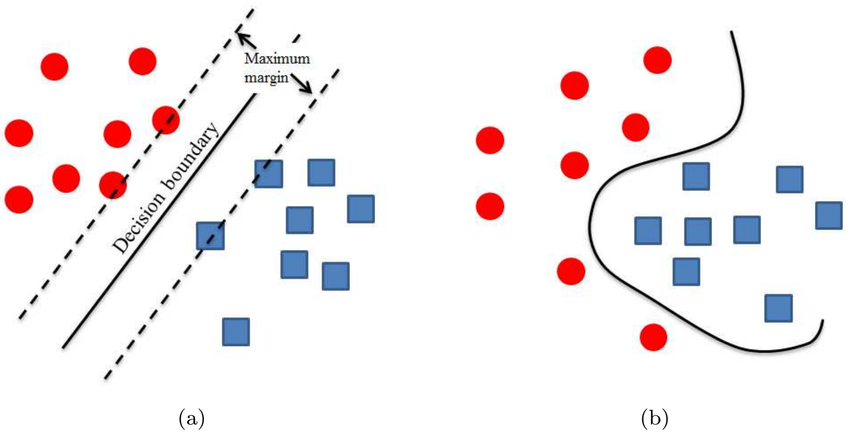


FIGURE 4.6: (a) Linear classification; (b) non-linear classification [170].

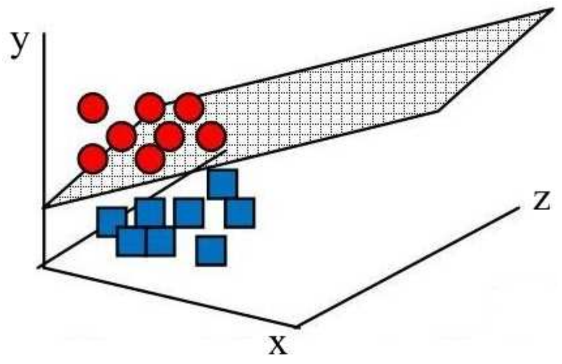


FIGURE 4.7: Linear classification of a hyperplane [170].

In a higher dimensional space, the boundary is regarded as a geometrical concept of a plane and is generally referred to as a hyperplane. The hyperplane, as depicted in Figure 4.7, is defined by the following equation:

$$f(x) = \mathbf{w} \cdot \mathbf{x} + b = 0; \ \mathbf{w} \epsilon \mathbb{R}^n, \ b \epsilon \mathbb{R} \tag{4.17}$$

where **w** is the normal vector, **x** is the feature set and b is the interim term. The normal vector, **w**, of the hyperplane is defined as a linear combination of data points, x_i, with weights, α_i, and formally expressed as:

$$\mathbf{w} = \sum_{1 \leq i \leq N} \alpha_i x_i y_i \tag{4.18}$$

If the data points are linearly separable, then two hyperplanes can be selected such that they separate the data and there are no points between them. These hyperplanes can be described by the following equations:

$$\begin{aligned} \mathbf{w} \cdot x_i + b &= +1 \\ \mathbf{w} \cdot x_i + b &= -1 \end{aligned} \tag{4.19}$$

Midway between these hyperplanes, a decision boundary can be defined as

$$\mathbf{w} \cdot x_i + b = 0 \tag{4.20}$$

Moreover, the distance, d, between the margin and the decision boundary can be expressed as:

$$d = \frac{2}{||\mathbf{w}||} \tag{4.21}$$

Two factors determine the selection of a hyperplane: firstly, it should separate the data points clearly, and secondly, it should have the maximum distance to the nearest data point from both classes. This distance is referred to as the margin and the data points that are situated closest to the hyperplane are referred to as the support vectors. When the separation between the two classes is greater, it is necessary to find the maximum margin, because it would allow the SVM to classify new data points more accurately. To determine this hyperplane, two requirements need to be met:

1. The first is that all training data points should be classified correctly [170]. Thus, **w** and b should be estimated such that:

$$y_i(\mathbf{w} \cdot x_i + b) \leq -1 \; for \; y_i = -1 \tag{4.22}$$

and

$$y_i(\mathbf{w} \cdot x_i + b) \geq 1 \; for \; y_i = +1 \tag{4.23}$$

Combining these two equations gives:

$$y_i(\mathbf{w} \cdot x_i + b) - 1 \geq 0, \; \forall \; i = 0, 1, 2, ..., N \tag{4.24}$$

2. The second requirement is that the margins should be as large as possible. Maximising the distance in equation 4.21 is the same as minimising $\frac{||\mathbf{w}||}{2}$. This results in minimising $f(\mathbf{w}) = \frac{1}{2}||\mathbf{w}||^2$. The optimal hyperplane can therefore be found by solving the optimisation problem, which is defined as:

Minimise

$$\frac{1}{2}||\mathbf{w}||^2 \tag{4.25}$$

Subject to

$$y_i(\mathbf{w} \cdot x_i + b) - 1 \geq 0, \; \forall \; i = 0, 1, 2, ..., N \tag{4.26}$$

Given the Langrange multipliers $\alpha_1, \alpha_2, ..., \alpha_N \geq 0$ and the saddle point of the Langrange function:

$$L(\mathbf{w}, b, \alpha) = \frac{1}{2}||\mathbf{w}||^2 - \sum_{i=1}^{N} \alpha_i (y_i(\mathbf{w} \cdot x_i + b) - 1) \tag{4.27}$$

The optimisation problem can be translated to:

Maximise

$$\sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{N} \alpha_i \alpha_j y_i y_j (x_i, x_j) \tag{4.28}$$

Subject to

$$\sum_{i=1}^{N} \alpha_i y_i = 0 \text{ and } \alpha_i \geq 0, i = 1, 2, ..., N \tag{4.29}$$

The selected hyperplane is referred to as the maximum margin hyperplane or the optimal hyperplane. Under this formulation, the optimal hyperplane discriminant function can be defined as:

$$f(x) = \sum_{i \epsilon S} \alpha_i y_i (x_i x) + b \tag{4.30}$$

where S is the subset of support vectors that correspond to positive Langrange multipliers.

In the classification of linear problems, a linear hyperplane is determined with a maximum margin that separates the data points. In the classification of non-linear problems, the data points are instead mapped onto a higher-dimensional space, referred to as the feature space. In this space, an optimal hyperplane that separates the data points linearly can be determined.

In reality, more complex structures are required to find a decision hyperplane in non-linear problems. As illustrated in Figure 4.6(b), the data points are unevenly distributed in these cases and non-separable compared to those in Figure 4.6(a).

In these problems, the classes are not linearly separable and therefore the first constraint (see equation 4.24) cannot be satisfied. To overcome these problems, a cost function can be formulated that combines the margin maximisation and the minimisation of error criteria. The formulation is achieved using a set of variables, ξ, where the cost function can be formulated as:

Minimise$_{\mathbf{w},b,\xi}$

$$\frac{1}{2}||\mathbf{w}||^2 + C \cdot \Sigma_{i=1}^{N} \xi_i \tag{4.31}$$

Subject to

$$y_i(\mathbf{w} \cdot x_i + b) \geq 1 - \xi_i \tag{4.32}$$

where

$$\xi_i \geq 0 \text{ and } C \text{ are constants} \tag{4.33}$$

The parameter, C, determines the trade-off between the margin maximisation and the amount of error to be tolerated. According to Mecer's theorem [152], the dot product of the vectors in the mapping space can be equally formed as a function of dot products of the corresponding vectors in the current space [154]. This equivalence can be expressed as follows:

$$\begin{aligned} K(x_i, x_j) &= \phi(x_i) \cdot \phi(x_j) \\ &= (x_i, x_i^2) \cdot (x_j, x_j^2) \\ &= x_i x_j + x_i^2 x_j^2 \\ &= x_i \cdot x_j + (x_i, x_j)^2 \end{aligned} \tag{4.34}$$

where $K(x_i, x_j)$ represents the kernel function. This equation holds true if — and only if — the following condition holds true for any function, h:

$$\int h(\mathbf{x})^2 d\mathbf{x} \text{ is finite } \Rightarrow \int K(\mathbf{x}, \mathbf{y}) h(\mathbf{x}) h(\mathbf{y}) d\mathbf{x} d\mathbf{y} \geq 0 \tag{4.35}$$

By selecting an appropriate kernel, any set of data points can be linearly separated in a higher-dimensional space without knowing the explicit form of ϕ. The dual optimisation problem can therefore be formulated as:

Maximise

$$\sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{N} \alpha_i \alpha_j y_i y_j K(x_i, x_j) \tag{4.36}$$

Subject to

$$\sum_{i=1}^{N} \alpha_i y_i = 0 \text{ and } \alpha_i \geq 0, \text{ where } i = 1, 2, ..., N \tag{4.37}$$

Hence, it should be noted that a complex curve would not be suitable to separate data and that an optimal hyperplane in the feature space can be found that would separate the data clearly and would allow an SVM to classify new test data accurately. The decision function thus becomes:

$$f(x) = \sum_{i \epsilon S} \alpha_i y_i K(x_i x) + b \tag{4.38}$$

where S are the support vectors, K is the kernel function and b is the interim term.

4.1.3.1 Kernel functions

In non-linear problems, an optimal hyperplane that separates the classes is required. This is achieved by using one of the different kernel functions that map data from a current space onto a higher-dimensional feature space. Following Mecer's theorem [152], the SVM makes use of four basic kernels for training and classification, where γ, d and r are kernel parameters and $\gamma > 0$ [75]:

- Linear: $K(x_i, x_j) = (x_i)^T \cdot (x_j)$
- Polynomial: $K(x_i, x_j) = (\gamma(x_i)^T \cdot (x_j) + r)^d$
- Sigmoid: $K(x_i, x_j) = \tanh(\gamma \cdot (x_i)^T \cdot (x_j) + r)$
- Radial basis function: $K(x_i, x_j) = exp(-\gamma \cdot ||x_i - x_j||^2)$

Selecting a suitable kernel is essential because it influences the prediction capabilities of SVMs. Although research over the past few years aimed at determining an appropriate kernel for a given problem [96], no standard method exists for finding the most appropriate kernel [170]. Selecting an appropriate kernel is therefore based on trial and error [37].

By employing the detection phase, the object to be tracked can be validated to determine if it is a hand. In Chapter 6 it will be determined which of the components will be the most suitable in validating the object.

4.2 Tracking phase

In this section the components in the tracking phase, implemented in the second DSR cycle (see Figure 4.8), is discussed. These components form part of the algorithm that tracks the right and left hand independently in both constrained and unconstrained environments.

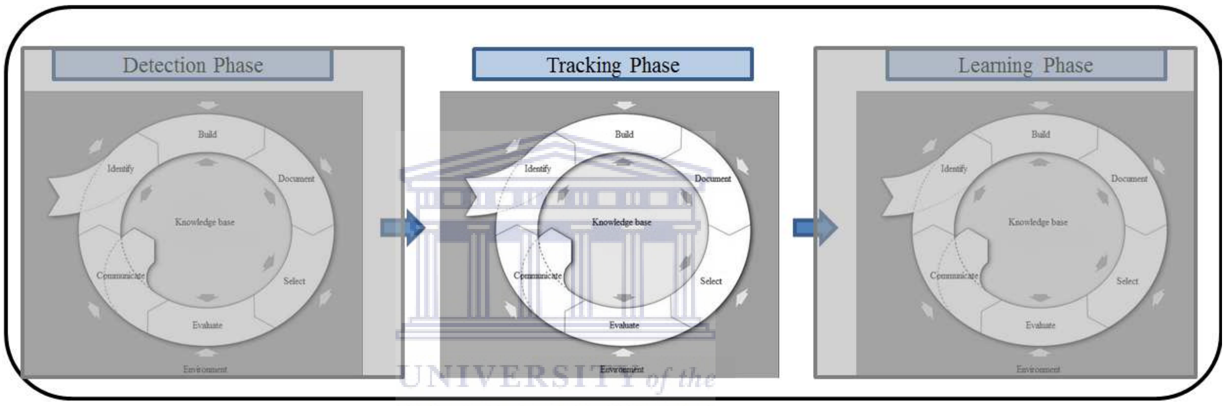


FIGURE 4.8: The second DSR cycle, consisting of the tracking phase.

In order to successfully track hand motion, the hands need to be clearly segmented from the rest of the scene. Since hands are articulated objects, it is a challenging task to employ geometric models to track hands due to the complexity involved in minimising the error cost between the hand and the model projection. A more cost-effective approach to hand-tracking that is invariant to the articulated nature of hands is to use skin colour. Combining a reliable skin segmentation with an effective data-association technique results in an efficient and robust hand-tracking algorithm. This hand-tracking algorithm forms a prerequisite for distinguishing different hand gestures in SASL using hand-shape recognition. In the following subsections the components that were utilised to form the proposed tracking algorithm are discussed.

4.2.1 Skin segmentation

Scientific studies [129] have shown that skin-colour diversity in South African and sub-Saharan African populations is the highest in the world. This diverse range of skin-colour tones and its variations under different illumination conditions requires that reliable skin segmentation should be dynamically adaptive. Although many skin-colour segmentation algorithms have been proposed [19, 82, 149], they can only handle slight changes in illumination and are negatively affected by background objects, resulting in a higher false-positive rate.

The main advantages of using colour to segment skin is that it is computationally inexpensive as a low-level feature and robust against variations in rotations, scaling and partial occlusions [80].

The aim of skin-colour segmentation is to separate pixels in an image into either skin-coloured or non-skin-coloured pixels. The simplest method to classify pixels as skin colour is to apply a threshold, representing a confined neighbourhood in a given colour space, in which skin colour is considered to be clustered. However, this method is often not sufficient, because skin colour may vary under different lighting conditions.

Skin segmentation can be regarded as a classification problem. To identify skin colour, three basic steps are involved [47, 80, 113]:

1. A suitable colour space should be selected to represent the pixels in an image.
2. Skin and non-skin pixels should be modelled using an appropriate classification algorithm.
3. Pixels should then be classified individually as either skin or non-skin pixels.

Many existing skin segmentation algorithms that follow these steps perform well for only a limited set of skin-colour tones and illumination conditions. In order to handle environmental changes and changes in illumination conditions, an additional step is required between steps 2 and 3:

- Before classifying pixels, colour constancy should be applied to the image or a dynamic adaptation technique should be used.

In colour-constancy techniques, colours in an image are corrected and the illumination is estimated [3]. In dynamic-adaptation techniques, skin-colour models are adapted to changes in a scene. A popular dynamic-adaptation technique is to sample skin colour from the face of an individual, based on the fact that the face and body shares a similar colour distribution [3]. However, this approach suffers from several drawbacks, as it is negatively affected by the lips, eyes, spectacle frames, facial hair and shadows [3].

As an alternative, a unique dynamic skin-invariant modelling framework is proposed in Section 4.2.1.3 that segments any skin-colour tone under varying backgrounds and illumination conditions. The framework presents two important properties: firstly, it does not require any training beforehand and, secondly, it samples a closer skin-colour distribution by using the area around the nose. Given that the framework was aimed at segmenting skin-coloured areas in a SASL application, it is restricted to images containing a face. It can, however, be easily extended to images that do not contain a face by training models such as Gaussian, Bayesian or neural networks using the sampled skin-colour distributions.

In summary, the algorithm consists of four steps: (i) face detection is applied using the well-known Haar-like features and the AdaBoost algorithm developed by Viola and Jones [158]; (ii) the default RGB colour space is transformed to the hue, saturation and value (HSV) colour space for efficiency and simplicity; (iii) the region around the nose is selected to sample a skin-colour distribution that is used to compile a colour histogram model; and (iv) the model is used to segment skin-coloured pixels in an image. In the following subsections, this algorithm will be further explored and explained.

4.2.1.1 Face detection

The Viola and Jones face-detection algorithm [158] is a tree-based technique that employs Haar-feature classifiers to build a boosted rejection cascade using AdaBoost. In this way, a positive detection rate is achieved by using a low rejection rate multi-tree classifier at every node in the cascade [79]. Their algorithm is designed to remove less-promising regions and focus more on areas that are likely to contain a face. A more comprehensive breakdown of their algorithm can be found in Viola and Jones [158].

To determine the effectiveness of this algorithm, Achmed [1] evaluated the method on 1 047 randomly selected images acquired from the internet and other sources. The results demonstrated an 89% detection rate on frontal-view faces, which is comparable to the results achieved by Viola and Jones [158].

4.2.1.2 Does a suitable colour space exist?

Many researchers [31, 45, 84] use colour-space transformations to segment skin colour for the following reasons: firstly, it is assumed that by applying a certain colour-space transformation, the separation between skin-colour and non-skin-colour pixels can be defined more clearly and, secondly, it is assumed that illumination invariance can be achieved.

In order to ascertain whether these assumptions hold, a number of studies [80, 141, 171] have investigated the effectiveness of colour-space transformation for the purpose of skin-colour detection. Based on the above assumptions, these studies have shown that colour-space transformations do not provide significant performance improvements when detecting skin-coloured pixels [80, 141, 171]. Furthermore, it was shown that by discarding the intensity component in a given colour space, the discrimination between skin-colour and non-skin-colour pixels does not improve. However, Kakumanu et al. [80] suggest that the selection of colour-space transformation should depend on the methodology and application for skin segmentation. Moreover, it is also suggested by Kakumanu et al. [80] and Vezhnevets et al. [157] that it should rely on the format in which an image is obtained, as well as the need for a certain colour space in post-processing steps. In addition, Shin et al. [141] and Zarit et al. [171] have suggested that it may support a better generalisation of training data in a classification process. Therefore, the question of whether a suitable colour space should be employed is left unanswered. Although many researchers [31, 45, 84] employ a particular colour space, their choice is not justified in their research. In addition, they do not justify that their chosen colour space is the optimal one for identifying skin-colour pixels.

Fleck et al. [53] argued that in the HSV colourspace the hue component has a restricted range on the human skin colour, which is made up of carotene, haemoglobin and melanin. Carotene consists of a distinctive yellow-orange colour that is mostly found in the palms and soles. Haemoglobin is responsible for carrying oxygen in red blood cells and forms the pink-red colour in skin. The amount and type of melanin is the primary determinant of skin colour. It consists of two types, namely eumelanin and pheomelanin. Eumelanin has a very dark-brown colour, whereas pheomelanin has a red colour. According to Fleck et al. [53] and other researchers [39, 40, 47], a combination of these colours is easily distinguishable by the hue constituent of the HSV colour space.

It is beyond the scope of this research to prove that an optimal colour space exists; however, it can be concluded that the HSV colour space is suitable to segment different skin-colour tones.

4.2.1.3 Dynamic skin model

Considering that the face is elliptic, once the face is located the dimensions are retrieved providing the minor axis, M_1, and the major axis, M_2. Given that the proportions of the face exhibit a symmetrical property with regard to the dimensions of the face, the nose can be determined by halving both M_1 and M_2. Next, the image is transformed from the RGB colour space to the HSV colour space. The area proposed as the best area to provide a closer skin-colour approximation is obtained by applying a radius of $M_1/5$ around the nose.

This method however only applies to frontal-view faces; for non-frontal-view faces a trained model should be used. This area is used as the region of interest to generate a 2D colour histogram model. The region of interest is used to extract the skin-colour distribution, which is quantised into a set number of bins, where each bin in the histogram holds the number of pixels per colour. Normalising the histogram results in a probability of a pixel belonging to an individual's skin colour. This is followed by using the histogram model to apply a back projection that operates by assigning a probability value to each pixel in an image.

Given that $P(S)$ is the probability that a pixel is skin and C is the colour of a pixel, then the probability that a pixel is a skin colour is given by [157]:

$$P(S|C) = \frac{P(S)}{P(C)} P(C|S) \tag{4.39}$$

where $P(C|S)$ is the probability of assigning a value to C if the pixel is skin. This back projection method results in a probability map in which lighter pixels have higher probabilities and darker pixels have lower probabilities, as shown in Figure 4.9.

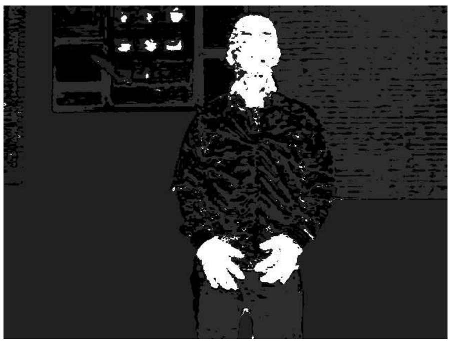


FIGURE 4.9: The probability map where lighter pixels have higher probabilities and darker pixels have lower probabilities.

To retain the most probable skin-coloured pixels and reduce pixels with lower-probability values, a threshold equal to 200 is applied, resulting in a binary image where skin-colour and non-skin-colour pixels are represented by a value of 255 and 0, respectively.

4.2.2 Connected-components labelling

One of the goals of detecting objects in an image is to first identify possible object locations. These locations can be identified by segmenting object regions using connected-components labelling. Connected-components labelling is a sequential two-pass algorithm that connects pixels in either binary or greyscale images to form segmented regions, referred to as components. In this section, connected-components labelling on binary images is discussed.

The algorithm operates by assigning a set of pixels into components using the level of its pixel connectivity, and thereafter labelling each pixel according to the component it was assigned. The algorithm consists of two types of labelling, namely, 4-connectivity or 8-connectivity. The 4-connectivity labelling mask only evaluates the horizontal and vertical neighbouring pixels around a given point, whereas, the 8-connectivity labelling mask includes the diagonal neighbouring pixels. In this research, the 8-connectivity labelling mask will be used, thereby searching for connected pixels in each direction and reducing the amount of noise.

Given a binary image consisting of skin-coloured and non-skin-coloured pixels, in the first pass the mask moves along the rows from the top left to the bottom right of the image. For each skin-coloured pixel a temporary label is assigned based on the neighbouring pixels' values that have already been processed. If none of the four top-left neighbouring pixels (pixels that have been passed) is a skin-coloured pixel, then the centre pixel is assigned a new label; however, if one of the four neighbouring pixels is a skin-coloured pixel, then its label is assigned to the centre pixel. Furthermore, if the centre pixel is a skin-coloured pixel and contains two ore more neighbouring skin-coloured pixels that have different labels, then the labels of these neighbouring pixels are stored as being equivalent. After the first pass, equivalence classes are determined using these equivalences, where a unique label is assigned to each class. During the second pass, each temporary label is replaced by the label of its corresponding equivalence class [4].

Once the two-pass algorithm has completed, the connected components can be analysed. This additional step makes it possible to use prior knowledge and the components' size to eliminate components considered to be noise.

4.2.3 Independent hand-tracking

A novel data-association method that enables the tracking of multiple skin-coloured objects entering a scene over time was proposed by Argyros and Lourakis [9]. This section discusses how this data-association method can be adapted and extended in a novel algorithm. In contrast to their data-association method, this algorithm can be used to track two objects with a similar appearance in unconstrained environments.

Traditionally, multiple object-tracking methods often rely on Kalman filters and their extensions. This is particularly the case if the object dynamics or observations are of a Gaussian nature; however, in the case of a non-Gaussian distribution, the Kalman filter assumptions would not suffice [9]. Object tracking is often categorised as either a Bayesian or non-Bayesian method. Single object tracking usually relies on powerful object models. In multiple object tracking, in some cases multiple instances of single object tracking are applied [76], while in other cases particle filters are used to track multiple objects simultaneously [144]. The data-association method by Argyros and Lourakis [9] was proposed to apply multiple object tracking in a non-Bayesian framework and is modified as follows:

Assuming that a binary image is obtained in which a right and left hand has been isolated as skin-coloured clusters, the method assigns a set of object hypotheses corresponding to these skin-coloured clusters and associates them with time. The aim of this association is to assign a unique label to the right and left hand, and associate pixel information according to each hand across all frames. In the following subsections a more detailed description of the data-association method is discussed.

4.2.3.1 Generating object hypotheses

Assume that at time t, N skin-coloured clusters exist and where each s_i, with $1 \leq i \leq N$, corresponds to a skin-coloured cluster. In addition, let the number of hands, M, at time t be $M \leq 2$ and let the set of skin pixels that image the j-th object be o_j with $1 \leq j \leq M$. Suppose that the spatial distribution of the pixels representing a skin-coloured cluster is approximated by an ellipse, then the ellipse model of the object can be denoted by $h_i = h_i(c_{x_i}, c_{y_i}, \alpha_i, \beta_i, \theta_i)$, where the point (c_{x_i}, c_{y_i}) is its centroid, α_i is the length of the major axis, β_i is the length of the minor axis and θ_i is its orientation on the image plane. Moreover, let $S = \cup_{i=1}^{N} s_i$, $O = \cup_{j=1}^{M} o_j$ and $H = \cup_{j=1}^{M} h_j$ denote the union of skin-coloured clusters, object hypotheses and ellipses, respectively. The tracking of the hands is therefore determined by the relationship between the skin-coloured clusters (s_i) and the object hypothesis models (h_j) in time. Therefore, by associating object hypotheses with skin-coloured clusters across time, multiple clusters can be tracked even when occlusion occurs.

It should however be noted that the correspondence between skin-coloured clusters and object hypotheses are not necessarily one-to-one. For example, when the right and left hand cross and occlude each another, each hand is a different object, but appears as a single skin-coloured cluster. This data-association method therefore assumes that an object hypothesis may correspond to either one or more clusters and, in a similar manner, one cluster may correspond to one or more object hypotheses.

When associating an object hypothesis with a skin-coloured cluster, represented by an ellipse, the distance of a pixel from the cluster to the ellipse is used to determine if the ellipse is associated with the cluster or not. The distance, $D(p,h)$, from a point, $p = p(x,y)$, to an ellipse, $h(c_x, c_y, \alpha, \beta, \theta)$, is defined as follows [9]:

$$D(p,h) = \sqrt{\vec{V}^T * \vec{V}} \tag{4.40}$$

where

$$\vec{V} = \begin{bmatrix} cos(\theta) & -sin(\theta) \\ sin(\theta) & cos(\theta) \end{bmatrix} (\frac{x - x_c}{\alpha}, \frac{y - y_c}{\beta}), \qquad \vec{V}^T = \text{The transpose of matrix } \vec{V}$$

It then follows that if the distance is less than one, equal to one or greater than one, then the given point exists inside, on or outside the ellipse, respectively. Thus, if the distance is less than or equal to one, then all pixels belonging to a cluster support the existence of the object hypothesis represented by the ellipse at time t and that the object hypothesis predicts the cluster at time $t + 1$. If the distance of all pixels belonging to a cluster is greater than one, then none of the object hypotheses supports the existence of this cluster.

When an individual communicates in sign language the default pose would be to naturally hold the right and left hand on the right and left side of the body, respectively. In the initial frame of the video, an object hypothesis is assigned to each hand, one for the right hand and one for the left. To directly derive the parameters for these initial object hypotheses, the statistics for the distribution of pixels belonging to a given cluster is used, where the centre of the ellipse is equal to the centre of the skin-coloured cluster and the rest of the parameters are computed from the covariance matrix, Σ, of the bivariate distribution of the cluster pixels' location.

It can therefore be shown that if the pixel distribution is represented by

$$\Sigma = \begin{bmatrix} \sigma_{xx} & \sigma_{xy} \\ \sigma_{xy} & \sigma_{yy} \end{bmatrix} \tag{4.41}$$

then the rest of the ellipse parameters can be defined as [9]:

$$\alpha = \sqrt{\lambda_1},\ \beta = \sqrt{\lambda_2} \text{ and } \theta = tan^{-1}(\frac{-\sigma_{xy}}{\lambda_1 - \sigma_{yy}}) \tag{4.42}$$

where

$$\lambda_1 = \frac{\sigma_{xx} + \sigma_{yy} + \Lambda}{2},\ \lambda_2 = \frac{\sigma_{xx} + \sigma_{yy} - \Lambda}{2} \tag{4.43}$$

and

$$\Lambda = \sqrt{(\sigma_{xx} - \sigma_{yy})^2 - 4{\sigma_{xy}}^2} \tag{4.44}$$

4.2.3.2 Tracking-object hypotheses

After generating the initial object hypotheses and assigning an ellipse model to each hand, the tracking of the hands involves associating the hand pixels to the respective object hypothesis. This association is governed by three rules [9]:

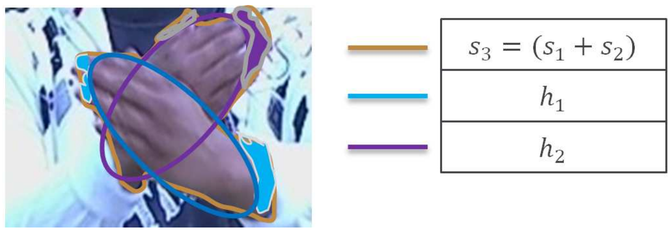
1. If a skin-coloured pixel of a cluster exists within an ellipse for a given object hypothesis, then that pixel is considered to belong to this object hypothesis.

2. If a skin-coloured pixel of a cluster exists outside both ellipses corresponding to each object hypothesis, then that pixel is assigned to the object hypothesis that is closest to it.

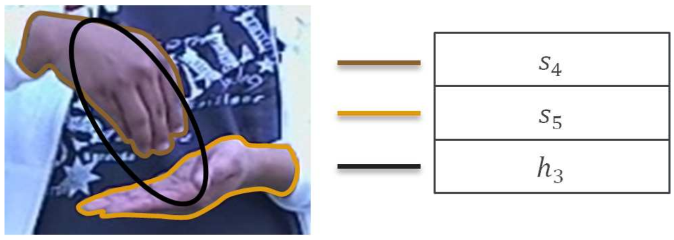
To deal with scenarios where an object hypothesis belongs to more that one cluster, the following rule is applied:

3. If an object hypothesis exists that is associated with only one cluster and at the same time is not associated with any other cluster, then the object hypothesis is assigned to that cluster. Otherwise the object hypothesis is assigned to the cluster with which it shares the largest number of skin-coloured pixels.

An illustrative example of these rules is presented in Figure 4.10. If, for example, in Figure 4.10(a), both hands (clusters, s_1 and s_2) cross each other and occlusion occurs, then it may appear that a single cluster is formed, $s_3(s_3 = s_1 + s_2)$. In this case, according to rule (1), all skin-coloured pixels located inside ellipse model h_1 will be assigned to it. In a similar way, all skin-coloured pixels located inside ellipse model h_2 will be



(a) An example of more than one object hypothesis associated to more than one cluster.



(b) An example of one object hypothesis associated to more than one cluster.

FIGURE 4.10: Examples of object-hypotheses association.

assigned to it. It should, however, be noted that those pixels that are located at the intersection of h_1 and h_2 will be assigned to both object hypotheses represented by the ellipse models. According to rule (2), all skin-coloured pixels that are not located inside either ellipse model but form part of s_3 will be assigned to the closest object hypothesis. These association rules allow for the hands to be tracked correctly after occlusion occurs. In another example (see Figure 4.10(b)), if the hands are apart, forming two separate clusters, s_4 and s_5, but have an object hypothesis, h_3, that is associated with both clusters, then according to rule (3), h_3 will be assigned to cluster s_4, with which it shares the largest number of skin-coloured pixels.

After associating all skin pixels with their respective object hypotheses, the parameters for the ellipse models are re-estimated based on the statistics of the skin-coloured pixels assigned to them.

4.2.3.3 Predicting of object hypotheses

The way in which multiple objects are tracked over a period of time is the fundamental property of the data-association method. Under the assumption that the immediate future can be predicted using the immediate past, the location of the hands at time t can be predicted using a simple linear function based on their locations at time $t-1$ and $t-2$. Therefore, by using the centre points of the ellipse models in the previous two frames and keeping all other parameters the same, the location of the hands can be predicted in the current frame. This can be formally expressed as [9]:

$$\widehat{h_i} = h_i(\widehat{c_{x_i}}, \widehat{c_{y_i}}, \alpha_i, \beta_i, \theta_i) \tag{4.45}$$

where

$$(\widehat{c_{x_i}}(t), \widehat{c_{y_i}}(t)) = 2C_i(t-1) - C_i(t-2) \tag{4.46}$$

and

$$C_i(t) = (c_{x_i}(t), c_{y_i}(t)) \tag{4.47}$$

However, this function assumes that by keeping all other ellipse model parameters the same, the predicted object hypothesis will maintain the same direction and magnitude of translation on the image plane. When associating the skin pixels with the predicted object hypothesis, these parameters are updated. The updated parameters can be then used as a good indication of the angle and size of each hand in the current frame.

The components discussed in the tracking phase form the foundation of the hand-tracking framework. In Chapter 6 it will be determined how well these components perform in tracking the hands accurately.

4.3 Learning phase

The learning phase was implemented in the third DSR cycle (see Figure 4.11). The idea of selecting special points relating to an object and performing a local analysis on these points could be advantageous, since these points contain much more contextual information about an object than the rest of the image. These points are referred to as feature points and are synonymous with the terms ‘interest points’ or ‘keypoints’. Pixels or regions that have distinctive or discriminative properties are often selected as feature points. As long as a sufficient number of these points can be found in an image, it can be accurately localised and used to establish correspondences between images or image regions.

In order to match feature points, the features need to be uniquely detected and described. Feature points can therefore be divided into three groups: feature detectors, feature

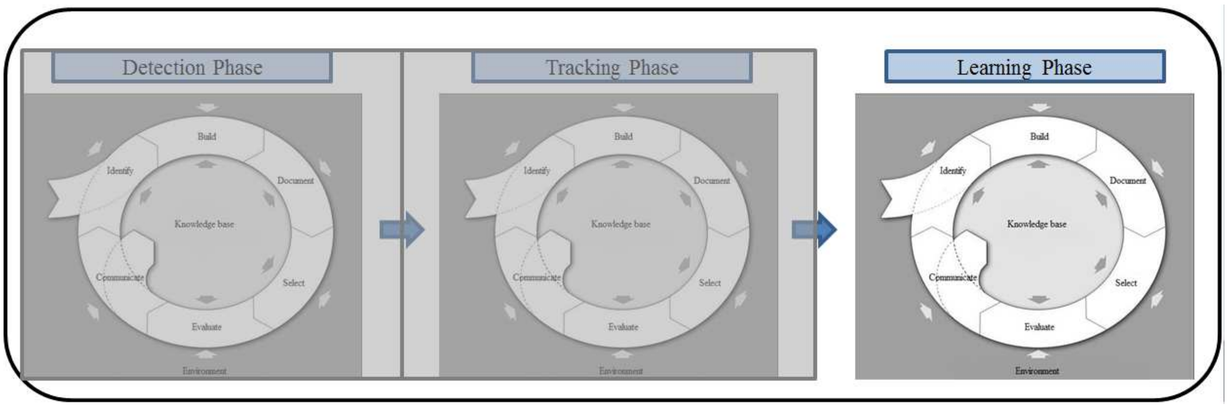


Figure 4.11: The third DSR cycle, consisting of the learning phase.

descriptors and feature matching. In the following sections each of these groups will be discussed and the best methods will be identified.

4.3.1 Feature detectors

Over the past two decades several feature detectors have been proposed with the aim of improving accuracy and efficiency [15]. Feature detectors are used to find a feature point in images that can later be extracted by a feature descriptor. In this section only the SIFT detector that is used will be discussed. Other state-of-the-art feature detectors can be referred to in Appendix A.

4.3.1.1 Scale-invariant feature transform

The SIFT detector was proposed by Lowe [99] as a method of locating scale, rotation and translation invariant features. These features are considered to be highly distinctive, thus allowing a feature to match correctly with a high probability against a large number of features. The algorithm takes a cascade filtering approach thereby applying the expensive operations only to the locations that pass an initial test. It operates by following a three-step strategy to determine the set of feature points:

1. *Scale-space extrema detection*: This step searches over all image scales and locations to identify potential feature points that are invariant to scale and orientation by determining the maxima or minima of a difference-of-Gaussian function, D. To detect the local maxima or minima, each candidate point is compared with its

eight neighbours at the same scale and with its nine neighbours one scale up and one scale down. If the value of D is the maximum or minimum of all the compared points, then this point is an extrema and considered to be a feature point.

2. *Feature-point localisation*: In this step, feature points are selected based on measures of their stability where unstable features are eliminated from the final list of feature points by locating those that are poorly localised on an edge or have a low contrast. This is achieved by determining the Laplacian value for each feature point.

3. *Orientation assignment*: For each feature-point location, one or more orientations are assigned based on the local image gradient directions. This ensures invariance to orientation, scale and location transformations of image data in all future operations.

4.3.2 Feature descriptors

After feature points have been located, the features are described by means of feature vectors that are referred to as feature descriptors. To couple the detectors, many descriptors have been proposed. In the following section, only the SIFT descriptor will be discussed. Other state-of-the-art feature descriptors can be referred to in Appendix A.

4.3.2.1 Scale-invariant feature transform descriptor

Using the selected scale of the SIFT detector, local image gradients are computed in the region around each feature point. The gradients' magnitude and orientation are then quantised into eight orientation histogram bins summarising the contents over 4x4 subregions. This results in a feature vector consisting of 128 elements. This is followed by normalising the vector, resulting in illumination invariance that allows for significant levels of local shape distortion [99].

Similar to the feature detectors, feature descriptors have their relative strengths and weaknesses. In Section 4.3.4 a comparison is made to determine a feature descriptor that is suitable.

4.3.3 Image matching

When matching distinctive characteristics between images, descriptors are very useful. For example, suppose two images are to be matched, feature points are first detected and then extracted into feature vectors using a descriptor. Given two feature vectors, a match can be found by measuring the similarity between them. The metric used to define the level of similarity is often the Euclidean distance or hamming distance, depending on the feature descriptor used. In the first image each feature vector is compared to all feature vectors in the second image. The feature vector pair that obtains the best similarity score, the one with the shortest distance, is selected as the best match.

Two main algorithms, namely, Brute-force and FLANN-based matcher, can be used for feature matching. In the following section, the FLANN-based matcher will be discussed. The Brute-force matcher is given in Appendix A.

4.3.3.1 FLANN-based matcher

The FLANN matcher [112] provides an efficient search method that is used to find correspondences between feature vector sets. The library was developed with the aim of minimising the predicted search cost while maintaining a high accuracy. This is achieved by using a cross-validation approach to automatically determine the best nearest neighbour algorithm and optimal parameter values for any given dataset.

Therefore, to optimise the performance, the FLANN-based method applies a training stage where index trees are constructed for *query* sets. Furthermore, experiments by Muja and Lowe [112] have shown that by using FLANN, a higher performance increase of 1 000 times can be achieved compared to the linear search, while retaining the ability to identify 95% of the correct nearest neighbours.

4.3.4 Comparison of feature detectors and descriptors

Several studies [15, 58, 108] have evaluated different feature detectors and descriptors with the aim of identifying the best one. The classic Harris detector has been used for its distinctive corner detection abilities; however, it is limited by its lack of scale invariance and computational complexities [16]. As an extension, the good-features-to-track detector locates corners more uniformly distributed across the image. Another extension is the FAST detector, which was designed to be more efficient; however, it does not calculate the size or orientation of the feature points [14]. The feature-point orientation was later addressed by the ORB detector. At the same time the BRISK detector aimed to improve the scale parameter of FAST by computing the score over several scales in a scale-space representation; however, it was shown to be sensitive to light-intensity changes [15].

Similarly, STAR is a multi-scale detector that preserves rotational invariance and improves efficiency with the use of integral images [137]. As a region detector, MSER does not extract feature points, but determines distinctive regions in images [138]; however, it generates low detections since it requires well-defined homogeneous regions.

The SIFT detector [99] was introduced to locate features that are invariant to scale, rotation and translation. The advantage of SIFT is that it has the ability to extract highly distinctive features, thereby enabling one to match features with a high probability [88]. To improve the computational efficiency of SIFT, the SURF detector was introduced [17].

To complement these detectors, a number of descriptors have been proposed. The SIFT descriptor extracts the feature points while describing the scale, rotational and translational properties of the features in a unique feature vector. The SURF descriptor extracts similar features to SIFT, but aims at being computationally faster; however, it is limited by its sensitivity to rotation [15]. BRIEF is a stand-alone descriptor that takes a different approach to feature vectors by describing features in the form of binary strings. Due to the nature of its design, it is not rotation or scale invariant [137]. It has, however, inspired more recent binary-based descriptors such as ORB and BRISK. The ORB descriptor extends BRIEF by adding rotational invariance, but is limited by a fixed scale. The BRISK descriptor, however, adds both rotation and scale invariance. FREAK is another stand-alone descriptor that improves the sampling pattern and pair selection of BRISK. The aim of the FREAK descriptor is to shorten the computation time, but this results in a lower accuracy [137].

These studies [15, 58, 108] commonly agree that none of the feature detectors or descriptors outperforms the others. They suggest that the best approach would be to select one that is most suited to the requirements of a particular application. Many of the recent detectors and descriptors are aimed at either improving the computation speed or overcoming the limitations of a previous detector or descriptor. These improvements often come at the expense of a lower accuracy [15, 88, 137].

In this research application, a detector that will result in a larger number of potential feature points that are scale, rotation and translation invariant is required. Furthermore, a descriptor that uniquely identifies feature points to improve accuracy is also needed. Several studies [15, 16, 88] agree that SIFT is considered to be one of the best-performing algorithms; it generates the most robust feature points and uniquely describes them for feature matching. Therefore, in the learning phase, the SIFT feature detector and descriptor will be used to learn uniquely identified features to help recover from tracking failure. In Chapter 6 it will be determined whether these features can exploit contextual information and result in more accurate hand-tracking.

4.4 Summary

In the preceding sections the components that form part of the detection, tracking and learning phases were explored. The detection phase includes algorithms such as LBPs, random forests and SVMs. Different LBP variants were described and a comparison was made between employing global versus local LBP histograms. It was concluded that the choice of a suitable LBP variant is dependent on many factors such as the type of application, computational efficiency and robustness to illumination. Furthermore, random forests were described and their strengths and weaknesses were compared. Similarly, the SVM and different kernels were explained.

The tracking phase includes a unique skin-segmentation algorithm and extends the data association method by Argyros and Lourakis [9]. The skin-segmentation algorithm operates by detecting the face and using the skin-colour distribution around the nose to determine an individual's skin colour. To cluster the skin pixels together, a connected-components labelling algorithm was described. Furthermore, the algorithm that extends the data association method to allow for multiple skin-coloured object tracking in both constrained and unconstrained environments was explained.

The learning phase includes algorithms that detect, describe and match distinct features in images. Several studies [15, 58, 108] confirmed that none of the feature detectors and descriptors outperforms the rest in all cases. They therefore suggest that the best approach would be to select one that is most suited to the requirements of a particular application. In the current application SIFT is well suited since it generates the most robust feature points and uniquely describes them for feature matching.

The following chapter addresses the way in which the three phases are integrated and linked to form three prototypes.

Chapter 5

Hand-Tracking Framework

In this chapter the systematic design and procedures that were implemented to investigate the research questions will be explained: the way in which the different algorithms were linked and integrated in a particular phase and collectively used to form the final hand-tracking framework. The high-level conceptual view of this framework can be described by the three phases identified in the previous sections.

Following the DSR methodology, three hand-tracking frameworks, represented by three prototypes,[1] were developed; namely, prototype T, prototype DT and prototype DTL. Fundamental to the framework is the tracking phase. It forms the basis of this research and is therefore engineered in the first design cycle by prototype T. This was followed by prototype DT in the second design cycle in which both the detection and tracking phase were integrated into the framework. The last design cycle culminated in prototype DTL — an integration of the detection, tracking and learning phases. In the subsequent sections each prototype is further explored.

5.1 Prototype T

Prototype T was developed with the aim of evaluating the tracking phase. This phase consists of an algorithm that continuously tracks both hands independently in constrained and unconstrained environments.

[1]The prototypes were named according to the phase that was included in the framework, where the framework in prototype T consists of the tracking phase only and the framework in prototype DTL consists of all three phases.

In sign language, the face and hands form an integral part in the communication amongst those who are hard-of-hearing or Deaf. While communicating in sign language it is important that both the face and hands are visible throughout the conversation. As part of a translation system, it will be necessary to capture the upper body of an individual, face forward. The tracking phase therefore builds on this fact and begins by locating an individual's face using the face-detection algorithm. An overview of the systematic design of the prototype is shown in Figure 5.1.

Face detection is implemented for two reasons: firstly, to determine if an individual is present and, secondly, to extract an individual's skin colour. Using the method described in Section 4.2.1.3, the skin-colour distribution is extracted from the region around the nose, since it is less likely to be negatively affected by any non-skin areas. This colour distribution is back projected to provide a probability map on which a threshold is applied to produce a skin-segmented image. In unconstrained environments this may lead to several other areas in the background that may fall within the identified colour distribution and may incorrectly be identified as an individual's body part, as illustrated in Figure 5.2. To avoid these areas and highlight the regions of interest such as the hands, a background subtraction method is applied, resulting in a segmentation as shown in Figure 5.4.

In constrained environments a simple background subtraction technique will meet the requirements of segmenting an individual from the background, as long as the background remains static. Everything else will be considered as foreground, as shown in Figure 5.3.

This technique, however, would not be suitable for unconstrained environments. In these environments the individual alone should be considered as foreground and everything else as background. This is a challenging task and is not achievable using any of the existing background subtraction techniques. Many of the recent background subtraction techniques[2] were developed with the aim of training a background model and continuously updating the model to determine the foreground. These techniques form different variants or improvements of the adaptive non-parametric Gaussian mixture model used by Stauffer and Grimson [145]. One of these techniques that attracts attention and that was used in the development of prototype T was that of Zivkovic and van der Heijden

[2]For a list of recently implemented background subtraction algorithms, see *BGSLibrary: An OpenCV C++ Background Subtraction Library* [143]

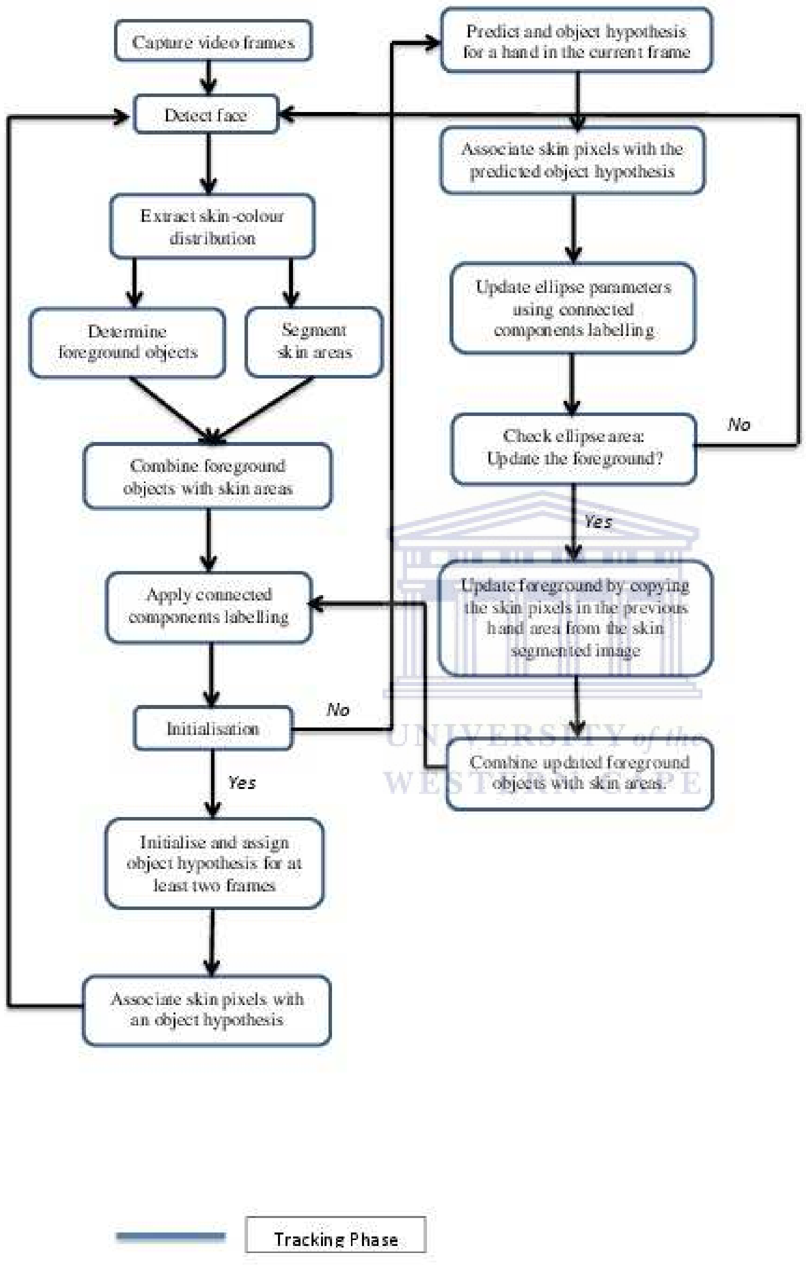
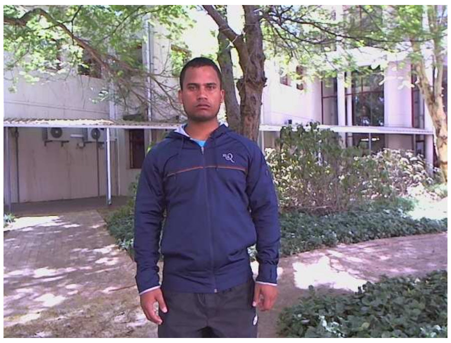

Capture video frames
Detect face
Extract skin-colour distribution
Determine foreground objects
Segment skin areas
Combine foreground objects with skin areas
Apply connected components labelling
Initialisation
No
Yes
Initialise and assign object hypothesis for at least two frames
Associate skin pixels with an object hypothesis
Predict and object hypothesis for a hand in the current frame
Associate skin pixels with the predicted object hypothesis
Update ellipse parameters using connected components labelling
Check ellipse area: Update the foreground?
No
Yes
Update foreground by copying the skin pixels in the previous hand area from the skin segmented image
Combine updated foreground objects with skin areas.
Tracking Phase

FIGURE 5.1: An overview of the systematic design of prototype T.


(a) The original image.

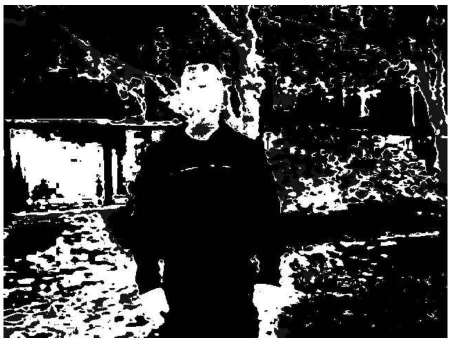
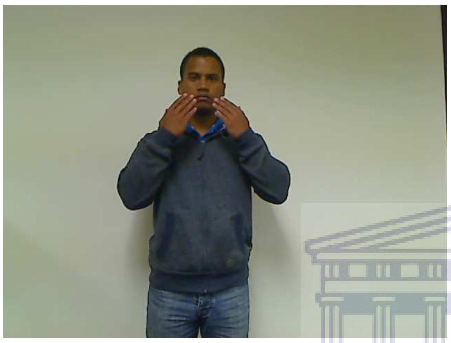

(b) The skin-segmented image.

FIGURE 5.2: An example of the skin segmentation output on an unconstrained environment.


(a) The original image.

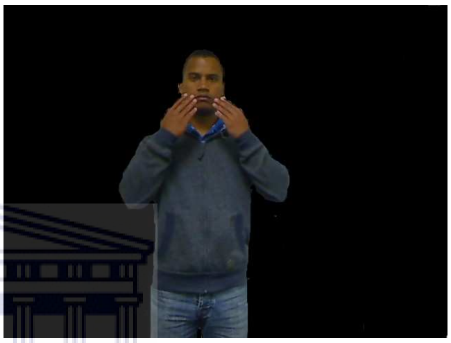
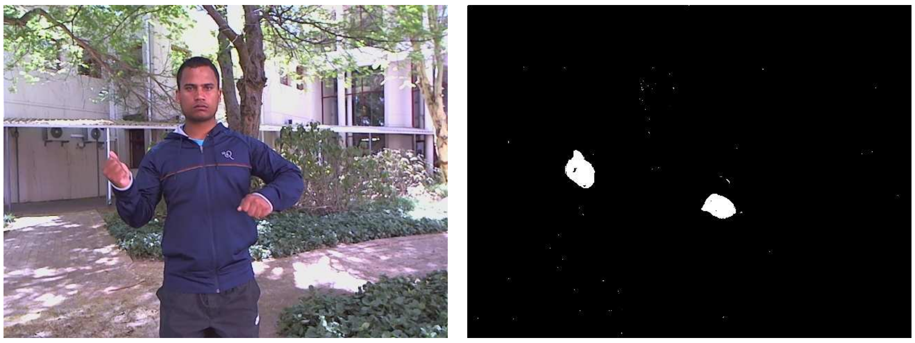

(b) The foreground image.

FIGURE 5.3: An example of the foreground segmented from a constrained environment with a static background.

[174]. Their background subtraction algorithm differs from that of Stauffer and Grimson [145] by enabling the selection of the number of Gaussian components and adapting these components per pixel.

By employing this background subtraction algorithm and updating the model at every two frames, the hands can be successfully segmented as the foreground while labelling everything else as the background. In unconstrained environments any new objects entering the scene may also be considered as the foreground. To eliminate these objects as appearing as foreground objects, the foreground image is combined with the skin-segmented image to result in a combined image that only contains foreground objects that are skin coloured, as shown in Figure 5.4. This combined image will hereafter be referred to as the foreground-skin image.

The pixels in the foreground-skin image therefore represent the pixels that are skin coloured and that are labelled to be in the foreground. These pixels are formed into clusters using the connected-components labelling algorithm. Pixels considered to be



(a) The original image. (b) The foreground-skin image.

FIGURE 5.4: An example of the skin segmentation output combined with the background subtraction output on an unconstrained environment.

noise are removed using a threshold applied by the connected-components labelling. Moreover, clusters that are smaller than a quarter of the face (the size of a fist) are also removed. These clusters are assumed to not belong to a hand, since the smallest hand form is a fist.

Under the assumption that an individual would be facing the camera and given the list of clusters, it is assumed that a cluster in the bottom left and bottom right of the image would represent the right and left hands, respectively, as shown in Figure 5.5.

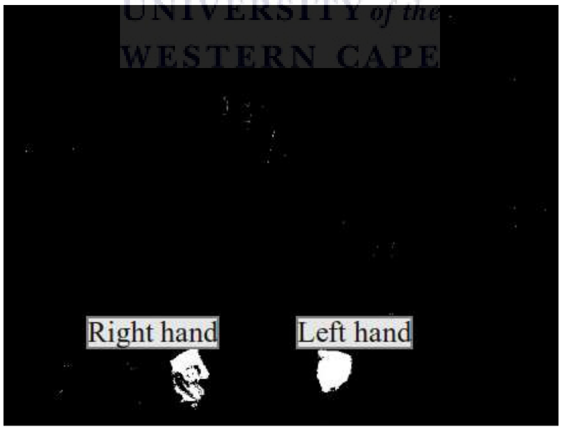


FIGURE 5.5: Association of the right and left hand in the initial frames.

If an object hypothesis has not been associated with the clusters, then an object hypothesis is initialised and an ellipse model is assigned to the respective clusters, based on the side of the body where the cluster lies. Here, only two object hypotheses are required, a left and right hypothesis for each hand. According to the rules in Section 4.2.3.2, all skin pixels are associated with either object hypothesis. If an object hypothesis has already been initialised and an ellipse model has been assigned to a respective hand in at least two previous frames, at time $t-1$ and $t-2$, then the location of that hand can be predicted with an ellipse model in the current frame, at time t, using equation 4.45. Skin-coloured pixels in the current frame are then associated with the predicted-object hypothesis using the distance metric in equation 4.40. By applying the connected-components labelling to the associated pixels in the current frame, the parameters for the current ellipse model can be updated.

When associating skin-coloured pixels in the current frame with the predicted-object hypothesis, if the number of pixels located inside the ellipse model is less than half of the ellipse area, where $area = \pi \cdot M_1 \cdot M_2$, then it is assumed that the hand is moving towards a stationary position and a flag is set to update the foreground image. This is motivated by the fact that when the hands do not move, the updated-background model labels the hand pixels as part of the background and the foreground image yields either very little information or no information at all, as shown in Figure 5.6. To address this limitation, the hand area in the foreground image is updated by selecting pixels from the same hand area at $t-1$, on the current skin-segmented image, as illustrated in Figure 5.6.

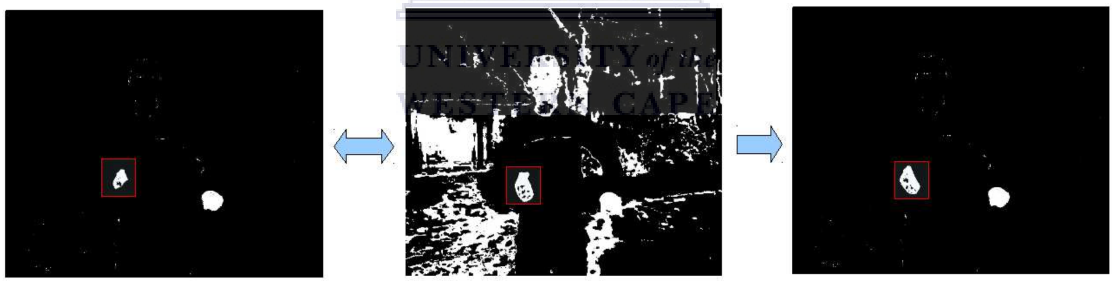


FIGURE 5.6: An example of the updating process of the foreground-skin image.

An approximation of the hand area is extracted by determining the convex hull from the set of points on the border of a given cluster. The convex hull can be visualised as the shape formed by stretching an elastic band around the cluster, as depicted in Figure 5.7.

This operation is based on the assumption that if the hand is stationary, then the position at which the hand is at time $t-1$ should contain skin pixels in the same position at time t. After the foreground image has been updated, it is combined with the skin-segmented image to form the updated foreground-skin image. This is followed by recomputing the list of clusters and associating skin pixels with the predicted-object hypothesis.

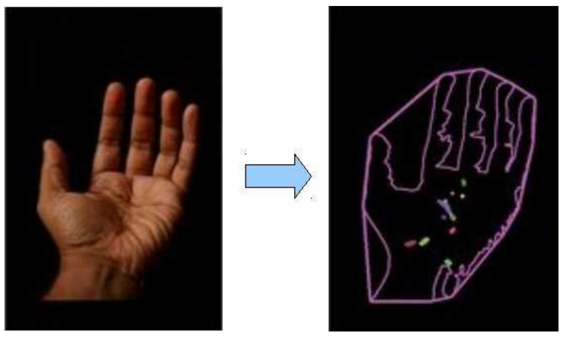


FIGURE 5.7: An example of the convex hull of a hand.

This process allows the hands to be tracked in both constrained and unconstrained environments.

5.2 Prototype DT

The aim of prototype DT is to add a hand-detection algorithm to the hand-tracking framework. This algorithm avoids the assumption made in prototype T, where clusters being tracked are assumed to be a hand. Instead, each cluster in prototype DT is queried to be a hand or not before it is tracked.

In the initial frames, instead of assuming that clusters on the side of the body are hands, the hand-detection algorithm is applied. If the cluster is determined to be a hand, an ellipse model is assigned to it. After associating skin pixels to a predicted-object hypothesis in every frame, new clusters are formed using the connected-components labelling. This is followed by determining if the cluster is a hand. These additional steps are integrated into the framework; their integration is shown in Figure 5.8.

To detect whether a cluster is a hand, features corresponding to the cluster area should be extracted and classified using a trained model. The success of LBPs used in encoding various object appearances such as people, food types, cars and faces is further extended to hands. This is motivated by the discriminative power of LBPs and their ability to uniquely represent image features. LBP features in the form of local histograms, global histograms and LBP images have previously been used in other applications [139, 169]. Their applicability to hand detection in these forms is therefore investigated in Section 6.6.2.1.

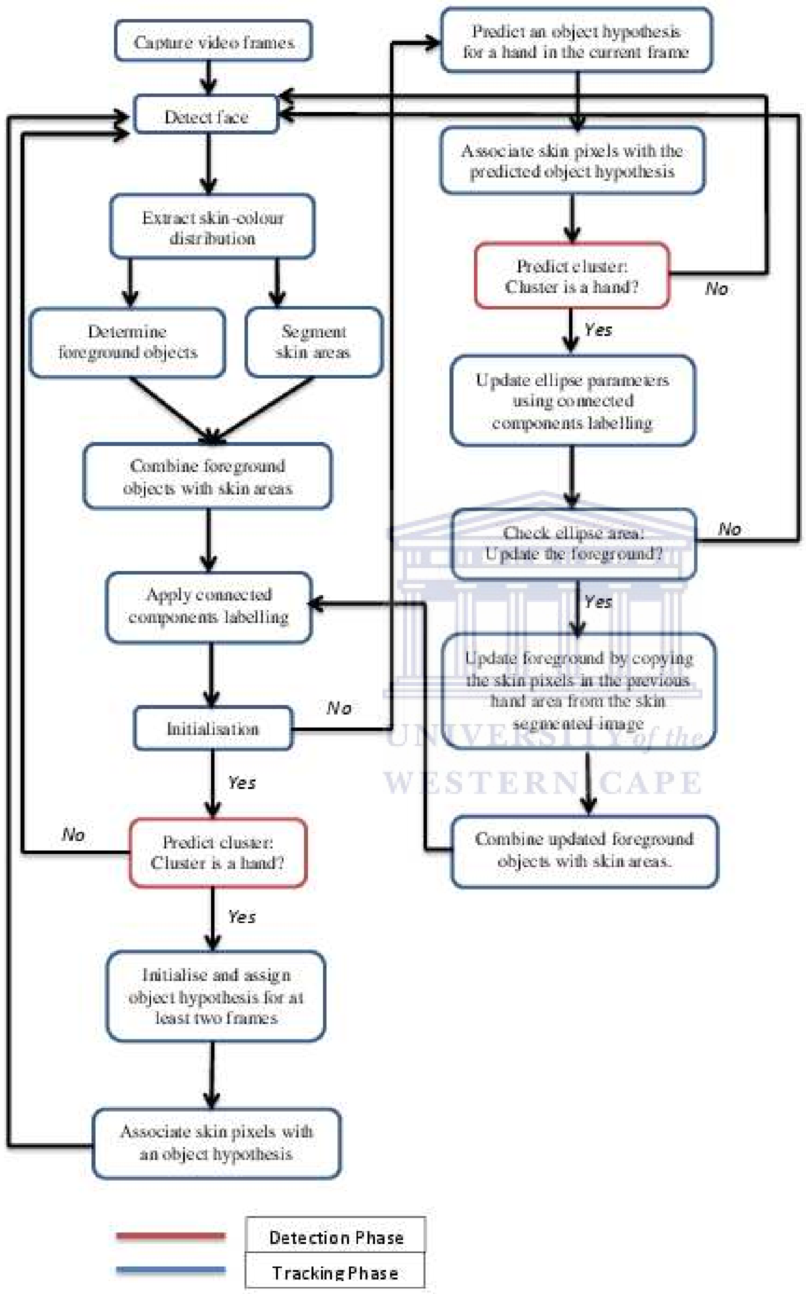
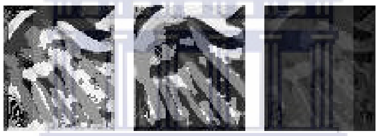

Capture video frames
Detect face
Extract skin-colour distribution
Determine foreground objects
Segment skin areas
Combine foreground objects with skin areas
Apply connected components labelling
Initialisation
No
Yes
Predict cluster: Cluster is a hand?
No
Yes
Initialise and assign object hypothesis for at least two frames
Associate skin pixels with an object hypothesis
Predict an object hypothesis for a hand in the current frame
Associate skin pixels with the predicted object hypothesis
Predict cluster: Cluster is a hand?
No
Yes
Update ellipse parameters using connected components labelling
Check ellipse area: Update the foreground?
No
Yes
Update foreground by copying the skin pixels in the previous hand area from the skin segmented image
Combine updated foreground objects with skin areas.
Detection Phase
Tracking Phase

FIGURE 5.8: An overview of the systematic design of prototype DT.

Since LBPs are based on pixel intensity values, the default RGB image is converted to greyscale. This is followed by applying a Gaussian blur to reduce image noise and ensure that spurious high-frequency information do not appear when resizing the image. In this research three LBP variants were used, namely, OLBP, ELBP and ULBP. In the OLBP a neighbourhood of eight and a radius of one were used. In the ELBP a neighbourhood of eight and radius of two were used. This (8,2) neighbourhood was shown to produce distinct features [6]. In the ULBP a neighbourhood of eight and radius of two were also used, with the additional step of using uniform LBP values. The uniform LBP values were obtained by employing a look-up table. Given a LBP decimal value, the corresponding uniform values were searched for in the look-up table from one of the 59 uniform values, as described in Section 4.1.1.1. For each image, the LBPs were computed using the respective methods as described in Section 4.1.1, resulting in LBP images as illustrated in Figure 5.9. To represent the LBPs visually, a normalisation method with a range of 0 to 255 was used.



(a) The original LBP image. (b) The extended LBP image. (c) The uniform LBP image.

FIGURE 5.9: A visual representation of the outputs of the different LBP variants, described in Section 4.1.1.

The images contained in the hand dataset (discussed in Section 6.2.3) consists of various image sizes. In order to maintain consistency among the images and reduce the computational load, each image was resized to an image size of 40x30 pixels. Given the set of positive and negative hand samples, a feature set equal to the sum of positive and negative hand samples was used.

From the LBP images, both spatially enhanced and global LBP histograms were computed. In spatially enhanced LBP histograms the shape information and location of LBP occurrences within an image is retained. This was achieved by dividing each LBP image into four equal regions. For each region a histogram with a size of 256 and a range of 0 to 255 was used. This represents the local information for each region. The histograms for each region in an image were then concatenated to form a single spatially enhanced feature histogram, as shown in Figure 4.5.

In global LBP histograms the LBP occurrences are encoded without any indication of their location. In this method, each image was not divided, but instead a histogram was computed over the entire image, as shown in Figure 5.10.

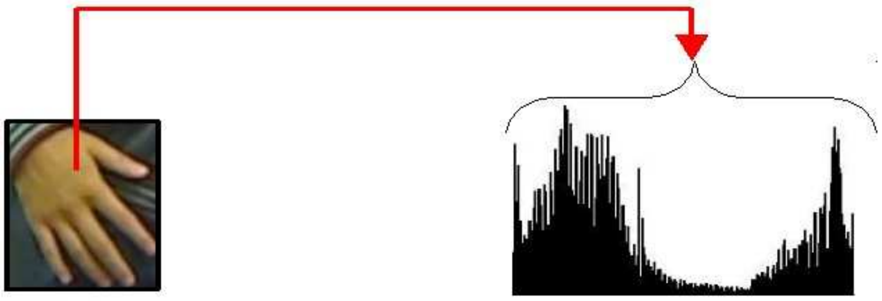


FIGURE 5.10: An example of the global representation of a LBP image in the form of a single histogram.

After completing the spatially enhanced and global histograms, the feature vectors for each image were created. These feature vectors consisted of a size of 1024 and 256 values for the spatially enhanced and global histograms, respectively. In addition, each feature vector was assigned a label of one or zero if the image was a positive or negative hand sample, respectively.

When the LBP images were used without histograms, the LBP values were directly used to compile the feature vectors, where the feature vector size was 1200, as depicted in Figure 5.11. After computing the feature vector for each image, a label of one or zero was assigned if the image was a positive or negative hand sample, respectively.

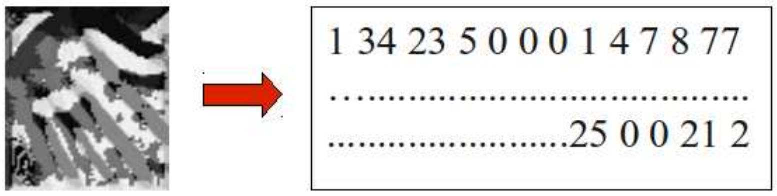


FIGURE 5.11: An example of a feature vector directly formed from the labelled pixel values of a LBP image.

The feature vectors described relate to the training set. In the testing set, however, each feature vector was assigned a label of zero by default where the label was subjected to change, based on the outcome of a trained model.

In order to generate a trained model, the feature set was trained using random forests and SVMs; the two classifiers are compared in Section 6.6.2.1.3.

Random forests: For random forests, the feature set was trained using the method described in Section 4.1.2. According to Breiman [21], only two parameters need to be evaluated when constructing the random forest, namely, the number of random split variables and the number of trees [12]. To determine the optimal parameters for training the random forest, different parameters are evaluated in Section 6.6.2.1.1.

Support vector machines: For SVMs, before the feature set can be trained or tested, the feature values need to be scaled. The advantage of scaling the features is to prevent values with a greater numeric range dominating those in a smaller numeric range [30]. Thus, for both the training and testing feature set, the values were scaled to the range $[-1, +1]$. To effectively train an SVM model and accurately predict the test data, an appropriate kernel and kernel parameters, C and γ, are required for the given problem. These parameters can be found using an exhaustive approach by trying each C and γ combination where each parameter is an exponentially growing sequence [75]. An alternative approach is to use a cross-validation strategy that divides the training into n equal-sized subsets, where the SVM is trained on the $n-1$ subsets and tested on the remaining subset for each parameter combination [75]. Therefore, the combination with the best cross-validation accuracy was chosen as the optimal parameters for the given problem. These parameters are determined in Section 6.6.2.1.2.

5.3 Prototype DTL

When tracking hands in unconstrained environments, objects in the background may negatively affect the tracking process, which may lead to tracking failure. For example, objects that are near to the hand and possess the same skin-colour distribution as the individual may be incorrectly tracked. The aim of this prototype is to recover from such failures by adding a learning phase algorithm to the hand-tracking framework. The prototype is based on the concept of context-based tracking where objects surrounding a tracked object may possess as much information about the tracked object as the object itself. This information is retrieved from the surrounding object features, where spatial and temporal context information is exploited.

These features, however, are only extracted if the cluster being tracked is indeed a hand. The algorithm is therefore integrated after the hand-detection algorithm, as illustrated in Figure 5.12.

Thus, for every image, if a hand has been identified, distinctive features are extracted using SIFT. These features correspond to the spatial context information. Features extracted using SIFT are considered to be highly distinctive, thus allowing a feature to be matched correctly with a high probability against a large number of previously stored features.

In the initial frames the algorithm operates by first detecting a hand. This is to ensure that incorrect features will not be stored. If only a single hand is detected, then feature points on and around the hand are generated and their descriptors are extracted. These feature points and descriptors are extracted using the methods described in Section 4.3 within a region around a skin-coloured cluster. A boundary region of the cluster is determined by finding the extreme points on that cluster. To allow for more features to be extracted that may be significant in linking their existence to a hand, the region around the hand is enlarged by one-fourth of the bounding region, as shown in Figure 5.13. In addition, features from objects such as wristwatches, bracelets, rings and various other objects assist in distinguishing one hand from the other.

If the detected hand is a right or left hand, then the generated feature points and descriptors are stored in two separate databases: one for right hand and one for the left hand, as illustrated in Figure 5.14.

Separate databases are used since the features in the right- and left-hand databases are used not only to recover from tracking failure, but also to help distinguish between the right and left hand by matching the features separately.

In the subsequent frames following the initialisation step, pixels are associated based on a predicted-object hypothesis position for either or both the right and left hand in the current frame. This allows the system to identify pixels that are closer to the predicted-object hypothesis position. These pixels are formed into clusters using connected-components labelling. This results in a list of potential clusters that can be associated with a hand. When both hands are near to each other, then clusters near to both hands can be associated with each other.

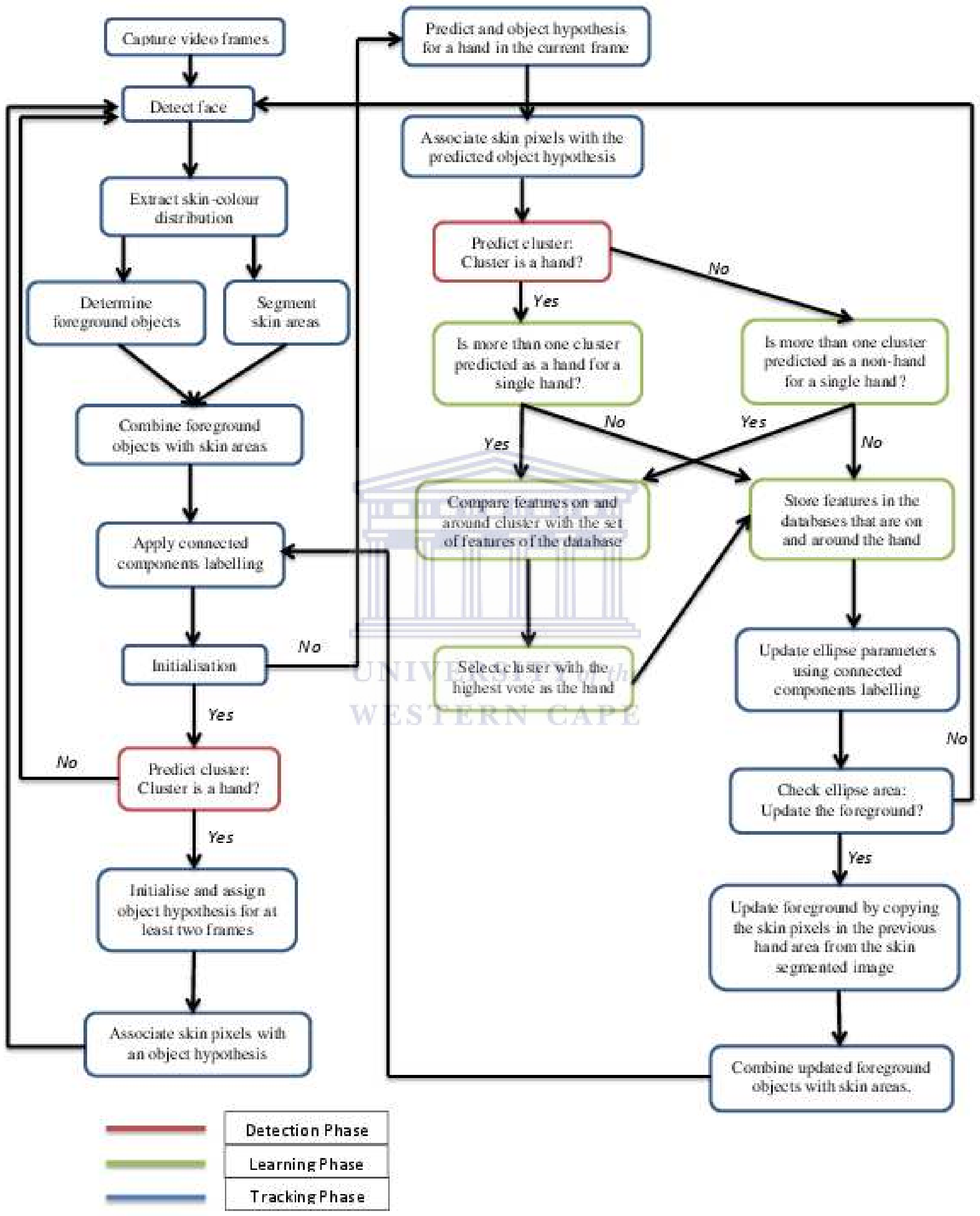

Capture video frames
Detect face
Extract skin-colour distribution
Determine foreground objects
Segment skin areas
Combine foreground objects with skin areas
Apply connected components labelling
Initialisation
No
Yes
Predict cluster: Cluster is a hand?
No
Yes
Initialise and assign object hypothesis for at least two frames
Associate skin pixels with an object hypothesis
Predict and object hypothesis for a hand in the current frame
Associate skin pixels with the predicted object hypothesis
Predict cluster: Cluster is a hand?
No
Yes
Is more than one cluster predicted as a hand for a single hand?
Is more than one cluster predicted as a non-hand for a single hand?
Yes
No
Yes
No
Compare features on and around cluster with the set of features of the database
Store features in the databases that are on and around the hand
Select cluster with the highest vote as the hand
Update ellipse parameters using connected components labelling
No
Check ellipse area: Update the foreground?
Yes
Update foreground by copying the skin pixels in the previous hand area from the skin segmented image
Combine updated foreground objects with skin areas.
Detection Phase
Learning Phase
Tracking Phase

FIGURE 5.12: An overview of the systematic design of prototype DTL.

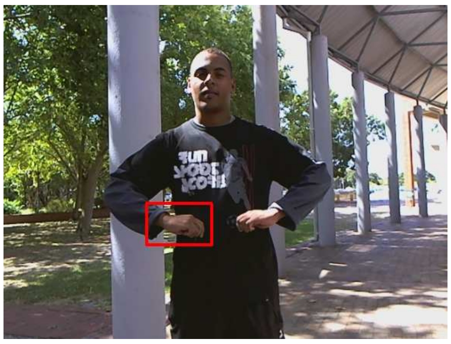


FIGURE 5.13: The bounding region of the hand used to extract contextual feature information.

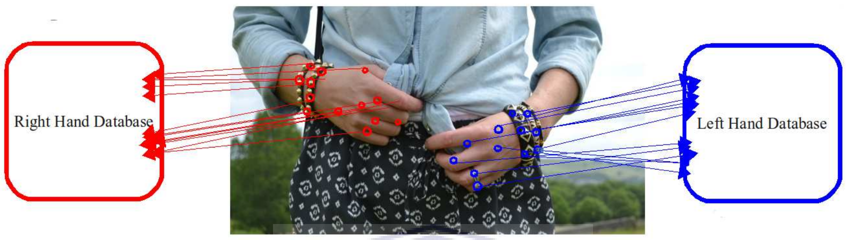


FIGURE 5.14: The features for each hand are stored in two separate databases.

When more than one cluster has been identified for a given hand, then features should be extracted for each cluster and stored in a query list. For example, if two clusters are associated with the right hand and only one cluster with the left hand, then features should only be stored in the query list for those features extracted from the two clusters associated with the right hand.

The features in the query list are then compared with the respective database, in this example, the database for the right hand. These features are compared by finding a match between descriptors using the FLANN-based matcher, described in Section 4.3.3.1. For each matched feature the distance between the feature in the query list and the database for the right or left hand is obtained. From the set of matched features, only the best matches are selected using the minimum distance from the set of distances for each feature point, where the best matches are those features that have a distance less than twice the minimum distance.

Similar to Grabner et al. [64], a voting strategy was adopted. For each feature that has been selected as a best matched feature, the cluster to which the feature belongs will be given a vote based on two rules:

1. If the feature shares the same location with a foreground pixel, then the cluster to which the feature belongs will be given two votes.

2. If the feature shares the same location with a background pixel, then the cluster to which the feature belongs will be given a single vote.

This voting strategy exploits the temporal context information and is based on the fact that objects or features that move with a hand, support the existence of that hand more than objects or features that are stationary. The total number of votes for each cluster in the query list is then obtained by adding the number of votes given to each cluster for each of the best-matched features. Therefore, given the total number of votes for each cluster, the cluster with the most votes is assigned as the respective hand. Referring back to the example of the two clusters, A and B, associated with the right hand, if cluster A and cluster B obtained a total vote of 23 and 47, respectively, then cluster B will be assigned as the right hand. The features belonging to cluster B will then be added to the database for the right hand. Furthermore, referring to the same example, if only one cluster, C, is associated with the left hand, then no comparison is performed and cluster C will be assigned as the left hand. This process is shown in Figure 5.15.

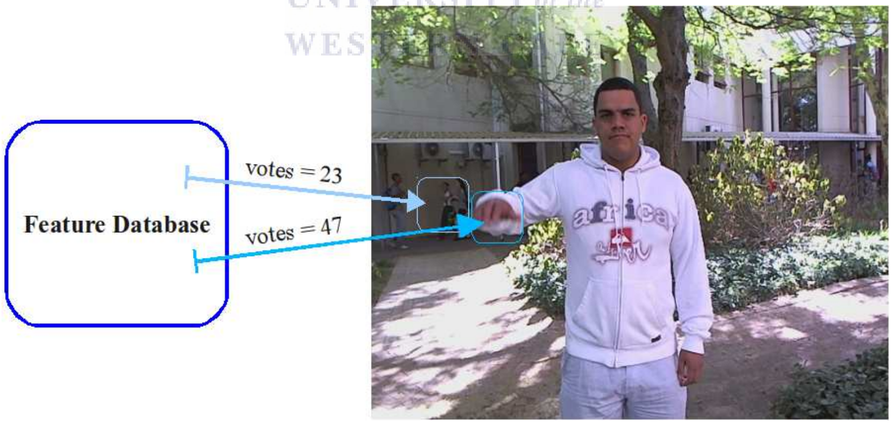


FIGURE 5.15: The voting strategy used in the learning phase, where the cluster with the highest number of votes is selected.

In the above example, both cluster A and B were assumed to have been predicted by the hand-detection algorithm as non-hands. The same procedure applies if both cluster A and B were predicted as hands. However, if cluster B was predicted as a hand and cluster A as a non-hand, then the voting strategy will not be applied and cluster B will be assigned as the right hand, with features belonging to cluster B added to the database. The same applies if cluster A has been predicted as a hand and cluster B as a non-hand.

This algorithm ensures that if an object is incorrectly tracked as a hand, then it would automatically recover the tracking of the ‘lost’ hand in the subsequent frames. It furthermore ensures that the right and left hands will be distinguishable from each other.

5.4 Summary

The chapter addressed the systematic design and procedures in which the hand-tracking frameworks were implemented and used. It proposed several new algorithms: (1) articulated hand detection using local binary patterns with either random forests or SVMs; (2) pixel-level data association for independent hand-tracking in unconstrained environments; and (3) exploring contextual features towards assisting hand-tracking. Moreover, it described the way in which the different algorithms were linked and integrated for each phase and collectively used to develop three prototypes: prototypes T, DT and DTL. Each prototype followed a new design cycle and illustrated the way in which an additional phase was added to the framework. In the following chapter the datasets used are described, the prototypes are evaluated and the results are discussed.

Chapter 6

Experimental Results and Analysis

6.1 Introduction

In this chapter the performance of the three prototypes (T, DT and DTL) will be evaluated. This involves comparing the three prototypes using a dataset of SASL videos. To perform these evaluations an evaluation protocol was designed entailing a pre-defined procedure for the implementation and evaluation of the proposed experiments. This evaluation protocol defines four main elements: the experimental setup, dataset, ground-truth criteria and metrics of accuracy.

The accepted practice for evaluating tracking algorithms is either subjective or objective [104]. The preferred analysis is objective, since it allows the prototypes to be compared and allows the experiments to be repeated. Furthermore, it allows one to find the best parameters for the hand-detection algorithm used in prototypes DT and DTL.

Before discussing the evaluation of the three prototypes, it is important to explain sign language data, not only to understand how it is linguistically used, but also how it influenced the creation of the dataset. Furthermore, the criteria used to provide the ground-truth data will be explained.

All experiments evaluated are aimed at answering the research questions and determining the suitability of the hand-tracking system as a component of a larger SASL translation system (see Figure 6.1). The SASL translation system consists of several components such as facial expressions, hand trajectory, hand location and hand shapes, which are important to recognise SASL. Hand-tracking is formed by two of these components, namely hand trajectory and hand location. Fundamental to the recognition of sign language is the effectiveness and accuracy of hand-tracking. This chapter therefore analyses the effectiveness and accuracy of hand-tracking toward SASL translation.

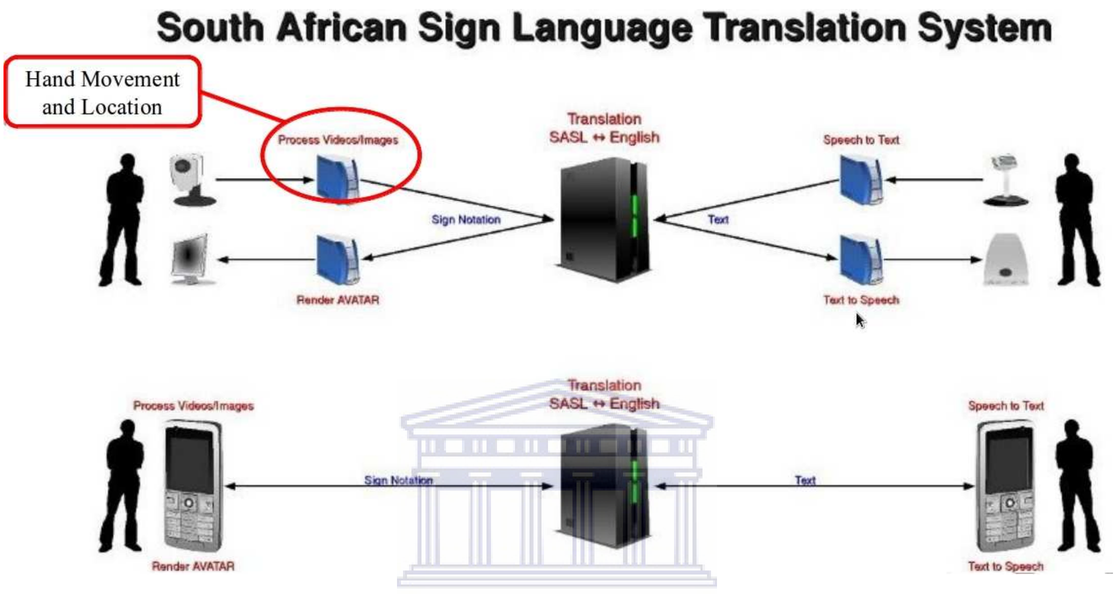


FIGURE 6.1: The South African sign language translation system.

In the following sections the elements of the evaluation protocol, which consist of the datasets, experimental setup, ground-truth criteria and metrics of accuracy, will be discussed in more detail, and the results and analysis of the experiments will be provided.

6.2 Dataset generation

In order to evaluate the prototypes, an appropriate dataset had to be selected with enough variation to cover the application of interest [104]. Publicly available datasets, especially large ones, are limited, due to factors such as the cost of computational resources of recording and the cost of their storage. Privacy and ethical concerns have further limited the recording and distribution of application-specific scenarios.

6.2.1 Sign language data

Sign languages differ in each country in the same way that spoken languages differ, and they have no resemblance to the spoken languages used in the country. Therefore SASL is unique to South Africa and a relationship cannot be drawn between SASL and the spoken languages used in South Africa. A large variation of signed gestures are used in SASL. Each gesture has a distinct meaning and is an important component of the conversation. The proposed hand-tracking framework should therefore deal with this variation and track the hands accurately to provide effective translation of the finer nuances of SASL. A standard set of signed gestures that cover all variations does not exist in the public sphere and thus had to be created.

In an effort to select signed gestures that cover the full variation of signs, three groups were identified from the *Fulton School for the Deaf SASL Dictionary* [73]. These groups consisted of signs that use a single hand; signs that use both hands without occlusion (i.e. the hands are always apart); and signs that use both hands with occlusion (i.e. they do sometimes obstruct each other). The groupings made provision for the fact that some signed gestures perform the same hand movement, but differ in terms of the hand shape.

6.2.2 SASL dataset

All signs in the vocabulary were grouped according to their hand movements and then allocated to one of three classes, as listed in Appendix C. From each class, ten signed gesture movements were selected, resulting in a total of 30 signs. These sign movements were carefully selected in order to evaluate the effectiveness of each prototype. All signed gestures were performed with an *open* hand, since hand shapes are beyond the scope of this research. To compile the dataset, 20 individuals were asked to perform these 30 signs. Before taking part in the research, each individual was given a consent form to complete, in order to adhere to the ethical standards set for constructing the dataset (see Appendix B). The individuals consisted of six females and fourteen males of different body types. This offers the opportunity to determine how well the prototypes performed in terms of these body types. In addition, the selected individuals represented a wide range of skin-colour tones in order to determine whether certain skin-colour tones affect the hand-tracking capabilities of each prototype.

For each sign, each individual began and ended in the *neutral* state, with the hands at the side of the body. This ensures consistency in the video dataset and the subsequent analysis. The signs were furthermore captured at different times of the day due to the lengthy duration of the video-capturing process.

The signs were captured in constrained and unconstrained environments in order to evaluate the effectiveness of each prototype in different environments. In total, the dataset consists of 1 200 sign videos where each video has an average number of 90 frames and a total of 108 015 frames.

The constrained environments include scenes that do not contain any background or foreground interference and where the background remains static. In this dataset a static white background was used where the individual was video recorded indoors. The unconstrained environments include scenes that allow for background and foreground interference, such as people walking, moving tree leaves, cars passing by and any object other than the individual performing sign language, and were recorded at different active outdoor locations. The differences between these environments are illustrated in Figure 6.2.

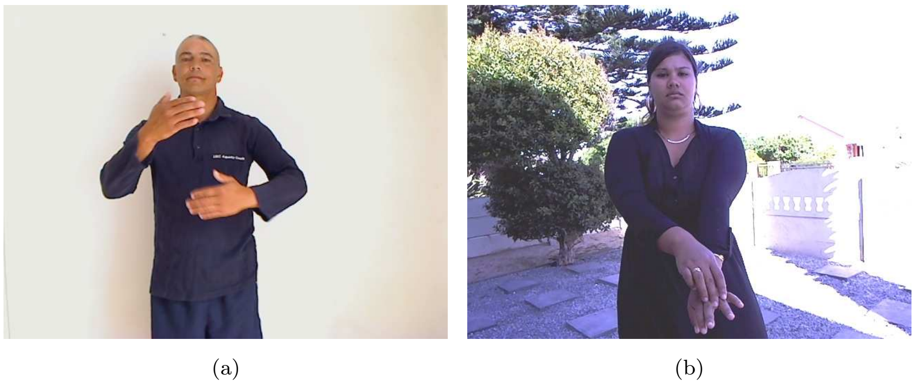


Figure 6.2: Comparison between the (a) constrained and (b) unconstrained environments[1].

6.2.3 Hand dataset

In addition to the tracking algorithm used in prototype T, in both prototypes DT and DTL a hand-*detection* algorithm was integrated into the framework to classify whether the object that was being tracked was a hand. To evaluate the effectiveness of this *detection* algorithm, a further dataset (consisting of hand and non-hand images) was constructed.

Datasets are often designed to be application specific. Therefore, to find an appropriate dataset in the public domain is no easy task. There are a few available datasets on hands; however, one that is appropriate for this application context did not exist. For example, a dataset that is publicly available is the Visual Geometry Group Hand Dataset [109]. The images in this dataset are, however, not suitable as the dataset contains images in unconstrained environments in which people and their hands are visible, as shown in Figure 6.3.

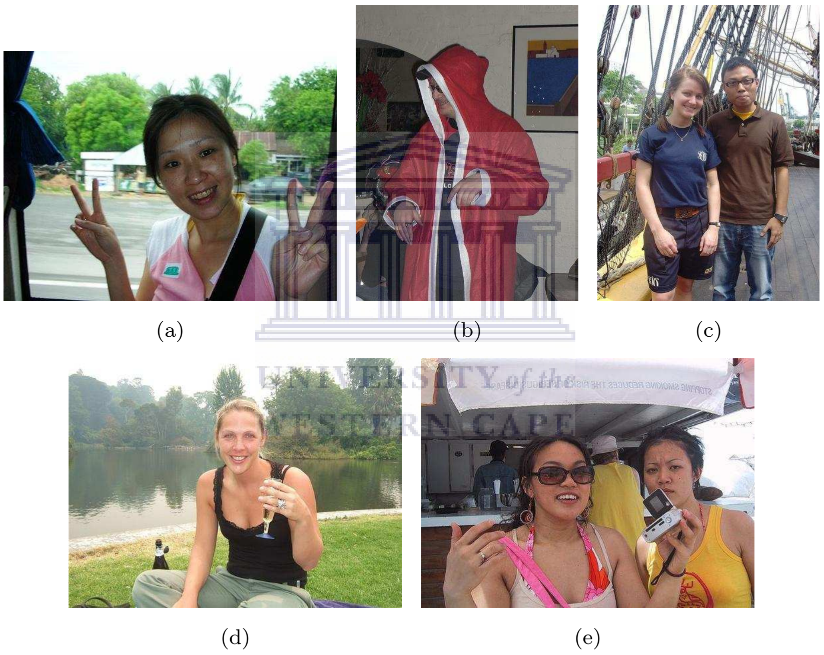


FIGURE 6.3: Sample images from the Visual Geometry Group Hand Dataset [109], which contain images in unconstrained environments in which people and their hands are visible.

To evaluate the hand-detection algorithm (in the DT and DTL prototypes), image regions that contain a hand, as depicted in Figure 6.4, were required. Two datasets that contain hand images in this form are the Cambridge Hand Gesture Dataset [85] and the NUS Hand Posture Dataset [87]. These datasets, however, have a plain background or have specific hand poses related to their application context, as shown in Figure 6.5, and thus were not suitable.

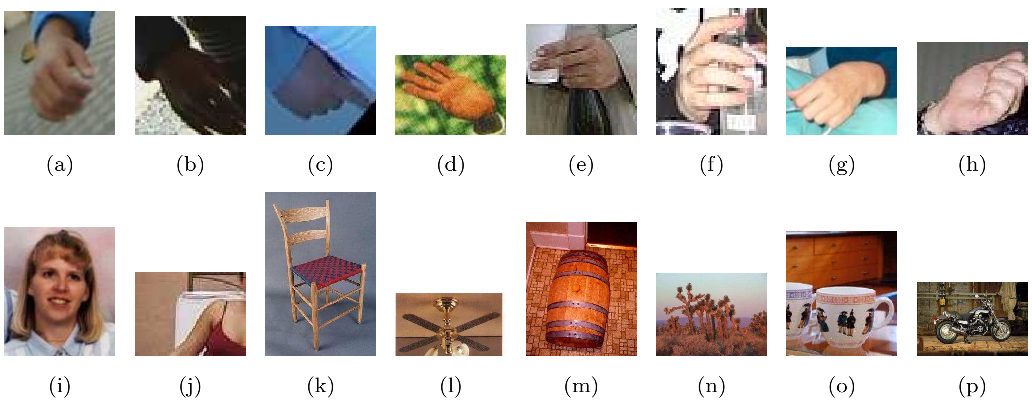


FIGURE 6.4: Sample hand images (a–h) and non-hand images (i–p) of the hand dataset used to evaluate the hand-detection algorithm.

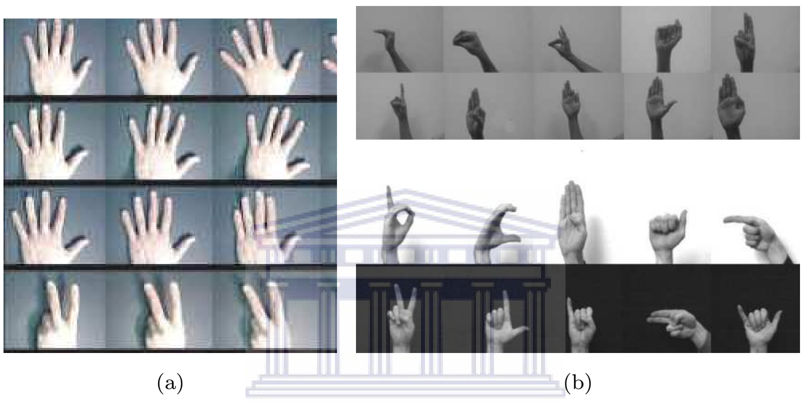


FIGURE 6.5: Datasets that contain hand images with a plain background and have specific hand poses: (a) Cambridge Hand Gesture Dataset [85]; (b) NUS Hand Posture Dataset [87].

Since none of the datasets was appropriate, a hand dataset was compiled. The hand dataset was compiled using the following datasets; hand images were selected from the Visual Geometry Hand Dataset [109] (the images were cropped around the hand) and were also selected from sign language videos not used in the SASL dataset. The non-hand images were selected from images of non-hand objects in the Caltech 101 dataset [50] and from the Visual Geometry Group Hand Dataset [109] in which non-hand regions were extracted. The dataset is summarised in Table 6.1.

This dataset consists of a training and testing set, where the training set is divided into a positive and negative set. The positive and negative training sets consist of 8 000 hand images and 8 000 non-hand images, respectively. In total the training set consists of 16 000 images and the testing set consists of 4 706 images.

Table 6.1: The hand dataset setup

Datasets	**Hand Images**	**Non-hand Images**
SASL videos	✓	
Visual Geometry Group Hand Dataset [109]	✓	✓
Caltech 101 [50]		✓
Hand Dataset		
Training	8 000	8 000
Testing	2 222	2 484

6.3 Experimental setup

In the experimental setup, a single Logitech K190 webcam with a notebook consisting of the following specifications were used:

- Operating system: Ubuntu Karmic Koala
- Memory: RAM 2GB
- Processor: Intel® Dual Core™.

This setup was only used to capture the video data and provided portability to capture video in the different environments. The evaluations of the prototypes were performed on an Intel i7 desktop computer with the following specifications:

- Operating system: Ubuntu Karmic Koala
- Memory: RAM 3GB
- Processor: Intel® Core™ i7 CPU @ 3.20GHz

6.4 Ground-truth annotation

In order to provide an objective comparison between prototypes, a standard ground-truth tracking trajectory was used. In general, generating ground-truth data manually is a time-consuming process that has limited the amount of available datasets on which objective comparisons can be made.

In video-tracking analysis, three types of ground-truth exist, based on the accuracy of tracking results [104]:

1. Pixel-level ground-truth – The projection of a target in a video frame is based on selecting a particular pixel at a particular time.

2. State-level ground-truth – The state of a target in a video frame is based on assigning a value to the target state or on a subset of the target state parameters at a particular time.

3. Life-span ground-truth – The target accuracy in a video is based on assigning a time interval to a target that corresponds to the life span of the target.

To provide ground-truth data for the SASL dataset, pixel-level ground-truth was employed. The frames in each video were annotated by selecting the centre of each hand, first the right hand and then the left. The location of each hand was stored in a text file in the same directory as the frames for each signed gesture. This location was represented by the x and y coordinates for the centre of each hand.

If the hand was partially or totally occluded, as illustrated in Figure 6.6, the location of the hand, in the occluded state, was selected as the ground-truth position.

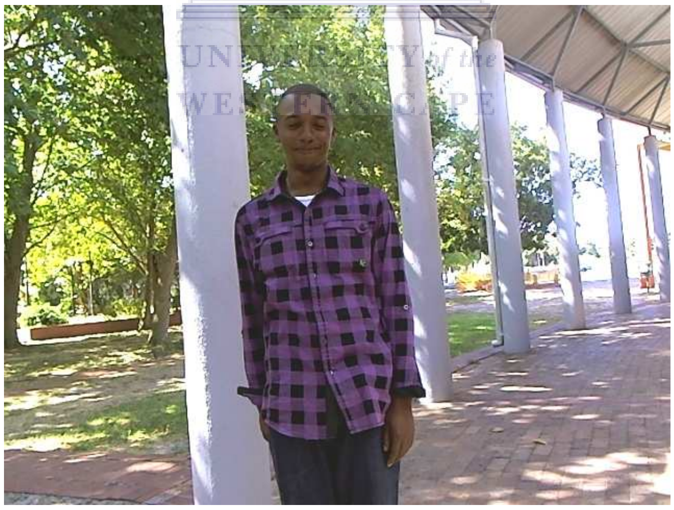


FIGURE 6.6: A frame in which the right hand is occluded by the participant's body.

To simplify the time-consuming process of generating ground-truth data, a common approach would be to record ground truth on a selected subset of frames in each image sequence that is of interest [104]. However, this approach was not implemented, since the SASL dataset provides ground-truth data for each frame in the dataset, which allowed for consistency and a comparison between signed gestures.

As with any ground-truth data, accuracy is especially important in the case of manual annotation. In order to verify its correctness, a blue circle for the right hand and a red circle for the left hand were drawn on each frame using the ground-truth coordinates, as depicted in Figure 6.7.

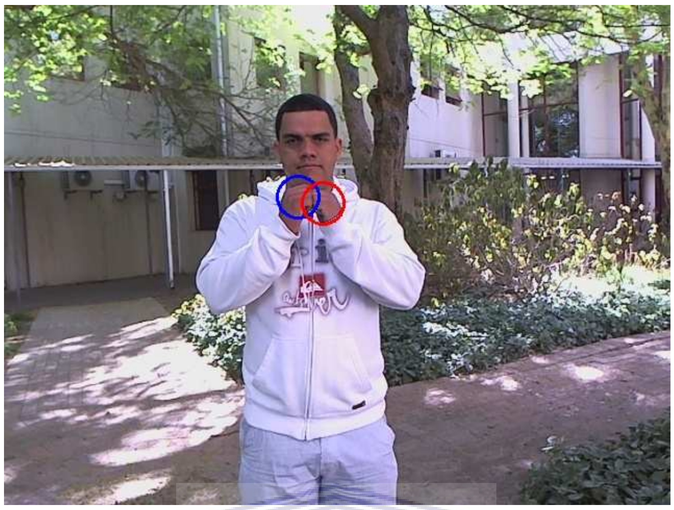


FIGURE 6.7: A frame in which the ground-truth coordinates have been verified where the blue circle represents the right hand and the red circle represents the left hand.

6.5 Performance evaluation metrics

In the following subsections the evaluation metrics are explained, where distance is used to determine the accuracy within a frame; tracking ratio is used to determine accuracy within a video; and the accuracy, precision and recall metrics are used to determine the correctness of hand classification in images.

6.5.1 Euclidean distance

For each prototype, the evaluation metrics were used to measure the accuracy of the tracking results. Different metrics evaluate different properties of a tracking system. In SASL it is important that the upper body of an individual can be seen, therefore each individual should be at a similar distance from the camera. For example, if an individual stands further away from the camera, the facial features that are very important in SASL would not be clearly visible. The tracking result does not depend on the target size, but rather on a point in the centre of the target. To objectively compare the tracking results between the prototypes, one solution would be to compute the distance between the centre point of the target and the ground-truth for the hands in each frame [104].

When evaluating the performance of multiple objects, an association should be drawn between the ground-truth states and the tracking result states [104]. Given that the ground-truth data in this study was structured by first stating the right-hand coordinates, followed by the left-hand coordinates, the tracking results can be computed as two single objects evaluated in that order.

Therefore, to measure the distance between the estimate target point, (x_1, y_1), and the ground-truth point, (x_2, y_2), of each hand, the Euclidean distance was used, defined as:

$$d(x, y) = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2} \tag{6.1}$$

In order to determine when a hand has been correctly located, a radius of 20 pixels from the center of a hand was used. When comparing the average accuracy of single sign performed by 10 individuals when using a radius of 10, 15 and 20 pixels, a standard deviation of 2.91% was achieved (refer to Table D.1 in Appendix D). Furthermore, as illustrated in Figure 6.8, a radius of 20 pixels (orange circle) is within the boundaries of a hand and therefore considered to be a valid threshold. Hence, for a distance greater than 20 pixels, it is assumed that the location of a hand has been incorrectly estimated and a *lost target* is recorded. When the target is lost for a long period of time, tracking is considered to have failed.

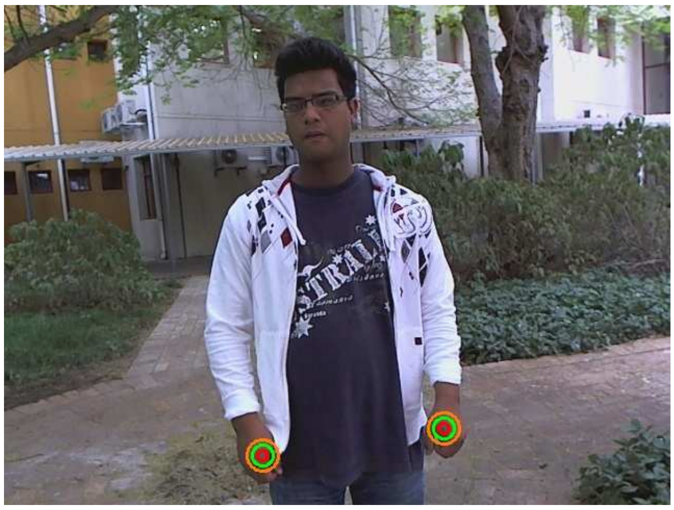


FIGURE 6.8: An illustration of a radius of 10 pixels (red circle), 15 pixels (green circle) and 20 pixels (orange pixels) from the center of the hand.

6.5.2 Tracking ratio

After determining the average of the tracking accuracies, the results can be used to compare the performance between the prototypes, the signs, the environments and the participants. When comparing, one can consider the tracking life span and the significance of the results [104].

The tracking life span can be determined by calculating the *tracking ratio*, λ, which is defined as:

$$\lambda = \frac{F_S}{F_T} \tag{6.2}$$

where F_S is the number of frames that were correctly tracked and F_T is the total number of frames in a video sequence. The tracking is therefore considered a success when λ is greater than 70%.

If the tracking performance results are not consistent over the entire dataset, a comparison of the average tracking results may be insufficient to reach a meaningful conclusion as to which prototype would be preferred. In this case, statistical hypothesis testing to determine the statistical significance between the results will be used.

6.5.3 Accuracy, precision and recall

In order to measure the accuracy of the hand-detection algorithm, which can be viewed as a binary classification problem, the results can be assessed based on the number of true positives (TP), false positives (FP), false negatives (FN) and true negatives (TN).

The number of successful and unsuccessful hand classifications is represented by true positives and false negatives, respectively, whereas the number of successful and unsuccessful non-hand classification is represented by true negatives and false positives, respectively. These values can be summarised in a confusion matrix as depicted in Table 6.2.

In order to compare the different models used, the accuracy, precision and recall can be determined based on the TP, FP, TN and FN. The accuracy, A, is the ratio of the

TABLE 6.2: The confusion matrix of a binary classification problem

		Predicted class	
		1	0
Actual class	1	TP	FN
	0	FP	TN

number of correct predictions with respect to the total number of predictions and defined as:

$$A = \frac{TP + TN}{TP + FP + TN + FN} \tag{6.3}$$

The precision, P, is the ratio of the number of correct hand classifications with respect to the number of predicted hand classifications.

$$P = \frac{TP}{TP + FP} \tag{6.4}$$

The recall, R, is the ratio of the number of correct hand classifications with respect to the number of ground-truth hand images.

$$R = \frac{TP}{TP + FN} \tag{6.5}$$

6.6 Results and discussion

Each prototype was evaluated using the SASL dataset evaluation protocol. The aim of the analysis was to identify patterns and trends in the results and determine how they relate to factors such as the type of signs, people and environment that affect these prototypes. In the following subsections the evaluation of each prototype is analysed where prototype T consisted of the tracking phase only; prototype DT consisted of both the tracking and detection phases; and prototype DTL consisted of the tracking, detection and learning phases.

6.6.1 Tracking evaluation

In the following section, prototype T is evaluated to determine how well the tracking phase performs as a hand-tracking framework.

6.6.1.1 Prototype T

In Figure 6.9 a graphical representation of the results of the tracking success rates of prototype T for each video of a signed gesture performed by the participants is shown (see Appendix D, Table D.2 for the detailed results).

TABLE 6.3: Summary of the average hand-tracking success rates of prototype T for each signed gesture in both environments.

Single hand	**Both hands without occlusion**	**Both hands with occlusion**	**Success rate**
Sign 3	Sign 11		Result $\geq$ 80%
Sign 1, 4, 5, 6, 7, 8, 10	Sign 14, 15, 19		70% $\leq$ Result $<$ 80%
Sign 2, 9	Sign 13, 16, 17, 18	Sign 21, 22, 23, 24, 25, 26, 27, 29	60% $\leq$ Result $<$ 70%
	Sign 12, 20	Sign 28, 30	Result $<$ 60%

The results in Figure 6.9 represent the average hand-tracking success rate of each signed gesture in both environments. In Table 6.3 the results indicate that 12 signs obtained an average success rate greater than 70% with sign 3 and sign 11 obtaining success rates greater than 80%. These results were the set of signs that involved a single hand and both hands without occlusion. In addition, 14 signs obtained an average success rate of between 60% and 70%, with six of those signs obtaining an average success rate very close to 70%. These results were the set of signs that involved both hands with and without occlusion, in which more than 50% of these signs involved occlusion. Therefore 60% of the signs obtained an average success rate close to or more than 70%. Furthermore, only four signs obtained an average success rate below 60%.

Since prototype T forms the foundation of the hand-tracking framework, the results presented were very encouraging. Overall, the comparison between the different signed gestures indicates that tracking two hands while distinguishing between them increases the complexity of hand-tracking. Furthermore, it also indicates that occlusion between the hands affects hand-tracking and is one of the main reasons for hand-tracking failure.

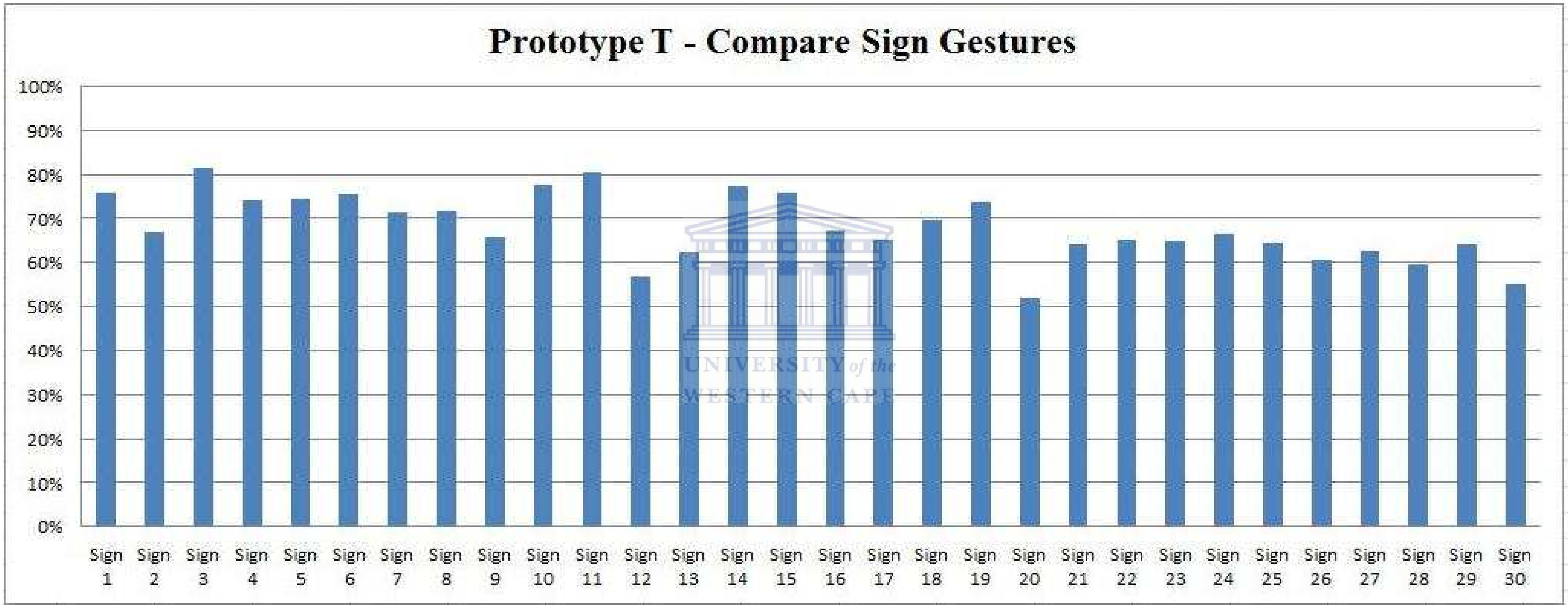


FIGURE 6.9: The average hand-tracking success rates of prototype T for each signed gesture in both environments.

In addition to these factors, the assumption in prototype T that the object being tracked is a hand could be a factor resulting in lower results. This assumption is investigated in the following section.

When comparing how well the hands were tracked in the different environments using prototype T (as seen in Figure 6.10), the results indicate that 17 signs obtained a higher result in constrained environments when compared to unconstrained environments, which suggests that the background in unconstrained environments negatively affects hand-tracking. The results also indicate that 70% of the signs using a single hand were higher in a constrained environment than an unconstrained environment in terms of the average success rate, while half of the signs using both hands were the same in either environment in terms of average tracking success rate (refer to Table D.2 in Appendix D). This signifies that the movement of two hands is more likely to be affected by moving objects in the background when compared to single hand movements.

In terms of accuracy when considering the signing of the participants, for more than half of the participants the prototype's average tracking success rate was greater than 70% (see Figure D.1 in Appendix D) and for four of the participants the average success rate of prototype T was greater than 80%. Furthermore, four participants obtained an average success rate of between 60% and 70%, and five participants obtained an average tracking success rate of below 60%. Overall, the results show an equal variation between the different body types and skin-colour tones. The hand-tracking results can therefore not be attributed to the participants that performed the gestures, but rather to other factors such as the moving objects in the background, light inconsistencies in a frame and the complexities of gestures using both hands.

When comparing related systems (with respect to hand-tracking) to prototype T, the comparison is restricted to passive approaches that use 2D video input streams. It is also restricted to those that followed an appearance-based method. Many of these related systems have opted for a subjective evaluation; however, those that have pursued an objective evaluation have evaluated their systems using their personal datasets.

In comparison to the single hand-tracking systems (in simple environments) of Rautaray and Agrawal [128], Coogan et al. [36] and Fogelton [54], prototype T provides a more accurate approach. It achieves above 70% for 80% of the total number of signs using a single hand and is able to handle occlusion between the face and hands. Even though

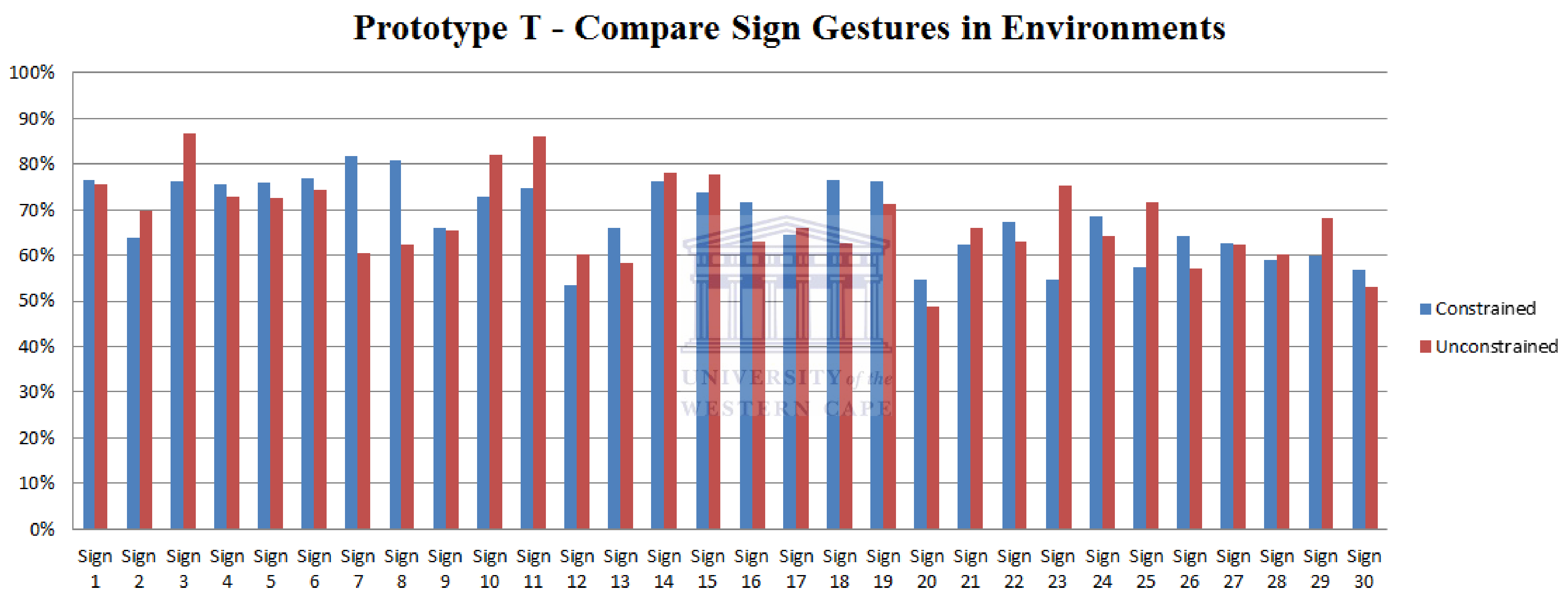


FIGURE 6.10: A comparison of the average hand-tracking success rates of prototype T for each signed gesture in constrained and unconstrained environments.

Liu et al. [97] improved the flocks-of-features algorithm of Fogelton [54] in terms of distractors in the background, their hand-tracking is still negatively affected when moving the tracked hand in front of the face, whereas this scenario is successfully handled using prototype T.

When comparing prototype T to the particle-filter frameworks of Liu and Zhang [98], Spruyt et al. [144], Ongkittikul et al. [120] and Chen et al. [32], it shows a comparable performance in terms of tracking both hands. In addition, it is able to distinguish between the right and left hand when compared to Liu and Zhang [98] and Ongkittikul et al. [120]. Moreover, prototype T extended the algorithm of Argyros and Lourakis [10], but with several improvements: it is able to distinguish between the right and left hand, it does not regard the hand as a different object when it recovers from tracking failure, and it is able to track both hands in unconstrained environments.

6.6.2 Detection and tracking evaluation

In order to determine how well a hand-detection algorithm would perform in a hand-tracking framework, different hand-detection algorithms are compared, with the aim of identifying the most suitable one. This is followed by an evaluation of prototype DT to determine how well the integration of the detection and tracking phases performs as a hand-tracking framework.

6.6.2.1 Hand detection

In order to determine an appropriate hand-detection algorithm for prototypes DT and DTL, a comparison is made between using a random forest model and an SVM model. It investigates whether an algorithm that filters the image region to only extract features from the skin-coloured pixels on the hands would improve hand detection using LBPs. It also investigates and compares between the original LBP, extended LBP and the uniform LBP computed with either the LBP image features, spatially enhanced histogram features and global histogram features. The algorithms are evaluated on the hand dataset using the classification metrics — accuracy, precision and recall.

6.6.2.1.1 Random forests

Before evaluating the random forest approach, three parameters, namely, the maximum number of trees, tree depth and the number of random-split variables, need to be determined. To determine the best parameters, different combinations of these parameters were used. The experiment began using the default parameters used for random forests in OpenCV[2]: a maximum number of 100 trees, a tree depth of 25 and a random split variable of 4. These parameters were then incrementally tuned by multiplying the maximum number of trees and the number of random-split variables by the power of two, and by multiplying the tree depth by two for each increment. The parameters were evaluated on the original LBP using global histogram features (see Figure 6.11).

The different combinations illustrate that the optimal parameters are reached when the maximum number of trees, the tree depth and the number of variables are 100, 50 and 64, respectively. Increasing the parameters any further results in a decrease in accuracy.

Using these parameters, the different LBP algorithms were evaluated on the original images in the dataset (experiment 1) and compared to the same images that were filtered based on skin-coloured pixels only (experiment 2). Figure 6.12 presents the results of both experiments using the different algorithms. The results show that extracting LBPs using global and spatially enhanced histogram features on skin-coloured pixels only achieves a consistently higher result than extracting these features on all pixels in the image. On the other hand, extracting LBPs using LBP image features on skin-coloured pixels in the image achieves a lower result than extracting these features on all the pixels in an image, except when applying the extended LBP algorithm. It can therefore be determined that extracting LBP features from skin-coloured pixels in an image to detect hands provides a higher accuracy. The comparison between LBP histogram features and LBP image features also shows that a higher accuracy is achieved when using LBP histogram features.

When comparing extracting LBPs using global and spatially enhanced histogram features, the results show that a consistently higher result is obtained using global histogram features. In order to determine a suitable LBP variant, a comparison is made between

[2]OpenCV is an open source computer vision and machine-learning software library, see www.opencv.org

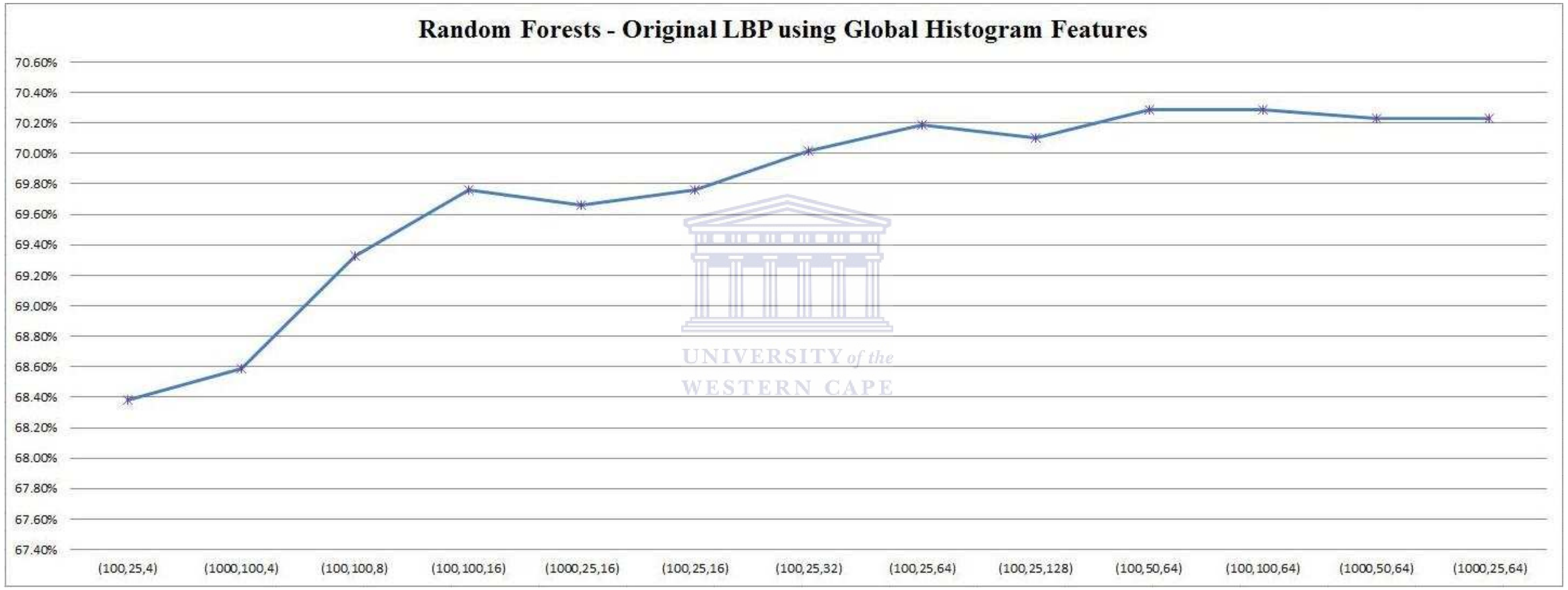


FIGURE 6.11: Evaluating different random forest parameters.

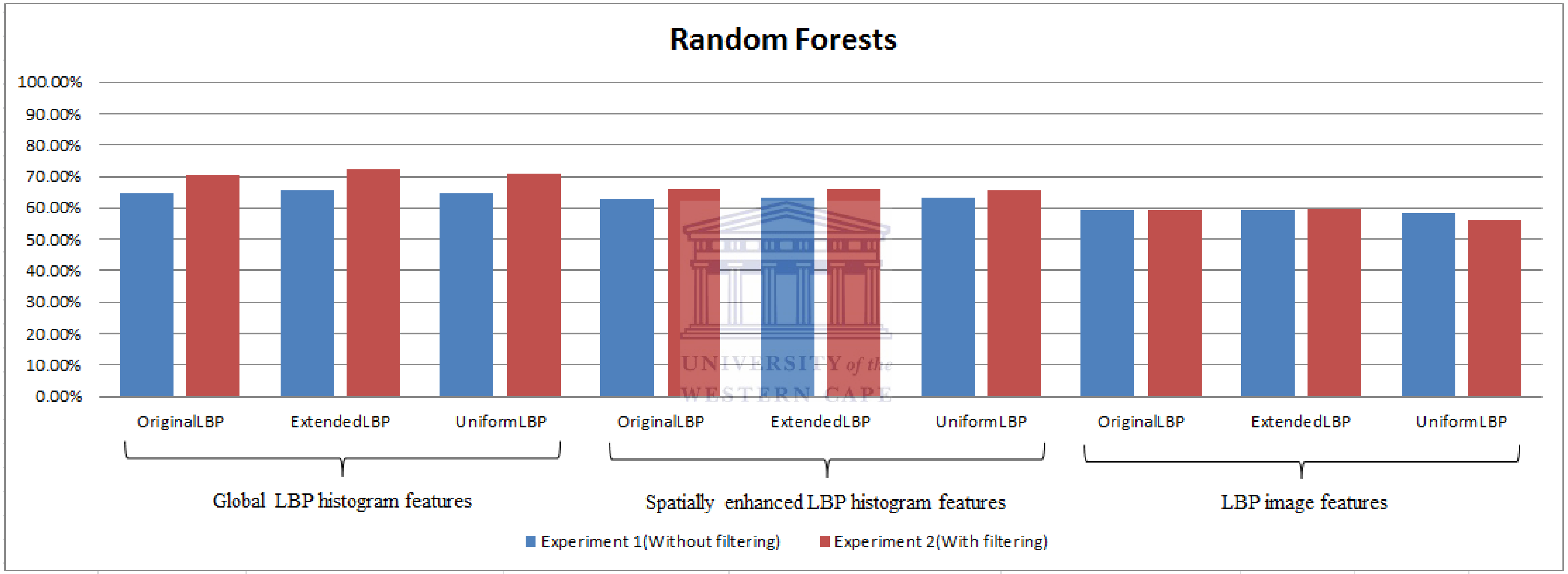


FIGURE 6.12: Evaluating different LBP algorithms on the original images in the dataset (experiment 1) and compared to the same images that were filtered based on skin-coloured pixels only (experiment 2), using a random forest model.

the original, extended and uniform LBP using global histogram features. This comparison reveals that a higher accuracy is obtained using the extended LBP with an accuracy of 72,25%, a precision of 0,658 and a recall of 0,855.

6.6.2.1.2 Support vector machines

With the random forest approach, an appropriate kernel with optimal parameters is required, since it greatly influences the predictive capabilities of an SVM [67, 75]. However, a standard method to find the most appropriate kernel does not exist [170], and finding an appropriate kernel is therefore a process of trial and error [37]. The four basic kernels, namely, linear, RBF, polynomial and sigmoid kernels, were compared to find the most appropriate one for further experimentation.

Each kernel was trained on the hand dataset discussed in Section 6.2.3 and the optimal parameters for each kernel were determined using a cross-validation strategy. These kernels were evaluated on the original LBP using global histogram features (see Figure 6.13).

A comparison between the different kernels shows that the RBF kernel outperforms the other kernels with an accuracy of 67,40% and is therefore the most appropriate kernel to use.

Using the RBF kernel, the SVM models were trained and evaluated on the original images in the dataset (experiment 3) and compared to the same images that were filtered based on skin-coloured pixels only (experiment 4). Features used to train and test the models from these images were extracted using different LBP algorithms (see Figure 6.14).

The results show that using LBPs combined with an SVM on skin-coloured pixels achieves a higher result than extracting these features on all pixels in an image, except when using LBP image features and when using the uniform LBP with global histogram features. Therefore, similar to the outcome of random forests, extracting LBP features from skin-coloured pixels in an image to detect hands yields a higher accuracy. When comparing LBP histogram features and LBP image features, a comparable performance

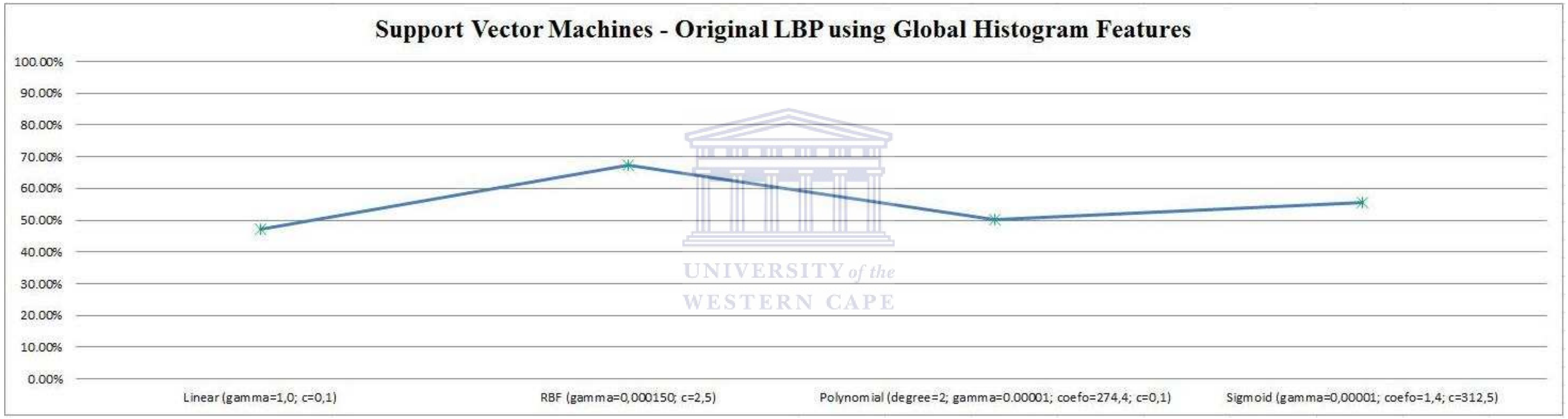


FIGURE 6.13: Evaluating and comparing different SVM kernels.

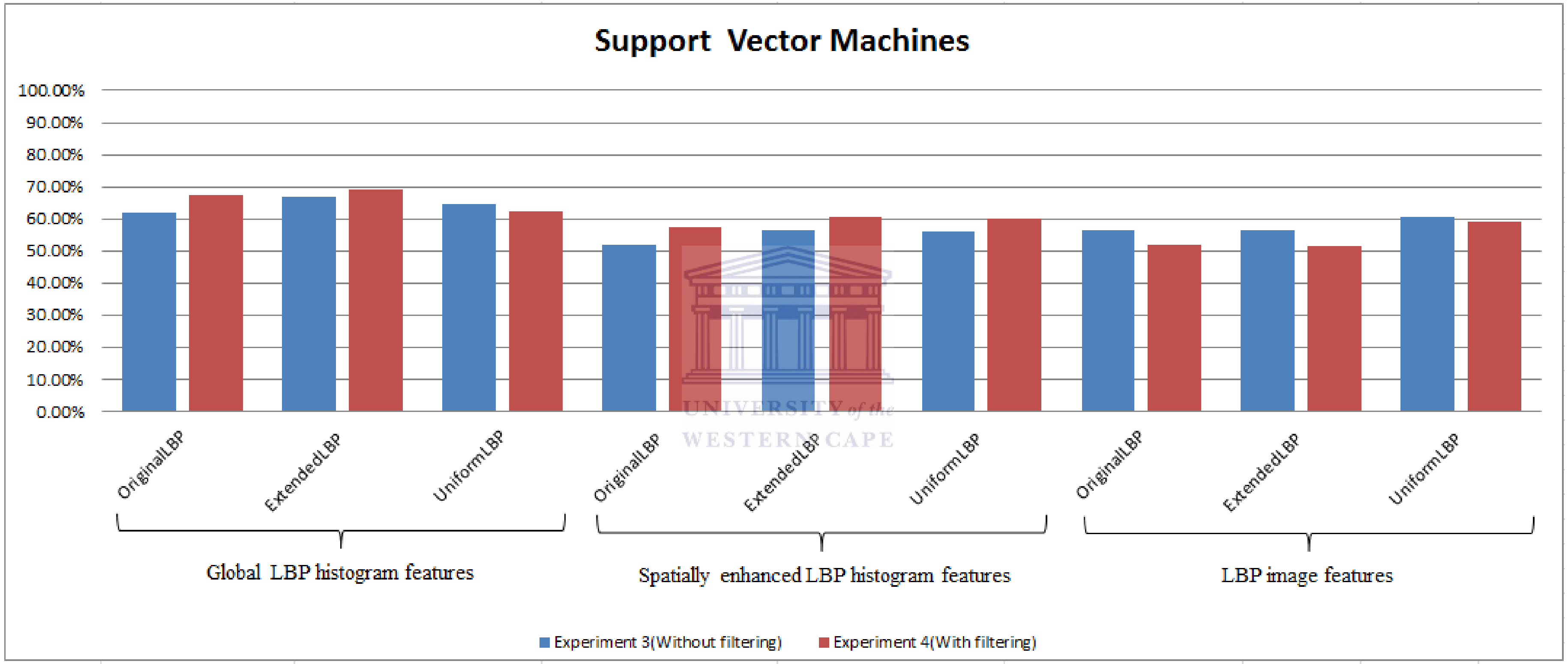


FIGURE 6.14: Evaluating different LBP algorithms on the original images in the dataset (experiment 3) and compared to the same images that were filtered based on skin-coloured pixels only (experiment 4), using an SVM model.

is obtained when using the spatially enhanced LBP histogram features and LBP image features, and a higher accuracy is achieved when using the global LBP histogram features. The latter comparison is consistent with the results of the random forest approach and it is therefore concluded that global LBP histogram features should be used for hand detection.

In order to determine a suitable LBP algorithm, a comparison was made between the original, the extended and the uniform LBPs, using global histogram features. This comparison revealed that a higher accuracy is obtained using the extended LBP, with an accuracy of 69,04%, a precision of 0,63 and a recall of 0,81, which again is consistent with the comparison made using random forests. It can thus be concluded that when using SVMs, the extended LBP with global histogram features would be more suitable.

6.6.2.1.3 Comparison between hand-detection systems

In order to determine the best method to use for hand detection, a statistical analysis was performed to compare the methods. The comparison considered four factors, namely:

1. A comparison between random forests or SVMs;
2. A comparison between extracting features from skin-coloured pixels only (filtering) or extracting features from all pixels in an image (no filtering);
3. A comparison between global LBP histogram features, spatially enhanced LBP histogram features and LBP image features; and
4. A comparison between the original, uniform and extended LBP.

When taking these factors into consideration, 36 different hand-detection methods were created. The statistical analysis was performed using a generalized linear model accounting for the dependencies inherent in having the same image classified by each of the 36 different methods. A compound symmetric structure was used to model the dependencies. The analysis was done using the GENMOD procedure in SAS[3] version 9.

[3]SAS Institute Inc. Cary, NC, USA

Table 6.4: Comparison between different hand-detection methods considering all four factors.

Observation	Factor 1	Factor 2	Factor 3	Factor 4	Combination	P-value
1	Random forest	Filtering	Global	Extended	1212	0.723
2	Random forest	Filtering	Global	Uniform	1213	0.70578
3	Random forest	Filtering	Global	Original	1211	0.70302
4	SVM	Filtering	Global	Extended	2212	0.69048
5	SVM	Filtering	Global	Original	2211	0.67389
6	SVM	No filtering	Global	Extended	2112	0.66922
7	Random forest	Filtering	Local	Extended	1222	0.65965
8	Random forest	Filtering	Local	Original	1221	0.65838
9	Random forest	No filtering	Global	Extended	1112	0.65668
10	Random forest	Filtering	Local	Uniform	1223	0.65625
11	SVM	No filtering	Global	Uniform	2113	0.6469
12	Random forest	No filtering	Global	Original	1111	0.64094
13	Random forest	No filtering	Global	Uniform	1113	0.64052
14	Random forest	No filtering	Local	Uniform	1123	0.63159
15	Random forest	No filtering	Local	Extended	1122	0.62946
16	Random forest	No filtering	Local	Original	1121	0.6233
17	SVM	Filtering	Global	Uniform	2213	0.62245
18	SVM	No filtering	Global	Original	2111	0.61777
19	SVM	Filtering	Local	Extended	2222	0.60374
20	SVM	No filtering	Image	Uniform	2133	0.60332
21	Random forest	No filtering	Image	Extended	1132	0.60098
22	SVM	Filtering	Local	Uniform	2223	0.59949
23	Random forest	Filtering	Image	Extended	1232	0.59779
24	Random forest	No filtering	Image	Original	1131	0.59609
25	Random forest	Filtering	Image	Original	1231	0.59247
26	SVM	Filtering	Image	Uniform	2233	0.59056
27	Random forest	No filtering	Image	Uniform	1133	0.58503
28	SVM	Filtering	Local	Original	2221	0.57207
29	SVM	No filtering	Image	Original	2131	0.56611
30	SVM	No filtering	Local	Extended	2122	0.56526
31	SVM	No filtering	Image	Extended	2132	0.56441
32	SVM	No filtering	Local	Uniform	2123	0.56144
33	Random forest	Filtering	Image	Uniform	1233	0.55527
34	SVM	Filtering	Image	Original	2231	0.51892
35	SVM	No filtering	Local	Original	2121	0.51658
36	SVM	Filtering	Image	Extended	2232	0.51594

When sorting the methods according to the estimated probability of predicting a hand correctly, several patterns emerged, as shown in Table 6.4. The results show that the top six best-performing methods use the global LBP histogram features, followed by some methods that use spatially enhanced LBP histogram features and 12 of the last 17 methods that employ LBP image features. The results also show that 8 of the top 10 methods extract features from skin-coloured pixels only, while 6 of the last 10 methods extract features from all pixels in the image. In addition, it shows that 12 of the top 16 methods use random forests, while 10 of the last 16 methods use SVMs. Moreover, 5 of the top 10 methods use the extended LBP when compared to the original and uniform LBPs.

In terms of statistical significance, the accuracy of the top three methods do not differ significantly from each other; however, using the extended LBP achieves a higher accuracy. In Appendix D a list of the non-significant differences between the different methods is given. When comparing the accuracy between random forests and SVMs using the extended LBP with global histogram features on skin-coloured pixels, the results were statistically significant. It was therefore established that random forests using the extended LBP with global histogram features on skin-coloured pixels will be a better approach than SVMs for the hand-detection algorithm. Hereafter, the hand-detection algorithm will be referred to as the ELBP-RF hand-detection algorithm.

When comparing related systems to the ELBP-RF hand-detection algorithm used in prototypes DT and DTL, the comparison is restricted to 2D views. For the subjective evaluation, the output was described and compared. For the objective evaluation, the output was compared in terms of its accuracy relative to the dataset.

When compared to the hand-detection system of Petersen and Stricker [125], which is limited to open hand postures where the fingers are visible; the ELBP-RF hand-detection algorithm successfully detects hands in any hand posture. According to Thangali and Sclaroff [151], extracting HoG features from the entire image region can be used to detect hands in images. However, the ELBP-RF hand-detection algorithm was used to compare using the features from the entire image region and from features on the hands only, and showed that by only using features on the hand, better results can be obtained (see Tables D.5 and D.7).

In studies that have used LBPs for hand detection, for example, Xiao et al. [166], which used the original LBP on a small dataset of 482 images, and Nguyen et al. [116], which used a simplified version of the LBP on a small dataset of approximately 673 frames, a higher average accuracy was achieved on the limited datasets. Furthermore, their systems were trained and tested on a limited set of hand postures, thus contributing to a higher accuracy.

In the hand-detection system of Mittal et al. [109], a recall rate of 85,3% was achieved on a larger dataset of 5 628 images, which is comparable to the recall rate of 85,5% achieved using the ELBP-RF hand-detection algorithm. Moreover, Mattheij and Postma [106] used a random forest model based on Haar-like features where they obtained an accuracy, recall and precision rate of 69,5%, 78,9% and 66,6%, respectively, on a testing sample of 320 images. In comparison to their results, the ELBP-RF hand-detection algorithm achieved a higher accuracy and recall rate of 72,25% and 85,5%, respectively, and a comparable precision rate of 65,8%.

Therefore, compared to related studies, the ELBP-RF hand-detection algorithm can be considered state-of-the-art.

6.6.2.2 Prototype DT

Prototype DT builds on the framework of prototype T and includes the ELBP-RF hand-detection algorithm. This prototype was used to investigate how well the ELBP-RF hand-detection algorithm would perform in a hand-tracking framework.

For a complete list of tracking success rates for each video of a signed gesture performed by the participants, refer to Table D.8 in Appendix D. In Figure 6.15 a graphical representation of the average tracking success rates for each sign in both environments is shown.

These results, listed in Table 6.5, indicate that 13 signs — which make up almost half of the total number of signs — obtained an average success rate greater than 80%, where 8 of these signs were performed with a single hand and 5 signs were performed with both hands without occlusion. In addition, 6 of the signs obtained an average success rate of between 70% and 80%, with 3 of the signs belonging to the set of signs involving both hands without occlusion. Moreover, 8 signs obtained an average success rate of between

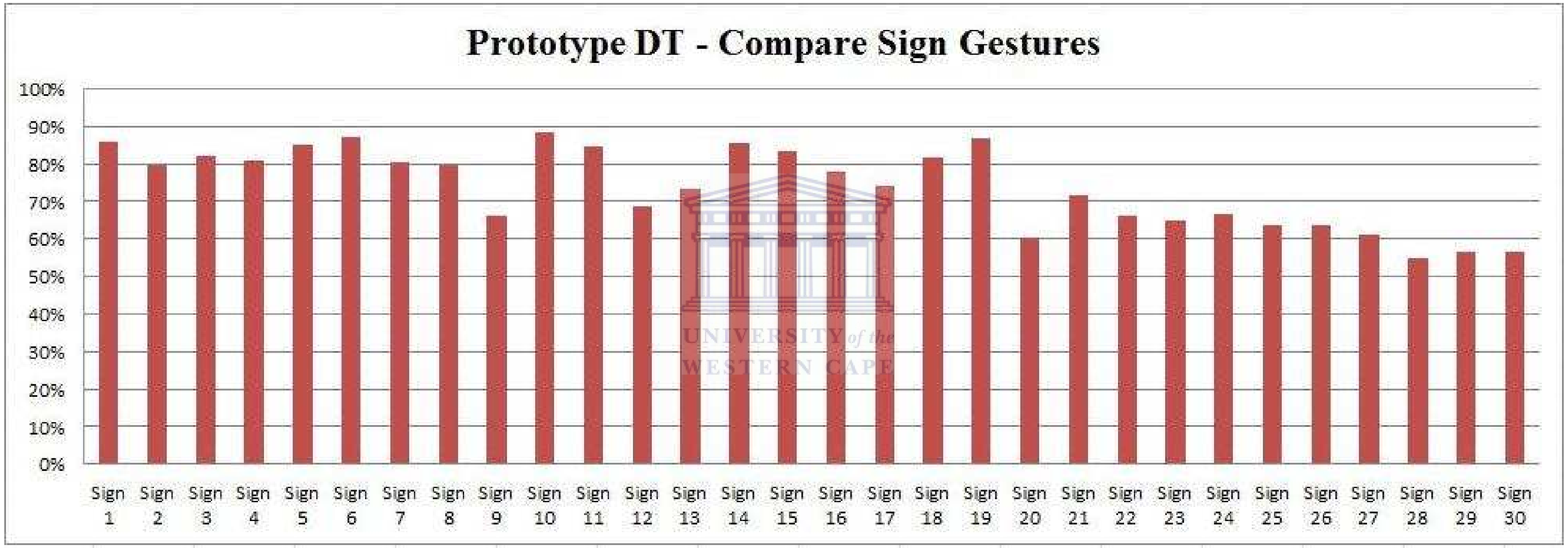


Figure 6.15: The average hand-tracking success rates of prototype DT for each signed gesture in both environments.

Table 6.5: Summary of the average hand-tracking success rates of prototype DT for each signed gesture in both environments.

Single hand	**Both hands without occlusion**	**Both hands with occlusion**	**Success rate**
Sign 1, 3, 4, 5, 6, 7, 10	Sign 11, 14, 15, 18, 19		Result $\geq$ 80%
Sign 2, 8	Sign 13, 16, 17	Sign 21	70% $\leq$ Result $<$ 80%
Sign 9	Sign 12, 20	Sign 22, 23, 24, 25, 26, 27	60% $\leq$ Result $<$ 70%
		Sign 28, 29, 30	Result $<$ 60%

60% and 70%, with 6 of the signs belonging to the set of signs involving both hands with occlusion. Furthermore, only 3 signs obtained an average tracking success rate below 60%.

Overall, the results showed a higher accuracy on signs where the right and left hand were separated from each other and where occlusion did not occur. This indicates that the hand-detection component may have been negatively affected when the hands crossed one another and when occlusion occurred, which can be attributed to the fact that samples where the hands cross each other were not part of the hand dataset. Furthermore, the comparison between the different signed gestures using prototype DT shares the same outcome as prototype T, in that the tracking of both hands while distinguishing between them increases the complexity of hand-tracking. Moreover, it shares the outcome that occlusion between the hands has a negative impact on hand-tracking, especially when distinguishing between the hands after occlusion has occurred. In addition to these factors, the variation between the tracking accuracies of the signs is indicative of occurrences when the tracking of a hand is 'lost' and not recovered. The following section, investigates whether recovery from such failures would improve the hand-tracking framework.

When comparing the accuracy of hand-tracking in different environments using prototype DT (as seen in Figure 6.16), the results show that 18 signs obtained a higher result in unconstrained environments compared to constrained environments. Furthermore, the results show that an equal number of signs using a single hand were successfully tracked in both environments, while results for signs using both hands with and without occlusion were higher in unconstrained environments than constrained environments. This indicates that these results are not attributed to the environment in which the hands were tracked, but rather to the participants who performed the gestures.

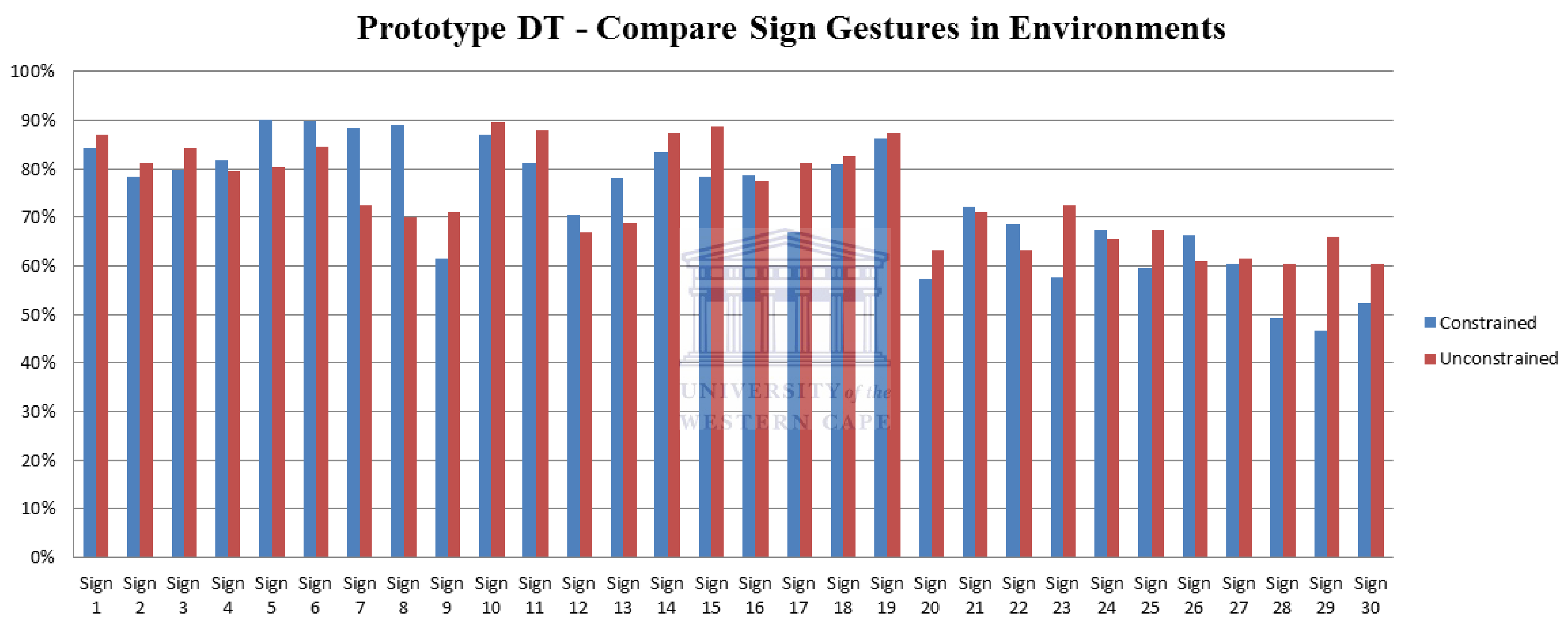


FIGURE 6.16: A comparison of the average hand-tracking success rates of prototype DT for each signed gesture in constrained and unconstrained environments.

In terms of accuracy according to the participants, 14 participants obtained an average tracking success rate greater than 70%, while half of these participants obtained an average success rate greater than 80%. Furthermore, 3 participants obtained and average tracking success between 60% and 70%, and 3 participants had an average success rate below 60%. Figure D.3 shows that only participants B, K, O and R, who obtained accuracies below 70%, have much lower accuracies in a constrained environment when compared to unconstrained environments. These accuracies are consistent with those achieved using prototype T and therefore suggest that factors such as motion blur caused by these participants or colour and motion differences relating to camera properties could have contributed to predicting the hands as non-hands at these specific times, thus leading to the hands not being tracked.

6.6.3 Detection, tracking and learning evaluation

In the following section, prototype DTL is evaluated to determine how well the integration between the detection, tracking and learning phases performs as a hand-tracking framework.

6.6.3.1 Prototype DTL

In order to assist hand-tracking and recover from tracking failure, a learning algorithm was included in the framework to form prototype DTL. The prototype was used to investigate how well a hand-tracking framework would perform when including a learning algorithm.

For a complete list of tracking success rates for each video of a signed gesture performed by the participants, refer to Table D.2 in Appendix D. In Figure 6.17 a graphical representation of the results of the average tracking success rates for each sign is shown.

The results listed in Table 6.6 indicate that 18 signs obtained an average tracking success rate greater than 80%, with sign 14 and sign 19 greater than or equal to 90%. Nine signs obtained an average success rate between 70% and 80%, and only 3 signs obtained an average tracking success rate between 60% and 70%. This indicates that 90% of the total number of signs obtained an average success rate greater than 70%. Furthermore,

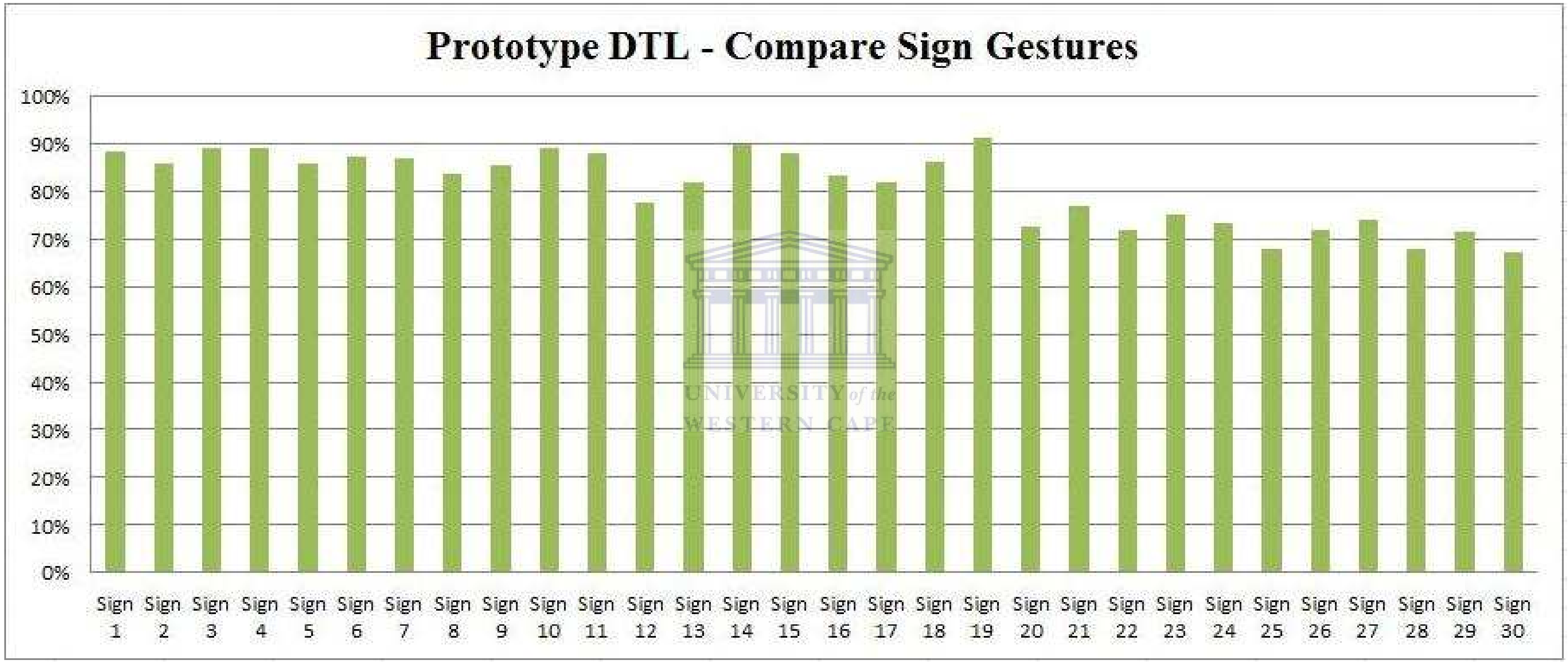


FIGURE 6.17: The average hand-tracking success rates of prototype DTL for each signed gesture in both environments.

Table 6.6: Summary of the average hand-tracking success rates of prototype DTL for each signed gesture in both environments.

Single hand	**Both hands without occlusion**	**Both hands with occlusion**	**Success rate**
Sign 1, 2, 3, 4, 5, 6, 7, 8, 9, 10	Sign 11, 13, 14, 15, 16, 17, 18, 19		Result $\geq$ 80%
	Sign 12, 20	Sign 21, 22, 23, 24, 26, 27, 29	70% $\leq$ Result $<$ 80%
		Sign 25, 28, 30	60% $\leq$ Result $<$ 70%
			Result $<$ 60%

it shows that these accuracies were obtained in all three categories — signs performed with a single hand, both hands without occlusion and both hands with occlusion.

Moreover, the results showed a consistent accuracy greater than 80% for signs using a single hand and for signs using both hands without occlusion. This suggests that features belonging to each hand are extracted more distinctively when the hands are separated. The results also showed a consistent accuracy greater than 70% for signs using both hands with occlusion, which is slightly lower than the other signs. This suggests that when the hands cross each other, features are extracted from the area in which the hands overlap, thereby storing these features in both the right and left hand databases. This may affect the system when distinguishing between each hand after occlusion has occurred; however, it remains effective when recovering from tracking failure, leading to consistently higher accuracies compared to prototypes T and DT.

When comparing the accuracies according to the different environments in which the gestures were performed (see Figure 6.18), the results indicate that 18 signs obtained a higher result in constrained environments when compared to unconstrained environments. These results confirm that the background in unconstrained environments negatively affects hand-tracking; however, hand-tracking is improved across all signs when including a learning algorithm and the ELBP-RF hand-detection algorithm. Furthermore, the results suggest that features on the body assist hand-tracking better than features around the body, especially when the features around the body are extracted in unconstrained environments.

When comparing the accuracy according to the participants (see Figure D.5), 18 of the participants obtained an average tracking success rate greater than 70%, with 13 of those participants obtaining an average tracking success rate greater than 80% and

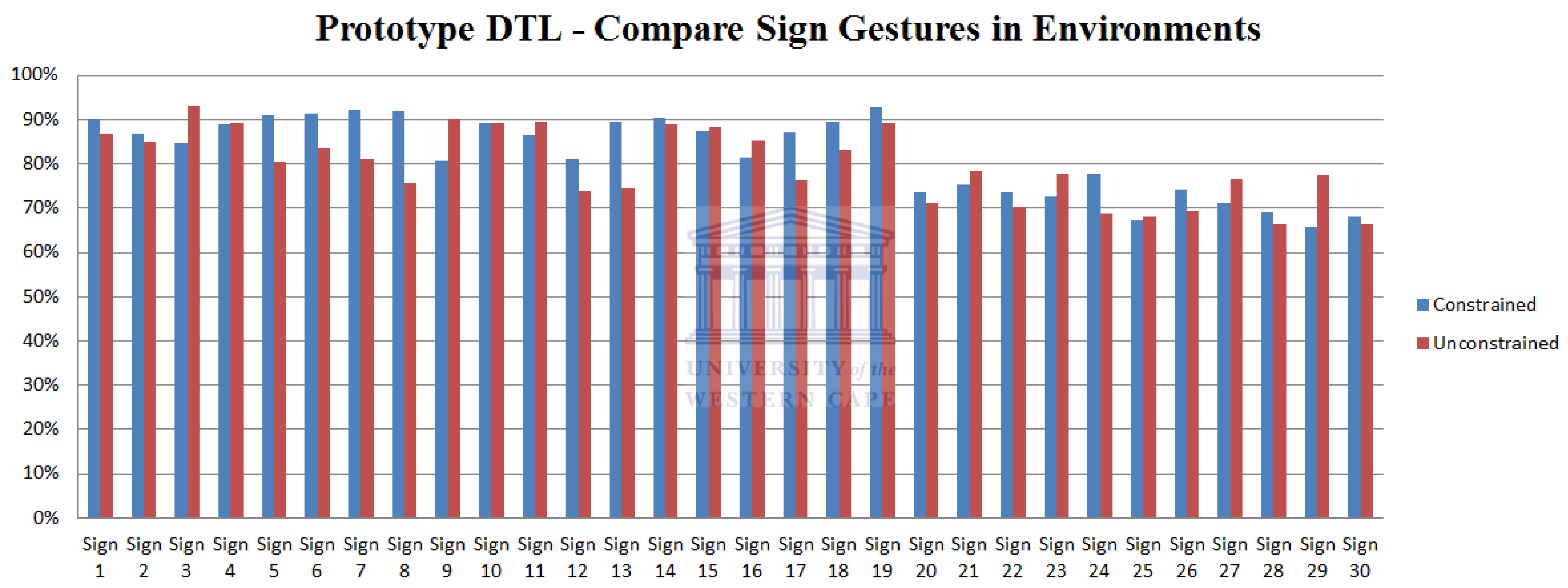


FIGURE 6.18: A comparison of the average hand-tracking success rates of prototype DTL for each signed gesture in constrained and unconstrained environments.

participant D obtaining the highest accuracy of 91,18%. Overall, using prototype DTL, the results show an equal variation between different body types and skin-colour tones. It furthermore shows that when tracking failure occurs due to motion blur caused by the participant, then the hand-tracking system would effectively recover from such failures.

Although the learning phase in prototype DTL employs context-based tracking, a comparison with related work in this study cannot be made, since related studies employed context-based tracking with a different approach.

Applying context-based information in hand-tracking is therefore a unique approach. Furthermore, applying to multiple objects, such as hands, has not yet been explored and is therefore a definite contribution. In general, the spatial and temporal contextual information used has been shown to considerably improve hand-tracking, similar to using contextual information in single-object tracking, as shown in the work of Wen et al. [163], Cerman et al. [26] and Grabner et al. [64].

6.6.4 Comparing prototypes T, DT and DTL

In order to determine whether any improvements were gained by integrating additional phases into the framework, the three prototypes are compared.

The comparison, as illustrated in Figure D.8, between prototype T and DT reveals that on the signs using a single hand, signs using both hands without occlusion, and some signs using both hands with occlusion, an improvement is shown when integrating the detection phase into the framework. These results indicate that validating the object as a hand improves the hand-tracking system, thereby only tracking what is considered to be a hand. However, on five signs — signs 24, 25, 27, 28 and 29 — that contain the use of both hands with occlusion, using prototype T has a higher accuracy than using prototype DT. These signs include the interaction between the two hands, and therefore a lower accuracy is expected, since the interaction between hands was not included in the hand dataset. Howver, this accuracy is improved significantly in prototype DTL when integrating the learning phase, which shows that the tracking of the hands is recovered after it has failed. The comparison between prototype T and DT also shows that for 83% of all signs, a higher tracking accuracy was obtained using prototype DT.

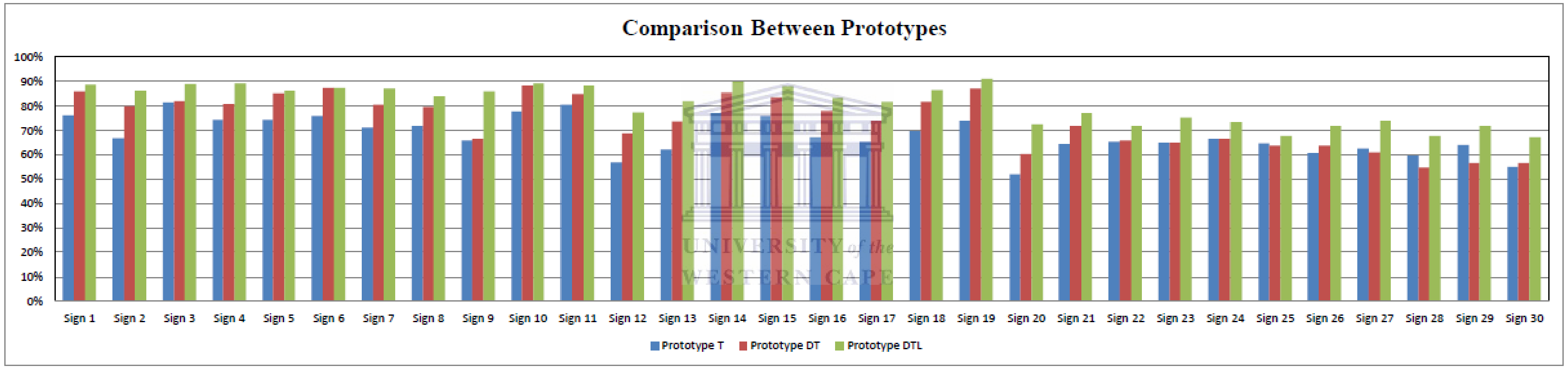


FIGURE 6.19: A comparison between the hand-tracking accuracies of each prototype.

The comparison between prototypes T, DT and DTL reveals that for each sign a significantly higher tracking accuracy is shown using prototype DTL compared to the other prototypes, as depicted in Figure D.8. Furthermore, the average of the total accuracy across all signs is 68,05%, 73,89% and 80,96% for prototypes T, DT and DTL, thus indicating that a significant increase in hand-tracking accuracy is obtained by integrating the detection and learning phase with the tracking phase in the hand-tracking framework. The results therefore suggest that in order to obtain a high hand-tracking accuracy across all signs, it would be best to use prototype DTL.

The comparison across prototypes in a constrained environment (refer to Appendix D) reveals that prototype T obtained a higher accuracy than prototype DT in only six signs, thereby indicating that the detection phase does improve hand-tracking accuracy in constrained environments. In addition, prototype DTL obtained a higher accuracy than both prototype T and DT for all the signs, thus showing that the integration of all three phases greatly improves hand-tracking accuracy in constrained environments.

The comparison across prototypes in an unconstrained environment (refer to Appendix D) reveals that prototype T achieved a higher accuracy than prototype DT for only five signs. This indicates that on these signs the ELBP-RF hand-detection algorithm may have incorrectly identified certain background objects as hands or on some occasions identified a hand as a non-hand object. This occurrence appears most often on signs involving interaction between the hands. However, on most signs it is shown that validating an object using the ELBP-RF hand-detection algorithm improves hand-tracking accuracy.

Furthermore, the results also reveal that on five signs, a slightly higher accuracy is obtained using prototype DT when compared to prototype DTL and on only one sign a slightly higher accuracy is obtained using prototype T when compared to using prototype DTL in unconstrained environments. This suggests that background features, especially those that have moved and which provided no contextual information about the hands, may have been incorrectly stored in the hand-feature database, leading to hand-tracking failures. On the other hand, 24 signs, which form 80% of all signs, obtained a higher accuracy using prototype DTL when compared to prototypes T and DT. Moreover, this accuracy was demonstrated for 90% of the signs using both hands with occlusion, suggesting that the contextual information learnt from features on and around the hand have assisted recovery from tracking failure and distinguishing between the right and left hand after occlusion has occurred.

The average of the total accuracy across all signs reveals that a higher tracking accuracy of 82,08% is achieved using prototype DTL in a constrained environment than using prototype DTL in an unconstrained environment, which had an average of 79,83%. This suggests that although it would be better to track hands using the hand-tracking framework in a constrained environment, a significantly high hand-tracking accuracy can still be achieved using the framework in an unconstrained environment.

6.6.4.1 Statistical analysis of the comparison between prototypes T, DT and DTL

The statistical analysis supports the conclusion that prototype DTL generally performs better than either prototype T and DT. The primary analysis used logistic regression with a compound symmetric model used to account for dependencies due to multiple observations on each individual performing the signs.

Initially, models with an interaction term between the prototype and the environment, in which the signed gesture was performed were considered. Interestingly, this term was significant on sign 17 only, as illustrated in Table 6.7. Therefore, simpler main effects models were used on all other signs. Table 6.7 illustrates that for sign 17 the effect of the prototype depends on the environment in which the sign is performed and/or the environment in which the sign is performed depends on the prototype. Furthermore, it shows that on sign 17 there is a difference between the prototypes used in a constrained environment, whereas no difference is found between the prototypes used in an unconstrained environment.

For all other signs the results of the analysis between prototypes for each sign are shown seperately in Table 6.8. Only results of the comparison between prototypes that were significantly different based on the adjusted p-value less than 0.05 are listed. This analysis show that on most signs, the results of prototype DTL are significantly different from both prototype T and DT, which confirms that prototype DTL is more suitable as a hand-tracking framework.

Table 6.7: Analysis on sign 17 representing an interaction between the prototype and the environment.

Environment	Prototype	_Environment	_Prototype	Estimate	Z-value	P-value	Adjustment	Adjusted p-value
Constrained	Prototype T	Constrained	Prototype DT	-0.14	-0.49	0.6232	Tukey-Kramer	0.9965
Constrained	Prototype T	Constrained	Prototype DTL	-1.2514	-3.14	0.0017	Tukey-Kramer	0.0207
Constrained	Prototype DT	Constrained	Prototype DTL	-1.1114	-4.25	<0.0001	Tukey-Kramer	0.0003
Unconstrained	Prototype T	Unconstrained	Prototype DT	-0.8127	-2.23	0.0256	Tukey-Kramer	0.2226
Unconstrained	Prototype T	Unconstrained	Prototype DTL	-0.4608	-1.39	0.1659	Tukey-Kramer	0.736
Unconstrained	Prototype DT	Unconstrained	Prototype DTL	0.3519	1.37	0.1701	Tukey-Kramer	0.744

TABLE 6.8: Analysis between prototypes in which the results were statistically significant with an adjusted p-value less than 0.05.

Observation	Sign	Prototype	_Prototype	Estimate	Standard error	Z-value	P-value	Adjustment	Adjusted p-value
1	1	Prototype T	Prototype DT	-0.6695	0.2132	-3.14	0.0017	Tukey-Kramer	0.0048
2	1	Prototype T	Prototype DTL	-0.975	0.2358	-4.13	<0.0001	Tukey-Kramer	0.0001
3	1	Prototype DT	Prototype DTL	-0.3055	0.117	-2.61	0.009	Tukey-Kramer	0.0245
4	2	Prototype T	Prototype DT	-0.7331	0.1458	-5.03	<0.0001	Tukey-Kramer	<0.0001
5	2	Prototype T	Prototype DTL	-1.0895	0.22	-4.95	<0.0001	Tukey-Kramer	<0.0001
11	4	Prototype T	Prototype DTL	-1.0946	0.2645	-4.14	<0.0001	Tukey-Kramer	0.0001
12	4	Prototype DT	Prototype DTL	-0.6475	0.2428	-2.67	0.0077	Tukey-Kramer	0.0209
13	5	Prototype T	Prototype DT	-0.8148	0.2484	-3.28	0.001	Tukey-Kramer	0.003
14	5	Prototype T	Prototype DTL	-0.8908	0.2434	-3.66	0.0003	Tukey-Kramer	0.0007
16	6	Prototype T	Prototype DT	-0.8418	0.3261	-2.58	0.0098	Tukey-Kramer	0.0266
17	6	Prototype T	Prototype DTL	-0.8471	0.3406	-2.49	0.0129	Tukey-Kramer	0.0344
19	7	Prototype T	Prototype DT	-0.5014	0.2122	-2.36	0.0181	Tukey-Kramer	0.0475
20	7	Prototype T	Prototype DTL	-0.9719	0.2158	-4.5	<0.0001	Tukey-Kramer	<0.0001
22	8	Prototype T	Prototype DT	-0.3955	0.148	-2.67	0.0075	Tukey-Kramer	0.0206
23	8	Prototype T	Prototype DTL	-0.7443	0.186	-4	<0.0001	Tukey-Kramer	0.0002
26	9	Prototype T	Prototype DTL	-1.1181	0.2609	-4.29	<0.0001	Tukey-Kramer	<0.0001
27	9	Prototype DT	Prototype DTL	-1.1181	0.1738	-6.43	<0.0001	Tukey-Kramer	<0.0001

28	10	Prototype T	Prototype DT	-0.7784	0.3154	-2.47	0.0136	Tukey-Kramer	0.0362
29	10	Prototype T	Prototype DTL	-0.8703	0.3332	-2.61	0.009	Tukey-Kramer	0.0244
34	12	Prototype T	Prototype DT	-0.5312	0.2083	-2.55	0.0108	Tukey-Kramer	0.029
35	12	Prototype T	Prototype DTL	-0.9595	0.1947	-4.93	<0.0001	Tukey-Kramer	<0.0001
36	12	Prototype DT	Prototype DTL	-0.4283	0.1151	-3.72	0.0002	Tukey-Kramer	0.0006
37	13	Prototype T	Prototype DT	-0.5148	0.1851	-2.78	0.0054	Tukey-Kramer	0.0149
38	13	Prototype T	Prototype DTL	-1.0127	0.1517	-6.68	<0.0001	Tukey-Kramer	<0.0001
39	13	Prototype DT	Prototype DTL	-0.4979	0.1527	-3.26	0.0011	Tukey-Kramer	0.0032
41	14	Prototype T	Prototype DTL	-1.098	0.3788	-2.9	0.0037	Tukey-Kramer	0.0105
43	15	Prototype T	Prototype DT	-0.4181	0.1733	-2.41	0.0158	Tukey-Kramer	0.0419
44	15	Prototype T	Prototype DTL	-0.7076	0.1879	-3.77	0.0002	Tukey-Kramer	0.0005
46	16	Prototype T	Prototype DT	-0.6562	0.2282	-2.88	0.004	Tukey-Kramer	0.0112
47	16	Prototype T	Prototype DTL	-1.025	0.239	-4.29	<0.0001	Tukey-Kramer	<0.0001
49	18	Prototype T	Prototype DT	-0.744	0.2292	-3.25	0.0012	Tukey-Kramer	0.0034
50	18	Prototype T	Prototype DTL	-1.1212	0.2325	-4.82	<0.0001	Tukey-Kramer	<0.0001
51	18	Prototype DT	Prototype DTL	-0.3773	0.1407	-2.68	0.0073	Tukey-Kramer	0.0201
52	19	Prototype T	Prototype DT	-0.7454	0.3037	-2.45	0.0141	Tukey-Kramer	0.0376
53	19	Prototype T	Prototype DTL	-1.2407	0.3303	-3.76	0.0002	Tukey-Kramer	0.0005
54	19	Prototype DT	Prototype DTL	-0.4953	0.2103	-2.36	0.0185	Tukey-Kramer	0.0486

56	20	Prototype T	Prototype DTL	-0.8765	0.2015	-4.35	<0.0001	Tukey-Kramer	<0.0001
57	20	Prototype DT	Prototype DTL	-0.551	0.1702	-3.24	0.0012	Tukey-Kramer	0.0035
59	21	Prototype T	Prototype DTL	-0.616	0.2566	-2.4	0.0164	Tukey-Kramer	0.0432
66	23	Prototype DT	Prototype DTL	-0.4803	0.1994	-2.41	0.016	Tukey-Kramer	0.0423
78	27	Prototype DT	Prototype DTL	-0.5595	0.1492	-3.75	0.0002	Tukey-Kramer	0.0005
81	28	Prototype DT	Prototype DTL	-0.504	0.1573	-3.2	0.0014	Tukey-Kramer	0.0039
84	29	Prototype DT	Prototype DTL	-0.6421	0.1407	-4.56	<0.0001	Tukey-Kramer	<0.0001
86	30	Prototype T	Prototype DTL	-0.5745	0.1804	-3.18	0.0014	Tukey-Kramer	0.0041
87	30	Prototype DT	Prototype DTL	-0.477	0.135	-3.53	0.0004	Tukey-Kramer	0.0012

When determining if the environment has a significant effect on the signs being performed, the results (as listed in Table 6.9) show that it does appear to, since the smallest p-value for all signs is 0.0211, which would not be considered significant using an adjustment for multiple testing. Although there were no significant differences between the environment and the sign, there were differences between the environment and the prototype. The results show that higher accuracies are achieved using prototype DTL in a constrained environment than an unconstrained environment.

TABLE 6.9: Evaluation to determine if the results are statistically significant between the environment and the sign being performed.

Observation	Sign	Environment	P-value	P-value	Method
1	1	Any	0	0.9723	Score
3	2	Any	0.1	0.7558	Score
5	3	Any	1.43	0.2325	Score
7	4	Any	0.74	0.3904	Score
9	5	Any	2.04	0.1536	Score
11	6	Any	0.83	0.363	Score
13	7	Any	4.98	0.0256	Score
15	8	Any	5.32	0.0211	Score
17	9	Any	0.79	0.3736	Score
19	10	Any	0.45	0.5045	Score
21	11	Any	0.33	0.5672	Score
23	12	Any	0.2	0.6578	Score
25	13	Any	2.43	0.1193	Score
27	14	Any	0	0.9879	Score
29	15	Any	0.41	0.5195	Score
31	16	Any	0.44	0.5049	Score
33	18	Any	1.07	0.301	Score
35	19	Any	0.05	0.8203	Score
37	20	Any	0.22	0.6406	Score
39	21	Any	0.39	0.5303	Score
41	22	Any	0.59	0.4416	Score
43	23	Any	2.03	0.1539	Score
45	24	Any	0.56	0.4561	Score
47	25	Any	2.61	0.1061	Score
49	26	Any	1.22	0.2701	Score
51	27	Any	0.26	0.611	Score
53	28	Any	0	0.9751	Score
55	29	Any	3.18	0.0747	Score
57	30	Any	0	0.9652	Score

In order to determine if there are any significant differences between the signs being performed, one approach is to focus on one prototype and one location. Since prototype DTL has the best overall performance and the environment does not seem to have a significant effect on the sign being performed, prototype DTL and the constrained environment were used. The results in terms of the proportion of frames tracked correctly across all signs are listed in decreasing order in Table 6.10. They show that sign 19 has the highest proportion of correctly tracked frames and does not differ significantly from those highlighted below it, whereas sign 29 has the lowest proportion of correctly tracked frames and does not differ significantly from those highlighted above it. Furthermore, the results show that the last 10 signs were performed using both hands with occlusion, which suggests occlusion between the hands does affect hand-tracking.

Table 6.10: Analysis to determine if there are any significant differences between the signs being performed on prototype DTL in a constrained environment.

Observation	Sign	Estimate	Standard error	Z-value	P-Value	Proportion
1	19	2.6362	0.4102	6.43	<0.0001	0.93316
2	5	2.5954	0.4801	5.41	<0.0001	0.93057
3	8	2.5429	0.4258	5.97	<0.0001	0.9271
4	6	2.5247	0.4434	5.69	<0.0001	0.92586
5	14	2.4354	0.319	7.63	<0.0001	0.91949
6	13	2.3947	0.4857	4.93	<0.0001	0.91642
7	7	2.3571	0.3797	6.21	<0.0001	0.9135
8	4	2.2789	0.3056	7.46	<0.0001	0.90711
9	1	2.2757	0.4431	5.14	<0.0001	0.90685
10	18	2.2609	0.4402	5.14	<0.0001	0.90558
11	10	2.145	0.4701	4.56	<0.0001	0.8952
12	15	2.0764	0.3575	5.81	<0.0001	0.88859
13	2	1.9754	0.3327	5.94	<0.0001	0.87819
14	11	1.9568	0.4183	4.68	<0.0001	0.87619
15	17	1.8916	0.3677	5.14	<0.0001	0.86893
16	3	1.6508	0.2929	5.64	<0.0001	0.83899
17	16	1.6006	0.3248	4.93	<0.0001	0.83211
18	12	1.5739	0.2803	5.62	<0.0001	0.82834
19	9	1.5495	0.2623	5.91	<0.0001	0.82485
20	24	1.2635	0.2573	4.91	<0.0001	0.77963
21	26	1.1657	0.2499	4.66	<0.0001	0.76236
22	22	1.1118	0.2367	4.7	<0.0001	0.75246
23	20	1.1041	0.1782	6.2	<0.0001	0.75104
24	21	1.0298	0.2118	4.86	<0.0001	0.73688
25	23	0.9887	0.2686	3.68	0.0002	0.72883
26	28	0.946	0.2266	4.18	<0.0001	0.72031
27	27	0.8111	0.2273	3.57	0.0004	0.69235
28	30	0.7655	0.1895	4.04	<0.0001	0.68255
29	25	0.7053	0.2257	3.13	0.0018	0.66937
30	29	0.6345	0.2144	2.96	0.0031	0.6535

6.7 Summary

In this chapter the design and setup of the SASL video dataset and the hand dataset were discussed. Along with the dataset, the criteria used to provide the ground-truth was explained. In addition, the performance across the three prototypes was evaluated and compared on the SASL video dataset, and the ELBP-RF hand-detection algorithm was evaluated on the hand dataset where a comparison was made between using LBP with random forests or SVMs.

To evaluate the algorithms, the evaluation protocol, which includes the metrics of accuracy and comparison, were discussed.

The comparison between the hand-detection algorithms analysed in Section 6.6.2.1 showed that it would be more suitable to employ the extended LBP with global histogram features when using either random forests or SVMs. It furthermore showed that random forests have a better performance compared to SVMs. Based on these results, the random forest approach was used in the detection phase in prototypes DT and DTL.

The analysis of the signs using prototype T showed encouraging results and it was thus the foundation of the hand-tracking framework. It also showed that tracking two hands while distinguishing between them increases the complexity of the algorithm and indicated that occlusion between the hands affects hand-tracking.

The analysis of prototype DT showed that hand-tracking accuracy increases when integrating the detection phase in the hand-tracking framework. On the other hand, it indicated that the detection phase may provide incorrect predictions when the hands cross one another, since images in which the hands crosses one another were not included in the hand-detection dataset. These incorrect predictions lead to lower accuracies, which represented tracking failures.

These tracking failures were addressed using prototype DTL, where improved hand-tracking accuracies were obtained by introducing a learning phase to the framework. Using prototype DTL demonstrated a high average accuracy of 82,08% in constrained environments and a high average accuracy of 79,83% in unconstrained environments, thus suggesting that prototype DTL can be used as a suitable framework for tracking hands in both constrained and unconstrained environments.

Chapter 7

Conclusion and Directions for Future Research

Advances in technology are rapidly progressing towards low-cost vision-based interfaces for human computer interaction that would eliminate the need to wear cumbersome equipment. One such advancement is a framework towards effective and accurate hand-tracking in any type of environment for the recognition of sign language from a single 2D view. The framework was developed around the same concept introduced by Kalal [81] that focused on a three-phase strategy, which in this study was detection-tracking-learning. Throughout the thesis these three phases are constantly referred to in order to provide a consistent and logical flow.

Existing literature relating to the approaches and aspects others have used on hand detection, hand-tracking and context-based object tracking was discussed in Chapter 2. It highlighted the relative strengths and weaknesses of each approach and emphasised the challenges that remain. The philosophical grounding that underpinned the research, the DSR methodology that guided it and the methods that were used to objectively evaluate the algorithms were described in Chapter 3. In Chapter 4 the different components that form part of the detection, tracking and learning phases were compared and their applicability to addressing the research problems was explained. In addition, it was concluded that an LBP variant combined with either random forests or SVMs is suitable for the detection phase. Chapter 4 described the unique skin segmentation algorithm and data association method used in the tracking phase; and in the learning phase it was suggested that SIFT is well-suited to extract and describe distinct features, since it generates the most robust feature points and uniquely describes them for feature matching. The systematic design and procedures used to link and integrate the algorithms to form three different hand-tracking frameworks were described in Chapter 5. The hand-tracking frameworks were represented by a progression of three prototypes. These prototypes were evaluated in Chapter 6. This involved describing the compilation of the SASL dataset and the evaluation metrics, a comparison of the hand-detection algorithms using the hand dataset, and a comparison between the three prototypes using the SASL dataset.

The experiments were aimed at evaluating the algorithms and answering the research questions posed in the thesis. Restating the hypothesis: *a single framework can be developed to track the hands of an individual independently from a single 2D camera in constrained and unconstrained environments.* The main research question was thus: How should a framework be developed to detect, learn and track the hands? To answer the research question, prototype DTL defines a framework that can be used to detect, track and learn the hands. It was shown that the ELBP-RF hand-detection algorithm is well-suited to detect hands in image sequences. To track the hands, it was shown that the pixel-level data-association algorithm performs very well in tracking the hands independently while handling occlusion. To learn features from the hand, it was shown that contextual features can be exploited to assist hand-tracking after tracking failure.

The experiments revealed that integrating all three phases in the framework, as was done in the final prototype, is suitable for tracking hands in both constrained and unconstrained environments, with a high average accuracy of 82,08% and 79,83%, respectively.

This chapter highlights the contributions of this study, states the limitations surrounding the research and recommends directions for future work. The chapter is concluded by reflecting on the study.

7.1 Contributions

In this thesis three fundamental contributions were made to the area of hand-tracking. These contributions originate from each of the phases that were collectively used to formulate a novel hand-tracking framework.

- Articulated hand detection using extended local binary patterns with random forests;
- Pixel-level data association for independent hand-tracking in unconstrained environments; and
- Exploiting contextual features towards assisting hand-tracking.

7.1.1 Articulated hand detection using extended local binary patterns with random forests

An integral part of detecting hands in video frames, especially in a sign language recognition application, is the ability to detect hands independently of the different hand configurations used in sign language. The detection of hands is further complicated by occlusion, complex environments and different illumination conditions.

The first contribution of this thesis is a unique hand-detection algorithm that addresses these challenges. The algorithm employs the extended LBPs with a random forest approach. When presented with a region of interest in an image, the region is filtered to only identify skin-coloured pixels using a new dynamic skin-colour model. The extended LBPs with a global histogram feature approach is applied to these skin-pixels and used to compile a feature vector that is later predicted using a trained random forest model to determine if the region of interest is a hand or not.

The evaluation of the algorithm on the hand dataset showed that by extracting features using the extended local binary patterns and the global histogram feature approach on skin pixels only, a higher hand-detection accuracy of 6.82% is achieved when compared to extracting these features from all the pixels in an image region. The evaluation also showed that the extended LBPs outperform both the original and uniform LBPs using global histogram features.

When compared to other hand-detection systems described in Section 4.1, the advantages of using this hand-detection algorithm is that it successfully detects hands independently of the hand posture formed and shows that a better detection accuracy is attained by only extracting features of the hand. In comparison to related hand-detection systems in terms of accuracy, the proposed hand-detection algorithm compares favourably and therefore should be considered state-of-the-art.

These noteworthy advantages make the algorithm well-suited for use in a hand-tracking framework for sign language recognition. Integrating the hand-detection algorithm in the hand-tracking framework showed that by validating that an object is a hand improves the hand-tracking system, thereby only tracking what is considered to be a hand.

7.1.2 Pixel-level data association for independent hand-tracking in unconstrained environments

Tracking hands from a 2D view is challenging when depth cannot be used to deal with occlusion. As an application towards SASL recognition, a hand-tracking framework is required that tracks the hands in both simple and complex environments. It is also required that a distinction can be made between the right and left hand.

The second contribution of this thesis is a data-association method that associates skin pixels according to whether they belong to either the right or left hand while discarding all other skin pixels not related to the hands. In this way it is able to distinguish between the right and left hand while tracking. The skin pixels are identified using a new dynamic skin model that selectively identifies an individual's skin colour. In addition, the method distinguishes between the hands even when they are stationary or moving in an occluded state. It was also shown that the dynamic skin model combined with the data-association method allows for the hands to be effectively tracked in both constrained and unconstrained environments.

The evaluation of the method on the SASL dataset showed that the method is a suitable for tracking both hands across a variation of signs and distinguishing between them while tracking. Distinguishing between them, however, increases the complexity in hand-tracking, which is the primary reason why some researchers [98, 120, 162] either do not distinguish between the hands or have difficulty in distinguishing between them. Furthermore, the method tracks the hands independently of the individual performing sign language, as the results show an equal variation between the different body types and skin-colour tones.

In comparison to related hand-tracking systems [36, 86, 97], the method successfully tracks the hand in the presence of background distractors and is not affected when the hands move in front of the face. Furthermore, it compares favourably in terms of tracking both hands against systems that employ particle-filter frameworks [32, 120, 144] and it is able to successfully distinguish between the right and left hand when compared to other related systems [98, 120, 162].

In addition, complementing the data-association method with the new dynamic skin model contributed to several improvements to the novel algorithm of Argyros and Lourakis [10]: it distinguishes between the right and left hand, it does not regard the hand as a different object when it recovers from tracking failure, and it is able to track both hands in both constrained and unconstrained environments.

7.1.3 Exploiting contextual features towards assisting hand-tracking

In sign language translation, if hand-tracking failure occurs, the hands need to be recovered and tracking should continue in order to provide continuous and accurate sign language translation. A few recent studies [26, 64, 163] have shown that context-based object tracking can be used to improve the accuracy of single object-tracking methods. To date, research on hand-tracking has not employed context-based strategies in the form of pixel-based features.

The third and perhaps most significant contribution of this thesis is a context-based algorithm that explores whether features on or around the hand can be used to assist hand-tracking and recover from tracking failure. These features are extracted from objects such as rings, wrist watches, bangles, bracelets, etc. that move with the hand and are assigned a higher confidence vote compared to features from objects that do not move with the hand, which have a lower confidence vote. These objects provide reliable features that can be used to recover hand-tracking failure and assist in distinguishing between hands. To provide reliable features, the algorithm employs SIFT detectors and descriptors, since they extract and uniquely describe the largest number of robust feature points that are scale, rotation and translation invariant. These features are stored in a database for each hand and are matched against newly extracted features. A voting strategy is then employed to recover the tracking of a 'lost' hand. Furthermore, it ensures that the right and left hands are the respective hands that are being tracked.

To determine how well the contextual information improves hand-tracking, the algorithm was integrated into the hand-tracking framework and evaluated using the SASL dataset. The evaluation showed that by using the context-based algorithm, a 7,07% increase in hand-tracking accuracy is achieved when compared to a system that excludes the algorithm from the hand-tracking framework. This indicates that the algorithm does recover from hand-tracking failure and assists in distinguishing between the right and left hand while tracking. The algorithm therefore demonstrates that features on or around the hand should be used to improve hand-tracking accuracies.

7.1.4 A framework for independent hand-tracking

The integration of the three main contributions of this thesis form the fundamental components of a novel hand-tracking framework proposed in this study. The framework has addressed some of the major challenges faced by hand-tracking and demonstrated that hands can be tracked in a sign language translation application with an average hand-tracking accuracy of 81% in both constrained and unconstrained environments.

The hand-tracking framework forms an integral component in the SASL translation system since it serves as a prerequisite for hand-shape recognition by finding the location of hands and indicates when the face is occluded by the hands, a scenario that requires information from the hand location, hand-shape recognition and facial expression recognition.

7.2 Limitations of the research study

The limitations and scope regarding this study defines a number of boundaries encompassed by the South Africa sign language translation research project.

Translating SASL gestures is specific to South Africa as gestures used to communicate using sign language differ in each country in the same way that languages differ from one country to another. The hand-tracking framework therefore focused on SASL gestures, but can be applied to the tracking of hands in general.

One of the aims of the SASL project is to provide translation using inexpensive hardware. The research therefore evaluated the hand-tracking frameworks with the focus on SASL videos captured from a 2D view using an inexpensive webcam. The analysis showed that by integrating the detection, tracking and learning phases, the hands can be tracked accurately and efficiently from a 2D view. It should be noted that while capturing the videos in unconstrained outdoor environments, windy conditions continuously caused the webcam to shake, thereby replicating the capturing of videos from a mobile phone camera. The results thus establishes the grounds that by using the proposed hand-tracking framework, the hands can be tracked when SASL videos are captured using mobile phone cameras. This was however not tested.

In addition, the videos were captured in uncontrolled light conditions in both constrained and unconstrained environments. This had the advantage of providing real-world scenarios of the type that would be encountered when using a mobile phone camera.

When communicating in SASL, it is considered disrespectful to not face the person with whom you are communicating. It was therefore required that the participant performing sign language faced the camera.

Furthermore, the participants were required to wear long-sleeved clothing, since it is more challenging than short-sleeved clothing, where the skin of the arms can be used to find and distinguish between the hands.

Sign language communication does not involve the entire body and requires only the upper body to convey the message, thus video capturing was only restricted to the upper body. This required the participants to be closer to the camera compared to video capturing the entire body, which contributed to richer face and hand features, which are important in SASL. The hand-tracking framework is not dependent on the participants' distance from the camera; however, the distance may influence the features extracted in the learning phase. Determining whether these features are influenced by distance in the hand-tracking framework is an area for possible future research.

7.3 Future work

Although substantial contributions to hand tracking were formed in this thesis, and the research objectives were achieved, directions for further improvement can still be made. In this section, several directions to further improve and extend the approaches presented in this thesis will be discussed. This includes suggestions that could increase the overall performance of the hand-tracking framework.

7.3.1 Arm-based hand-tracking

As an extension to the hand-tracking framework, a method could be integrated that would allow for the hands to be tracked when short-sleeved clothing is worn. One way to address this problem is to detect skin pixels from both the hands and arms, and create a binary image of skin and non-skin pixels. When the hands move from one location to another, the arms move with them at all times, and therefore the hands and arms can be isolated using a background subtraction technique. Combining the foreground image representing the moving objects with the skin-pixel binary image would result in only skin-coloured foreground objects represented by silhouettes. Segmenting elongated silhouettes would be a means to identify the hands and arms. Finding the skeleton of these silhouettes and using the skeletons to identify the end at which the hand is situated, is a way to segment the hands from the arms. Since the arms are bound by physical constraints, they have a limited area in which they can move. This fact can then be used to track and distinguish between hands.

7.3.2 Enhancing skin-colour segmentation

A fundamental function of the tracking algorithm is to identify the hands based on the skin colour of the individual, since it provides shape and scale invariance. In the current research, the individual's skin colour was identified based on a selected area around the nose. This requires the individual to face forward towards the camera, which is a valid assumption, since facial expressions are the non-manual features that are used to distinguish between signs.

If the individual turns his/her face or looks away from the camera, a Gaussian model, for example, should be used. Therefore, an extension to the skin-segmentation algorithm in the hand-tracking framework would be to continue segmenting skin-coloured regions even when an individual happens to not face towards the camera. A possible approach would be to train an online Gaussian model using the skin-colour distribution selected from the face at times when the individual is facing towards the camera and employing the model when either the individual's face cannot be detected or when the individual does not face the camera. An alternative approach would be to apply an online trained model in every frame and logically 'AND' it with the output of the skin-segmentation algorithm of the hand-tracking framework, which would possibly provide a more accurate and enhanced skin-colour segmentation.

Another possible approach would be to employ a semantic-segmentation algorithm similar to the work of Liu et al. [95] and use the semantic labels to isolate the individual in the scene. The individual's skin colour should then be determined or pixels should be grouped together based on their similarity, so more non-hand objects or areas can be excluded from the scene, which would lead to a more accurate and efficient tracking process.

7.3.3 Improved hand detection

The ELBP-RF hand-detection algorithm proposed in this study requires an image region to determine if this region is a hand. Given an image region, the extended LBPs are applied and used to populate a histogram using a global feature approach. The histogram features are then used to train a random forest model, where the model is used to predict the image region. As an extension to this algorithm and an improvement to the hand-tracking framework, the algorithm should include a 'sliding window' mask that scans the entire image from the top-left to the bottom-right corner and assigns probability values to each region in the image using the trained random forest model. These probability values will define each image region's likelihood of containing a hand. Based on these likelihoods, regions with lower probability values can be excluded, which would decrease the probability of hand-tracking failure. In SASL, hand shapes are an additional feature, together with hand-tracking and facial expressions, that are used to describe sign language gestures. Therefore, to further improve hand detection, the hand shape can be used. The hand shape can be determined using several 3D model approaches, where templates are extracted and matched, or appearance-based methods that, for example, predict the hand shape based on extracted low-level features and a trained model. Knowing the hand shape helps to further eliminate areas in an image that do not contain a hand.

7.3.4 Optimised hand-tracking framework

The current hand-tracking framework was aimed at improving the hand accuracy using several algorithms; however, these algorithms were not optimised to achieve real-time performance. A performance evaluation was therefore not done; however, several areas can be optimised to increase hand-tracking performance. To improve the face-detection algorithm, rather than scanning the entire image for a face, only the top half of the image should be scanned. Another improvement would be to determine the foreground objects and skin-coloured objects in parallel, followed by combining these areas using a logical ‘AND’ function. To identify the skin-coloured clusters in an image, the connected-components labelling algorithm was used. By applying the optimised connected-components labelling algorithm of Wu et al. [165], the time required to scan an image can be reduced by half and the total execution time can be reduced by a factor of five or more.

Furthermore, the time required to compare features in and around the hand with features from the respective databases can be reduced by excluding features that are identical in both the right- and left-hand databases. These features are often extracted from the region where the hands overlap and are then stored in both databases. Only storing and comparing the features that are unique to each hand will not only reduce the computational time, but might also improve the accuracy of the hand-tracking framework.

In order to further improve the performance of the hand-tracking framework, the proposed algorithms can be ported to run on a GPU, thereby exploiting the GPU’s parallel processing capabilities. These capabilities are attributed to the GPU’s architecture, which contains hundreds of cores that allows it to run millions of threads concurrently.

7.4 Reflecting on the study

In this section, the current interests of the study, the significance of the study and its relation to previous studies are considered.

7.4.1 Current interest of the study

With the current interest in vision-based gesture interfaces, this research is timely. It addressed the challenging process of hand tracking and recommended a framework that aims to track the hands from a single 2D-view camera. This framework can be easily adapted to several application areas such as virtual controllers, alternative computer interfaces and immersive gaming technology. The fact that the framework applies to single 2D-views, even 2D-views from mobile phone cameras, makes it more appealing than tracking hands using multiple cameras or specialised 3D-view cameras.

The study will appeal to researchers interested in hand-tracking, body-part tracking, body-pose estimation and multiple-object tracking, as the research can be used as a source of reference in each of these areas.

7.4.2 Significance of the study

This thesis provides several significant and value-added contributions to the knowledge base. It provides novel algorithms that have been shown to significantly improve hand-tracking accuracy. Although these algorithms have been applied to hand-tracking, they can also be applied to the general case of tracking multiple objects with a similar appearance. These algorithms have been combined to form a novel hand-tracking framework that effectively tracks the hands while distinguishing between the right and left hand. The framework is an important contribution to the translation of South African sign language and will therefore change the way in which the hearing-enabled and Deaf communicate with each other.

7.4.3 Relation to previous studies

The research highlighted the challenges and shortcomings that previous studies faced in terms of hand-tracking systems. It presented arguments explaining these challenges and highlighted the areas where the algorithms in these studies may have been flawed; for example, some studies [32, 120, 144] believe that a particle-filters approach would address the problem of distinguishing between each hand after occlusion has occurred; however, this approach assumes the hands will follow the same direction and would be feasible only if the likelihood function of the particle filters can distinguish between two similar objects such as hands. Alternatively, Argyros and Lourakis [10] proposed a more effective approach that associates pixel-level information from multiple objects and does not assume that the hands will always follow the same direction. This approach was therefore adopted in this research and forms the underlying structure of the hand-tracking framework. The research confirmed that associating pixel-level information should be used for hand-tracking systems. In addition, the research showed that detecting whether an object is a hand is a better approach compared to blindly tracking an object without knowing whether it is a hand or not. Moreover, the research demonstrated that extracting contextual information from the hands and objects on the hand significantly improves hand-tracking accuracy. This approach has never been explored in any hand-tracking system. It is therefore recommended that future research should adopt this approach not only for hand-tracking, but for other objects that may have a similar appearance.

Appendix A

State-of-the-art feature detectors and descriptors

A.1 Feature detectors

Several feature detectors have been proposed with the aim of improving accuracy and efficiency [15]. Feature detectors are used to find a feature point in images that can later be extracted by a feature descriptor. In the following sections, state-of-the-art feature detectors will be described.

A.1.0.1 Harris

The Harris feature detector is a classical approach commonly used to detect corners in an image [69]. It considers the change in average directional intensity in a small window around a candidate pixel by computing the auto-correlation matrix, A, which is represented as [88]:

$$A = \begin{bmatrix} u & v \end{bmatrix} \begin{bmatrix} \sum[\frac{\partial I}{\partial x}]^2 & \sum \frac{\partial I}{\partial x}\frac{\partial I}{\partial y} \\ \sum \frac{\partial I}{\partial x}\frac{\partial I}{\partial y} & \sum[\frac{\partial I}{\partial y}]^2 \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} \tag{A.1}$$

where (u, v) is the displacement vector, $\frac{\partial I}{\partial x}$ and $\frac{\partial I}{\partial y}$ are the first derivatives of image $I(x, y)$. From this matrix, two eigenvalues are determined that represents the maximum change in average intensity, and the change in average intensity for the orthogonal direction. The eigenvalues are then compared to a given threshold with the following outcome: if both eigenvalues are low then a relatively homogeneous area is defined, if one eigenvalue is low and the other is high then the point is an edge, and if both eigenvalues are high then the point is a corner.

It then follows that the intensity strength can be computed with the following score [88]:

$$S = det(A) - \lambda \cdot trace(A)^2 \tag{A.2}$$

where λ is selected to be in the range of 0.05 and 0.5 [88]. To determine the final detection result, a local non-maxima suppression step is performed. A point is thus considered to be a corner, only if it is a local maximum and has an intensity strength higher than a specified threshold.

A.1.0.2 Good-features-to-track

The Good-Features-To-Track method was proposed by Shi and Tomasi [140] as an extension of the Harris feature detector by making its corners more uniformly distributed across an image. This is achieved by imposing a minimum distance between two feature points. Starting from a feature point with the strongest Harris score, S, feature points are accepted only if they are situated within a given distance from the points that have already been accepted. The score is therefore determined as follows [138]:

$$S = min(\lambda_1, \lambda_2) \tag{A.3}$$

where λ_1 and λ_2 represent the eigenvalues.

A.1.0.3 Feature from accelerated segment test

The Feature from Accelerated Segment Test (FAST) detector algorithm [133] is another extension of the Harris feature detector and designed to be a quick corner detector. The algorithm operates by examining the intensity values of pixels on a circle of radius,

r, around a candidate point, p. Each pixel on the circle is considered either bright or dark if its intensity value is brighter or darker by at least, t, when compared to the intensity value of p and where t is an arbitrary threshold. A segment test then classifies the candidate pixel as a feature point if a contiguous arc of at least, n, bright or dark pixels is found on the circle perimeter. Similar to the Harris detector, the non-maxima suppression is only applied to those feature points that have successfully passed the segment test, where the corner strength is determined by the sum of the absolute difference between the pixels identified on the arc and the candidate pixel.

As an extension to FAST, Rublee et al. [134] proposed the Oriented FAST and Rotated BRIEF (ORB) feature detector. The aim of the detector is to estimate the feature point orientation where the FAST detector does not. This is achieved by efficiently computing the orientations based on the intensity centroid moments.

Another extension to FAST was proposed by Leutenegger et al. [92], which they referred to as the Binary Robust Invariant Scalable Keypoints (BRISK) detector. It improves the score computation of the FAST detector by using a binary decision tree instead of the original classifier. Furthermore, it improves the scale parameter of FAST by computing the score over several scales in a scale-space representation.

A.1.0.4 Speeded up robust feature

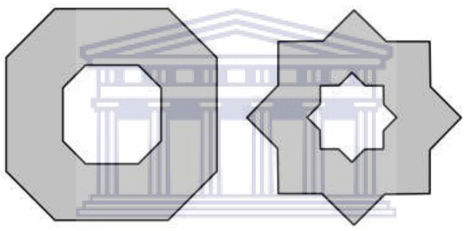
Inspired by the SIFT detector, Bay et al. [17] proposed the Speeded Up Robust Feature (SURF) detector. Their main motivation was to overcome the computational complexity and slow execution speed of SIFT. In contrast to SIFT that uses the Laplacian filter response, SURF operates by searching for the local maxima of the Hessian determinant in the scale space. For each pixel in the image, the Hessian matrix is computed as follows [88]:

$$H(x,y) = \begin{bmatrix} \frac{\partial^2 I}{\partial x^2} & \frac{\partial^2 I}{\partial x \partial y} \\ \frac{\partial^2 I}{\partial x \partial y} & \frac{\partial^2 I}{\partial y^2} \end{bmatrix} \tag{A.4}$$

The matrix is used to measure the local curvature of a function and the determinant measures the strength of this curvature. The aim is to find image points with a high local curvature. In order to improve the efficiency of the detector, it uses integral images and calculates the Hessian determinants by using approximated Gaussian kernels of different scales. Using SIFT, scale estimation is obtained by decreasing the size of the image after smoothing; however, using SURF, it is obtained by increasing the size of the kernels. Once the local maxima is identified, the location of the feature points are obtained through interpolation in both image and scale space and to which a scale value is associated to each feature point [88].

A.1.0.5 STAR

The STAR feature detector was proposed by Agrawal et al. [5] as a derivative of the Center Surround Extrema (CenSurE) detector. It approximates the Laplacian using a bi-level centre-surrounded filter consisting of two rotated squares as shown in Figure A.1.



(a) CenSurE (b) STAR

Figure A.1: The CenSurE and STAR detectors.

The star-shaped filter allows the detector to preserve rotational invariance. Moreover, the use of integral images allows for efficient detection of feature points in scale-space, where the scale-space is constructed without interpolation, by applying filters of different sizes.

A.1.0.6 Maximally stable extremal regions

The Maximally Stable Extremal Regions (MSER) detector was proposed by Matas et al. [105] to detect regions in an image. This is achieved by using local intensity extrema obtained by applying a water-based segmentation iteratively. The regions are detected as connected components in an image in which a threshold is applied from the maximum to the minimum image intensity value and the stability of the connected components are evaluated. Thus, all pixels inside a region will either have a higher or lower intensity value than all other pixels on its outer boundary.

Each of the feature detectors have their relative strengths and weaknesses. In Section 4.3.4 a comparison is made to determine one that is suitable.

A.2 Feature descriptors

After feature points have been located, features can be described by means of feature vectors that is referred to as feature descriptors. To couple the detectors, many descriptors have been proposed. In the following sections, state-of-the-art feature descriptors will be described.

A.2.0.7 Speeded up robust feature descriptor

First, the orientation is assigned to the feature point. Similar to the SIFT descriptor, local image gradients are computed within a 4x4 grid rotated according to the orientation around the feature point. Each grid block is further divided into sub-regions. At each sub-grid block, the gradient is computed and accumulated by the angle into histogram bins, whose counts are increased by the magnitude of the gradient that is weighted by a Gaussian. These gradients are calculated based on first-order Haar wavelet responses, in the horizontal and vertical directions, that are computed efficiently on different scales using integral images. These histogram of gradients result in a feature vector consisting of 64 elements. Furthermore, by normalising the vector, illumination invariance is achieved [16].

A.2.0.8 Binary robust independent elementary features descriptor

The Binary Robust Independent Elementary Features (BRIEF) was proposed by Calonder et al. [24], to describe features using binary strings. This allows the hamming distance to be used for matching, which is faster than using the euclidean distance. Before extracting features, the image is smoothed with a sample averaging filter, due to the descriptors' sensitivity to noise. The value of each bit in the binary string is determined by comparing the intensity value between two points inside an image region centred on a given feature point. If the intensity value of the first point is greater than the second point then the bit is set to one, otherwise it is set to zero. Calonder et al.

[24] have shown that the best performance is achieved when the points are chosen with the Gaussian distribution centred on the feature point.

A.2.0.9 Oriented FAST and rotated BRIEF descriptor

The ORB descriptor extends the BRIEF descriptor by making two improvements [134]. First, it applies orientation information from the ORB detector to the descriptor that enables rotation invariance. This is achieved by using the detected orientation angle to rotate the coordinates of the point pairs in the binary tests around a given feature point. Second, instead of using random sampling to select the point pairs as in the BRIEF descriptor, a sampling scheme that employs machine learning for de-correlating BRIEF features under rotational invariance is used. This improves the matching accuracy when using the nearest neighbour search [137].

A.2.0.10 Binary robust scalable keypoints descriptor

In contrast to the random or learned patterns of BRIEF and ORB descriptors, the Binary Robust Scalable Keypoints (BRISK) descriptors employ a symmetrical pattern [92]. The sample points are positioned around a feature point, in concentric circles, where each sample point represents a Gaussian blur of its surrounding pixels. The orientation of the feature point is determined by comparing the sample points on opposite sides of the descriptor pattern. The pairs of points are separated into two subsets, long-distance and short-distance pairs. The vector displacements between the compared long-distance pairs are stored and weighted by the relative gradient. The weighted vectors are then averaged to obtain the dominant gradient direction of the image patch. Finally, the short-distance pairs are used to build binary descriptors after rotating the sampling pattern.

A.2.0.11 Fast retina keypoint descriptor

The Fast Retina Keypoint (FREAK) descriptor proposed by Alahi et al. [8] is based on the basic concept of BRIEF. In a similar manner to BRIEF, it obtains the feature point orientation by taking the sum of the estimated gradients over selected point pairs, with the additional task of using a cascade approach. In contrast to the BRIEF, ORB and

BRISK descriptors, the FREAK descriptor uses a specific sampling point pattern that is biologically inspired by the retinal pattern of the eye. This sampling pattern uses a *coarse-to-fine* approach where the averaging areas are overlapped and becomes more concentrated near the feature point, while becoming less concentrated when moving further away from it. They suggest that the approach leads to a more accurate feature point descriptor. Furthermore, they suggest the computation time is decreased by first comparing the point pairs with the most distinctive characteristics in the neighbourhood.

A.3 Image matching

When matching distinctive characteristics between images, descriptors are very useful. To match these descriptors, as an alternative to the FLANN-based matcher, the Brute-force matcher can be used. In the following section, this feature matcher will be described.

A.3.0.12 Brute-force matcher

When searching for a pattern in an image, the set of features belonging to the pattern can be referred to as the *query* set and the set of features belonging to the image can be referred to as the *test* set. The more correspondences between the pattern and the image, the higher the probability will be that the pattern can be found in the image.

The brute-force algorithm is an exhaustive search method that searches for a corresponding feature vector in set A by comparing it with each one in set B [88]. The result is a list of correspondences between the two sets.

Appendix B

Consent form

I, ____________________________, understand that my participation in this sign language recognition research is solely for the collection of data, to improve the quality of software in general and I agree to participate. I understand that images in which I can be seen will be used in the documentation of this research, to illustrate the sign language performed and additional information, and that my name will not be attached to the images. I am also aware that I am allowed to withdraw from participating in this research, at any time. I furthermore give permission that the video data can be shared in a repository and made available to other researchers on their request for the purpose of academic research only.

For further information, please do not hesitate to contact:

Imran Achmed
Dept of Computer Science
University of the Western Cape
Bellville 7535
Email: 2507311@myuwc.ac.za
Cell: 073205 5114

	Video capturer		**Participant**
Name:	____________	Name:	____________
Signature:	____________	Signature:	____________
Date:	____________	Date:	____________

Appendix C

Dataset compilation

The signs listed in the tables below contain the full vocabulary of the “Fulton School for the Deaf SASL Dictionary” [73]. These groups were identified based on the similarity between signs. The group of signs were then assigned to one of three classes, namely, gestures containing a single hand, gestures containing both hands without occlusion and gestures containing both hands with occlusion.

TABLE C.1: Group 1: These sign gestures contain single hand movements only.

Gestures Containing One Hand Only	
Group 1	Yes, bye-bye, goose, spider, beautiful, aeroplane, cut (using scissors), animal, dolphin, salt, another, other, to flush, scissors, kill, Joseph, spider, firefly, bell, borril, mirror, wand, what for, handbag, aeroplane, daughter, wife, boy, child, juice, fried egg, plum, naartjie, gem squash, sponge, sore (pain), money, pen, Christmas, our, he, she, they, this microwave, bank
Group 2	Laugh, telephone, man, king, happy, sad, laugh, dirty, full, cruel, kind, rich, truth, black, Saturday, birthday, pill/tablets, thermometer, pencil, lunch, bull, goat, hen, duck, oil, park, sweet, quiet, red, pink, doesn’t matter, feather, wolf, fox, chicken, oil, frog, ant, nose-stud, sour, sweet, ugly, lie, news, ice-cream, ice-cream cone, peanut butter, fruit, apple, orange, cherry, chemist, flower, sneeze, baptize
Group 3	Voice, who, vinegar, vegetable, tea spoon, prince, candle, dentist, dog, bird, neck, while, coke, cough, camel

TABLE C.1: Group 1: These sign gestures contain single hand movements only.

Group 4	Mane, rooster, lion, shark, cap/peak, shower, hat, giraffe, whale, fly, Hindu, Muslim, wizard
Group 5	Go, Sunday, bakkie, lorry, tennis, fish, tadpole, work, snake, pan, which, what, fish, fry, plug-basin, young, hand spade, cricket, staff, button, money, chest, powder, hungry, stomach, belly button, sugar, flour, to lock, good, stomach-ache, me, mine, you, your, myself
Group 6	Sorry, name, talk, oral, drink, come, tomorrow, sweet (to eat), sweet (cute), when, beef, God, parents, grandparents, grandmother, grandfather, niece, granddaughter, bubble-gum, chewing gum, strawberry, lemon, breakfast, lunch, supper, carrot, facecloth, make-up, lipstick, uncle, aunt, all, key, kettle, pour, down
Group 7	Why, we, feather duster, Friday, hospital, clinic, doctor, Christian, jewellery, necklace, mother, princess, seat-belt, fast
Group 8	Plug-wall plug, short
Group 9	Go (using a flag), water, a drill, vomit
Group 10	Hearing person, deaf person, hearing aid, sand, to show, shell, ear, nephew, naughty, morning
Group 11	Hair, head, television, music, radio (2), interesting, easy, boring, eye, eyebrow, eyelashes, father, granny, grandson, onion, evening
Group 12	Girl, first, yellow, orange, Monday, Tuesday, Wednesday, Thursday, stable
Group 13	Please, thank you, teeth, tomato sauce, apple juice, tomato, right, spit out, powder, hair brush, comb, coat hanger, sun, sunny, lightning
Group 14	Hello, hair dryer, temperature, headache
Group 15	Elephant
Group 16	That
Group 17	Bucket
Group 18	Fish hook

Table C.2: Group 2: These sign gestures contain the movement of both hands without any interaction or occlusion.

Gestures Containing Two Hands and No Occlusion	
Group 1	Wake up, queen, devil, cry, visit, glasses, snorkel, owl, cry, awake
Group 2	Give, crab, clothes, dress, pull-over, oven, pot, stool, taps, sheets, blanket, duvet, piano, vase (1), different, today. Grass, bush, garage, soccer, trophy, swim, present (gift), wrapping paper, biscuit, happy (as in birthday song), material, fine/feeling well, coffin, dentist chair, box, sticky tape, bless, puppy, lizard, feet, toys, bake, boil, cold, tackles, vest, gumboots, untidy (2), bitter, wheel, train, act, dance, to pass away, rinse
Group 3	Kiss, balloon, cat, kitten, loud
Group 4	Tracksuit-top, tie, milk, camping
Group 5	Hat (with brim), loud, rain, earrings, bus, helmet, rabbit
Group 6	Run, play, grated cheese, grape juice, egg, popcorn, bathroom
Group 7	Any, plate, dustpan, hand broom, kitchen, bubbles, curtains, mattress, praise
Group 8	To sign, soft, die, cold water, now, water, splash, foam, disappear, tent, mayor
Group 9	Purse, nipples, breasts, children, jelly, grapes, watermelon, fire, couch, lounge, radio (1), people, narrow, many, difficult, hard, soft, transport, bicycle, scooter, motorbike, fireman, fingers, light, number, count, love, assault, lord, “kraal”
Group 10	Scrambled eggs, boiled egg, macaroni, avocado, bowl, colour, small, short
Group 11	Dress, skirt, bath mat, bath, arm chair, beach, sand, costume (swimming), bikini, born, funeral, fat, tractor, swimming, bench, a plank
Group 12	Hail, snow
Group 13	Opposites, thin, wet, dry, waves, foam, life, wind, windy, earthquake, road
Group 14	Change, heavy, tidy, untidy, car, car lights, fork watering can, bucket, spade, basket, station
Group 15	Fishing rod, reel, paddle-ski, concert
Group 16	Big, wide, bright, fairy, sea gull, angel
Group 17	Finish (with determination), milkshake, week-end
Group 18	Boxing, wrestling, deck-chair, holly
Group 19	Cow, calf, donkey, rabbit, bear, snail, teddy bear, pillow, loud, weet-bix, rice-crispies, Worcester sauce, hut

Table C.2: Group 2: These sign gestures contain the movement of both hands without any interaction or occlusion.

Group 20	Pot lid, jug, broom
Group 21	Themes, weather, clouds, rain, misty, snowing, stars, sky, morning, light, thunder bus, flashing lights, weight lifting, gym, party, board, holiday, religious celebrations
Group 22	Pope, ambulance, reindeer, decorations, christmas tree
Group 23	Toast, grill, toilet paper, above, below, under, rough, full, shallow, prize, bandage, Easter, on, smooth, brown
Group 24	Hamburger, cabbage, shampoo, card (greeting), mealie, dinner, sour, restaurant, eat, build
Group 25	Photograph, hairdresser, camera, magic , paint brush, video-camera
Group 26	Razor, light the fire, injection
Group 27	Put on clothes, take off clothes, cook, stop, colour, finish, pyjamas, apron, pegs, piano, newspaper, bridge, double garage, gala, castle
Group 28	What are you doing?
Group 29	Went, golf, a hammer
Group 30	Crocodile, hippo, chisel, ladder
Group 31	Ostrich, octopus
Group 32	Seal, penguin, where, girl panties, boys underpants, shorts, tracksuit pants, pockets, train, motorbike, toy car, scooter, lawnmower
Group 33	Dressing gown, dust (2)
Group 34	Shoulders, dark, light, night, breeze, nurse, sister, stage, Mary
Group 35	Skateboard, blunt, sharp, spade, dig, skate-barding, surf
Group 36	Hot chips, hot plates, nurse/sister
Group 37	Basket
Group 38	Electricity, dust (1)
Group 39	Towel
Group 40	Prime minister
Group 41	Minister/priest
Group 42	Traffic policeman, teacher, Koki-pen
Group 43	Over

TABLE C.2: Group 2: These sign gestures contain the movement of both hands without any interaction or occlusion.

Group 44	Old
Group 45	Storm
Group 46	Fish pond, sewing, sewing machine, operation
Group 47	Sea foam
Group 48	Scuba-dive
Group 49	Sick
Group 50	Stethoscope
Group 51	Fire-station
Group 52	Story, paper, braai
Group 53	Baptize (full immersion in a pool or river)
Group 54	Sleigh

TABLE C.3: Group 3: These sign gestures contain the movement of both hands that includes the interaction and occlusion between hands.

Gestures Containing Two Hands With Occlusion	
Group 1	Stand, sit, lie down, hop, tidy up, to colour, finish, paint, socks, long-sleeved shirt, stamp, principle, glue, jump, fall, toothpaste, clean, purple, a screw, bee, grasshopper, mosquito, credit card, jam, pepper, chocolate, pear, tea-time, this (very specific), soap, cream, chair, silver, seed, petrol, tea, butternut, slow, in, inside, blue, bottle, Christmas cake, x-ray
Group 2	No, wrong, engine, taxi, dear (as in birthday song), love, change, cotton, tape measure, porcupine, squirrel, bat, butterfly, beetle, dragonfly, short-sleeved shirt
Group 3	Web, few, dark, jersey
Group 4	Porridge, cereal, letter, soup, yoghurt, cup, saucer, fun, to sew, medicine, crayons
Group 5	Sheep, bracelet, arm, doll, baby, coke (1), lamb, suntan cream, computer, bones

Table C.3: Group 3: These sign gestures contain the movement of both hands that includes the interaction and occlusion between hands.

Group 6	A rake, hosepipe, cake, fire-engine, flowers (as a gift)
Group 7	Painting, river, paint-2 (to paint), Jesus
Group 8	Policeman, jail (2)
Group 9	Cheese, sandwich, baby Jesus
Group 10	Begin, pony, umbrella, coffee, stove, wash, washing machine, basin, cupboard, carpet, rug, a year, police, games, match, sangoma, cracker, rat, family, turtle
Group 11	Hand, thumb, toe, juice, hot dog, nail brush, afternoon, a month, calendar, van, cousin, husband, knit
Group 12	House, home, vase (2), team, wool, window, room, lift, dining room, garden, envelope, ruler
Group 13	Wallet, rice, poached egg, omelette, peach, dishwasher, knife, fork, a shelf, few, sports, woodwork, wood, jacket, cross, bun, bicycle, ship, pie, chips, floor, fridge, deep freeze, toilet, to poo, blue, green, gate, race, caravan
Group 14	Sleep, bed, asleep
Group 15	Pray, wash, shoes, boat, lettuce, book, church, jail (1), same, rocket, sport, needle, plaster, school, school bag, temple, mosque, valentine
Group 16	Cut (using a knife), potato, pea, bean, tow truck, boat, fishing
Group 17	Farm
Group 18	Wild, tortoise, peacock, how many, how, brother, marmite, puffed wheat, sausage, ceiling, brickets/coal, wood, shelf, strong, long, short, secateurs, ship
Group 19	Rhino, warthog, handkerchief, ball
Group 20	Mole, mouse
Group 21	Stockings, leg, knee
Group 22	Glove
Group 23	Slippers, sandals, slops, ring, bread, butter, meat
Group 24	Yacht, flag
Group 25	Banana
Group 26	Pineapple, cauliflower
Group 27	Broccoli, pumpkin, lettuce

Table C.3: Group 3: These sign gestures contain the movement of both hands that includes the interaction and occlusion between hands.

Group 28	Door, open door, close door, friend
Group 29	Corner, wall
Group 30	Escalator, branch, sea-weed, card (birthday)
Group 31	Chimney, light, standing lamp, smoke, switch on light, switch off, tall, up, gold, yesterday, basketball, go (using a gun)
Group32	Lamp, week, helicopter
Group 33	Rubbish bin, table, a week
Group 34	Day, post office, palm tree
Group 35	Empty
Group 36	Open, close, volcano
Group 37	Deep, leaf
Group 38	Poor, tree
Group 39	Next week
Group 40	Moon, rainbow
Group 41	Rugby, hockey, blood, bleeding
Group 42	Table tennis
Group 43	To score a goal
Group 44	Goal posts
Group 45	Win, kick
Group 46	Sea, chain
Group 47	Bait, to saw, a nail
Group 48	Save, a saw, a spanner
Group 49	Bible
Group 50	Bag
Group 51	Manger/crib
Group 52	Lamb chop
Group 53	Prize

Table C.4: The selected groups of signs used to evaluate the prototypes.

Gestures containing a single hand		Gestures containing both hands without occlusion		Gestures containing both hands with occlusion	
Group 1	Sign 1	Group 2	Sign 11	Group 1	Sign 21
Group 2	Sign 2	Group 3	Sign 12	Group 2	Sign 22
Group 5	Sign 3	Group 5	Sign 13	Group 10	Sign 23
Group 7	Sign 4	Group 7	Sign 14	Group 12	Sign 24
Group 9	Sign 5	Group 8	Sign 15	Group 16	Sign 25
Group 12	Sign 6	Group 20	Sign 16	Group 18	Sign 26
Group 13	Sign 7	Group 23	Sign 17	Group 21	Sign 27
Group 14	Sign 8	Group 26	Sign 18	Group 25	Sign 28
Group 16	Sign 9	Group 28	Sign 19	Group 28	Sign 29
Group 17	Sign 10	Group 49	Sign 20	Group 53	Sign 30

Appendix D

Additional results

TABLE D.1: Evaluation results using a threshold of 10, 15 and 20 pixels from the center of the hand.

	10	15	20	Standard deviation
Person A	89.09%	98.18%	100.00%	4.77%
Person B	64.38%	76.71%	79.45%	6.55%
Person C	96.43%	99.11%	100.00%	1.52%
Person D	38.89%	40.28%	38.89%	0.65%
Person E	37.80%	43.90%	47.56%	4.02%
Person F	97.67%	98.84%	100.00%	0.95%
Person G	81.25%	95.83%	100.00%	8.04%
Person H	100.00%	100.00%	100.00%	0%
Person J	98.78%	98.78%	100.00%	0.57%
Person K	92.42%	98.48%	98.48%	2.86%
Average	**79.67%**	**85.01%**	**86.44%**	**2.91%**

D.1 Tracking

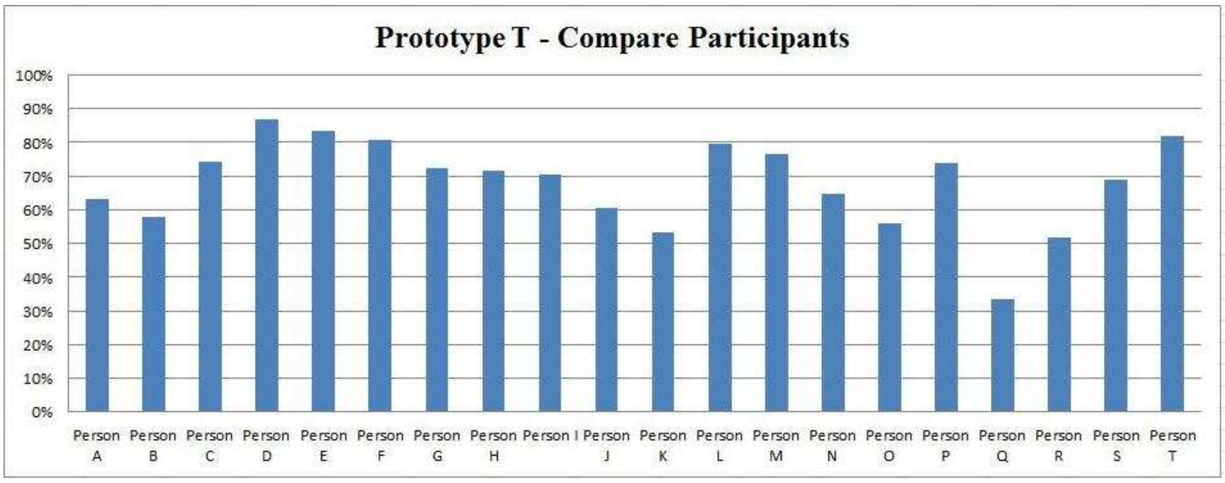


FIGURE D.1: The average hand-tracking success rates of prototype T for each participant in both environments.

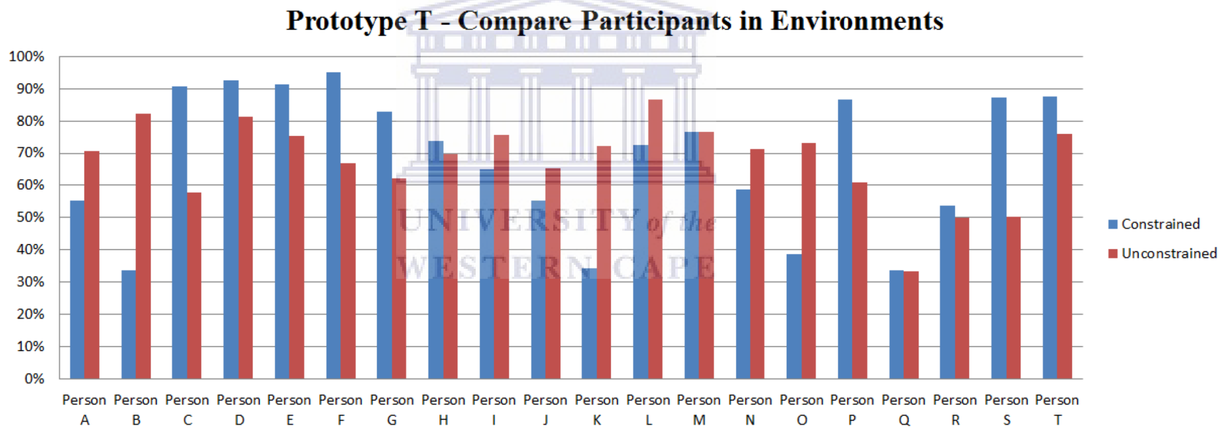


FIGURE D.2: A comparison of the average hand-tracking success rates of prototype T for each participant in constrained and unconstrained environments.

Table D.2: Compare sign gestures using prototype T in constrained and unconstrained environments.

	Constrained	Unconstrained	Average
Sign 1	76.42%	75.46%	**75.94%**
Sign 2	63.97%	69.72%	**66.84%**
Sign 3	76.26%	86.69%	**81.48%**
Sign 4	75.47%	72.72%	**74.09%**
Sign 5	76.00%	72.66%	**74.33%**
Sign 6	76.84%	74.42%	**75.63%**
Sign 7	81.78%	60.64%	**71.21%**
Sign 8	80.99%	62.41%	**71.70%**
Sign 9	66.07%	65.41%	**65.74%**
Sign 10	72.85%	82.14%	**77.49%**
Sign 11	74.62%	86.17%	**80.40%**
Sign 12	53.47%	60.19%	**56.83%**
Sign 13	66.07%	58.33%	**62.20%**
Sign 14	76.08%	78.11%	**77.09%**
Sign 15	73.72%	77.84%	**75.78%**
Sign 16	71.54%	62.86%	**67.20%**
Sign 17	64.41%	65.96%	**65.18%**
Sign 18	76.54%	62.68%	**69.61%**
Sign 19	76.37%	71.40%	**73.88%**
Sign 20	54.73%	48.83%	**51.78%**
Sign 21	62.38%	66.01%	**64.20%**
Sign 22	67.36%	62.96%	**65.16%**
Sign 23	54.53%	75.26%	**64.89%**
Sign 24	68.64%	64.33%	**66.49%**
Sign 25	57.45%	71.62%	**64.54%**
Sign 26	64.25%	57.10%	**60.68%**
Sign 27	62.68%	62.44%	**62.56%**
Sign 28	58.88%	60.29%	**59.58%**
Sign 29	59.78%	68.17%	**63.98%**
Sign 30	56.77%	53.22%	**54.99%**

Table D.3: Compare participants using prototype T in constrained and unconstrained environments.

	Constrained	Unconstrained	Average
Person A	55.40%	70.71%	**63.06%**
Person B	33.77%	82.20%	**57.98%**
Person C	90.70%	57.77%	**74.24%**
Person D	92.56%	81.18%	**86.87%**
Person E	91.30%	75.34%	**83.32%**
Person F	94.91%	66.84%	**80.87%**
Person G	82.75%	62.07%	**72.41%**
Person H	73.68%	69.58%	**71.63%**
Person I	64.99%	75.79%	**70.39%**
Person J	55.37%	65.34%	**60.36%**
Person K	34.26%	72.34%	**53.30%**
Person L	72.36%	86.54%	**79.45%**
Person M	76.70%	76.67%	**76.69%**
Person N	58.62%	71.13%	**64.88%**
Person O	38.59%	73.01%	**55.80%**
Person P	86.67%	61.06%	**73.86%**
Person Q	33.61%	33.50%	**33.56%**
Person R	53.60%	50.10%	**51.85%**
Person S	87.35%	50.20%	**68.77%**
Person T	87.43%	75.99%	**81.71%**

D.2 Detection

Table D.4: Results on diffeernt random forest parameters using the original LBP with global histogram features.

Parameters	Accuracy
(100,25,4)	68.38%
(1000,100,4)	68.59%
(100,100,8)	69.33%
(100,100,16)	69.76%
(1000,25,16)	69.66%
(100,25,16)	69.76%
(100,25,32)	70.02%
(100,25,64)	70.19%
(100,25,128)	70.10%
(100,50,64)	70.29%
(100,100,64)	70.29%
(1000,50,64)	70.23%
(1000,25,64)	70.23%

Table D.5: Results on random forests using different LBP algorithms on pixels with and without filtering.

		Random Forests	
		Experiment 1(Without filtering)	Experiment 2(With filtering)
Global LBP	OriginalLBP	64.58%	70.29%
	ExtendedLBP	65.43%	72.25%
	UniformLBP	64.45%	70.72%
Local LBP	OriginalLBP	62.64%	65.85%
	ExtendedLBP	63.07%	66.02%
	UniformLBP	63.20%	65.62%
Image LBP	OriginalLBP	59.46%	59.26%
	ExtendedLBP	59.39%	59.77%
	UniformLBP	58.37%	56.01%

Table D.6: Results on different SVM kernel parameters using the original LBP with global histogram features.

Kernel(Parameters)	Accuracy
Linear (gamma=1,0; c=0,1)	47.22%
RBF (gamma=0,000150; c=2,5)	67.40%
Polynomial (degree=2; gamma=0.00001; coef=274,4; c=0,1)	50.17%
Sigmoid (gamma=0,00001; coef=1,4; c=312,5)	55.42%

Table D.7: Results on SVMs using different LBP algorithms on pixels with and without filtering.

		Support Vector Machines	
		Experiment 3(Without filtering)	Experiment 4(With filtering)
Global LBP	OriginalLBP	61.77%	67.40%
	ExtendedLBP	66.91%	69.04%
	UniformLBP	64.68%	62.26%
Local LBP	OriginalLBP	51.68%	57.22%
	ExtendedLBP	56.52%	60.37%
	UniformLBP	56.14%	59.97%
Image LBP	OriginalLBP	56.63%	51.87%
	ExtendedLBP	56.46%	51.57%
	UniformLBP	60.35%	59.07%

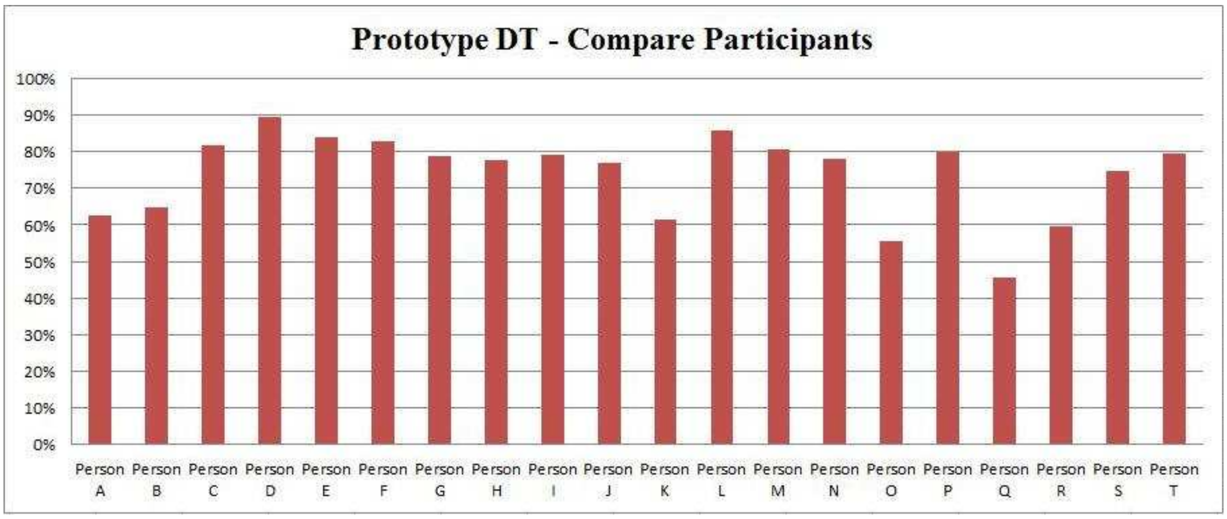


Figure D.3: The average hand-tracking success rates of prototype DT for each participant in both environments.

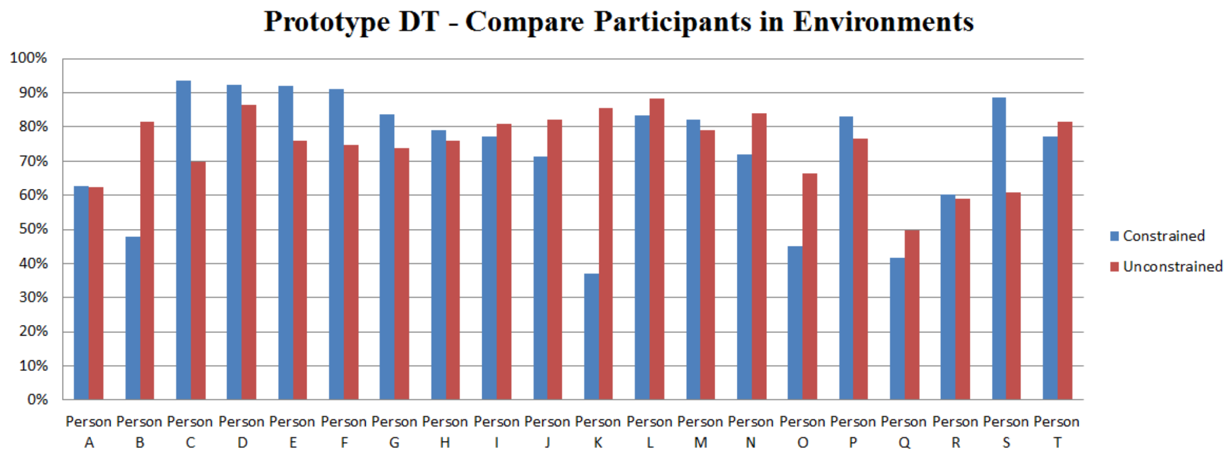


FIGURE D.4: A comparison of the average hand-tracking success rates of prototype DT for each participant in constrained and unconstrained environments.

Table D.8: Compare sign gestures using prototype DT in constrained and unconstrained environments.

	Constrained	Unconstrained	Average
Sign 1	84.28%	87.15%	**85.71%**
Sign 2	78.35%	81.23%	**79.79%**
Sign 3	79.70%	84.34%	**82.02%**
Sign 4	81.76%	79.62%	**80.69%**
Sign 5	90.12%	80.24%	**85.18%**
Sign 6	89.87%	84.66%	**87.27%**
Sign 7	88.32%	72.52%	**80.42%**
Sign 8	89.07%	70.02%	**79.54%**
Sign 9	61.65%	71.07%	**66.36%**
Sign 10	87.06%	89.54%	**88.30%**
Sign 11	81.28%	87.98%	**84.63%**
Sign 12	70.49%	66.82%	**68.66%**
Sign 13	78.17%	68.80%	**73.48%**
Sign 14	83.38%	87.32%	**85.35%**
Sign 15	78.45%	88.67%	**83.56%**
Sign 16	78.64%	77.46%	**78.05%**
Sign 17	66.75%	81.25%	**74.00%**
Sign 18	80.81%	82.49%	**81.65%**
Sign 19	86.16%	87.45%	**86.80%**
Sign 20	57.29%	63.19%	**60.24%**
Sign 21	72.18%	71.16%	**71.67%**
Sign 22	68.62%	63.36%	**65.99%**
Sign 23	57.66%	72.46%	**65.06%**
Sign 24	67.52%	65.42%	**66.47%**
Sign 25	59.66%	67.48%	**63.57%**
Sign 26	66.18%	60.95%	**63.57%**
Sign 27	60.53%	61.54%	**61.04%**
Sign 28	49.11%	60.51%	**54.81%**
Sign 29	46.79%	66.14%	**56.46%**
Sign 30	52.37%	60.48%	**56.42%**

Table D.9: Compare participants using prototype DT in constrained and unconstrained environments.

	Constrained	Unconstrained	Average
Person A	62.71%	62.31%	**62.51%**
Person B	47.93%	81.43%	**64.68%**
Person C	93.59%	69.80%	**81.69%**
Person D	92.43%	86.37%	**89.40%**
Person E	92.06%	75.82%	**83.94%**
Person F	90.95%	74.63%	**82.79%**
Person G	83.64%	73.72%	**78.68%**
Person H	79.13%	76.01%	**77.57%**
Person I	77.08%	80.85%	**78.96%**
Person J	71.42%	82.16%	**76.79%**
Person K	37.12%	85.67%	**61.40%**
Person L	83.51%	88.22%	**85.87%**
Person M	82.04%	79.07%	**80.56%**
Person N	71.83%	83.95%	**77.89%**
Person O	45.17%	66.30%	**55.73%**
Person P	83.23%	76.71%	**79.97%**
Person Q	41.81%	49.77%	**45.79%**
Person R	60.08%	58.88%	**59.48%**
Person S	88.60%	60.89%	**74.75%**
Person T	77.13%	81.66%	**79.39%**

D.3 Learning

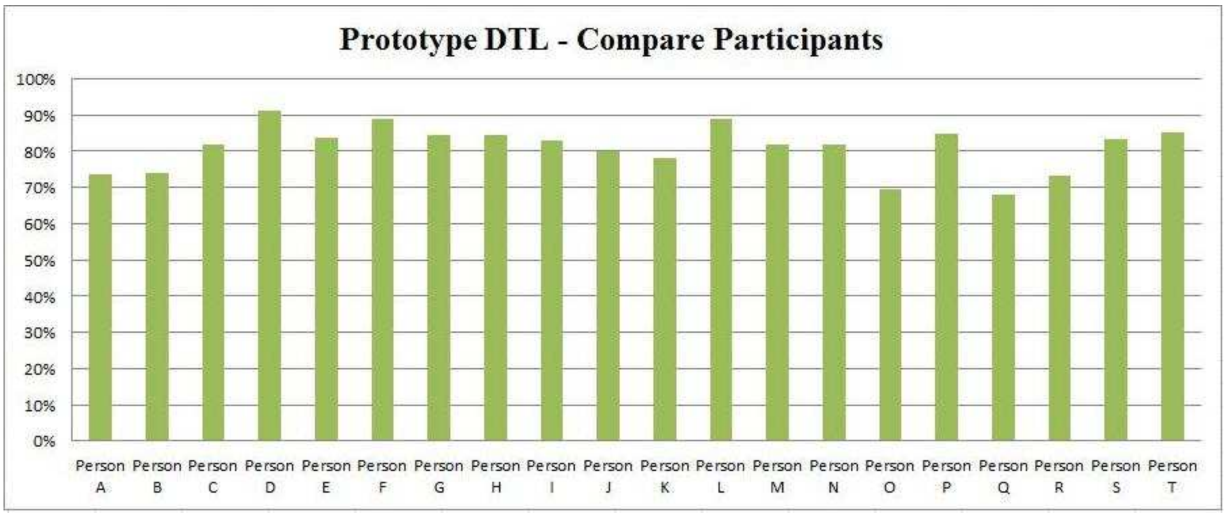


FIGURE D.5: The average hand-tracking success rates of prototype DTL for each participant in both environments.

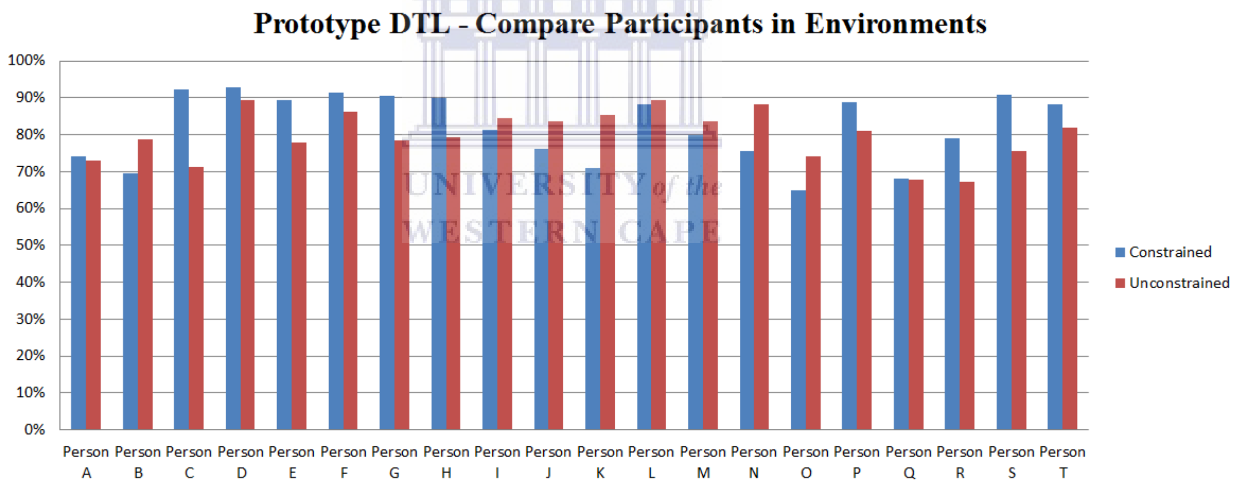


FIGURE D.6: A comparison of the average hand-tracking success rates of prototype DTL for each participant in constrained and unconstrained environments.

Table D.10: Compare sign gestures using prototype DTL in constrained and unconstrained environments.

	Constrained	Unconstrained	Average
Sign 1	90.06%	86.94%	**88.50%**
Sign 2	86.93%	84.95%	**85.94%**
Sign 3	84.62%	93.20%	**88.91%**
Sign 4	88.89%	89.28%	**89.09%**
Sign 5	91.15%	80.63%	**85.89%**
Sign 6	91.40%	83.38%	**87.39%**
Sign 7	92.39%	81.21%	**86.80%**
Sign 8	91.84%	75.79%	**83.82%**
Sign 9	80.94%	90.28%	**85.61%**
Sign 10	89.14%	89.23%	**89.19%**
Sign 11	86.49%	89.59%	**88.04%**
Sign 12	81.07%	73.80%	**77.44%**
Sign 13	89.64%	74.44%	**82.04%**
Sign 14	90.55%	89.07%	**89.81%**
Sign 15	87.52%	88.20%	**87.86%**
Sign 16	81.47%	85.22%	**83.34%**
Sign 17	87.16%	76.29%	**81.73%**
Sign 18	89.47%	83.13%	**86.30%**
Sign 19	92.82%	89.37%	**91.10%**
Sign 20	73.55%	71.31%	**72.43%**
Sign 21	75.48%	78.48%	**76.98%**
Sign 22	73.46%	70.12%	**71.79%**
Sign 23	72.66%	77.88%	**75.27%**
Sign 24	77.89%	68.79%	**73.34%**
Sign 25	67.31%	68.27%	**67.79%**
Sign 26	74.23%	69.36%	**71.79%**
Sign 27	71.14%	76.55%	**73.85%**
Sign 28	69.15%	66.39%	**67.77%**
Sign 29	65.86%	77.48%	**71.67%**
Sign 30	68.19%	66.25%	**67.22%**

Table D.11: Compare participants using prototype DTL in constrained and unconstrained environments.

	Constrained	Unconstrained	Average
Person A	74.01%	72.97%	**73.49%**
Person B	69.55%	78.64%	**74.10%**
Person C	92.37%	71.29%	**81.83%**
Person D	92.91%	89.45%	**91.18%**
Person E	89.27%	77.87%	**83.57%**
Person F	91.34%	86.31%	**88.83%**
Person G	90.45%	78.43%	**84.44%**
Person H	89.95%	79.19%	**84.57%**
Person I	81.21%	84.62%	**82.92%**
Person J	76.17%	83.65%	**79.91%**
Person K	70.88%	85.34%	**78.11%**
Person L	88.21%	89.48%	**88.84%**
Person M	79.84%	83.58%	**81.71%**
Person N	75.57%	88.30%	**81.93%**
Person O	64.87%	74.12%	**69.49%**
Person P	88.72%	81.11%	**84.91%**
Person Q	68.15%	67.76%	**67.96%**
Person R	78.95%	67.13%	**73.04%**
Person S	90.82%	75.47%	**83.14%**
Person T	88.40%	81.88%	**85.14%**

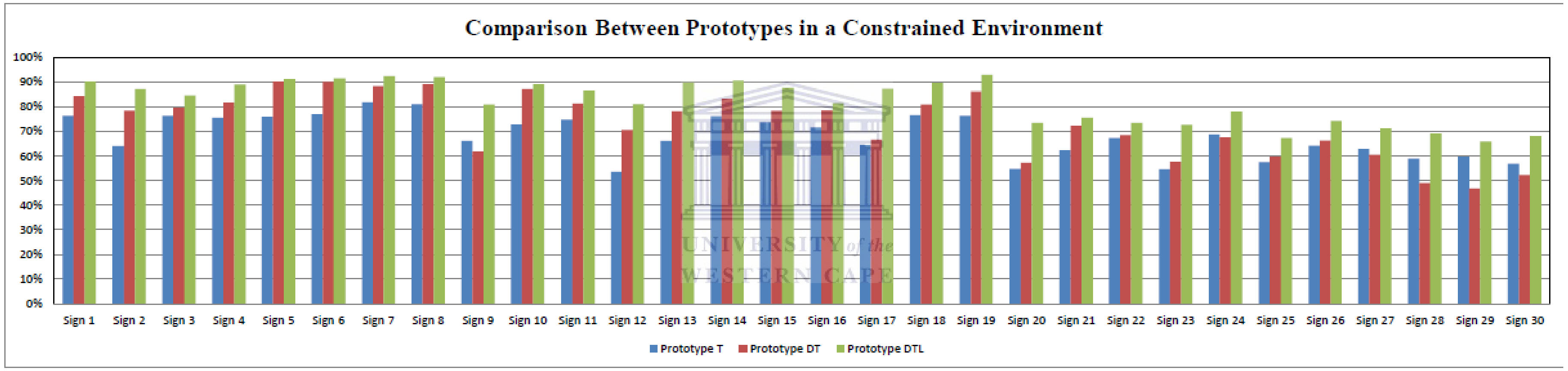


FIGURE D.7: A comparison between the hand-tracking accuracies of each prototype in the constrained environments.

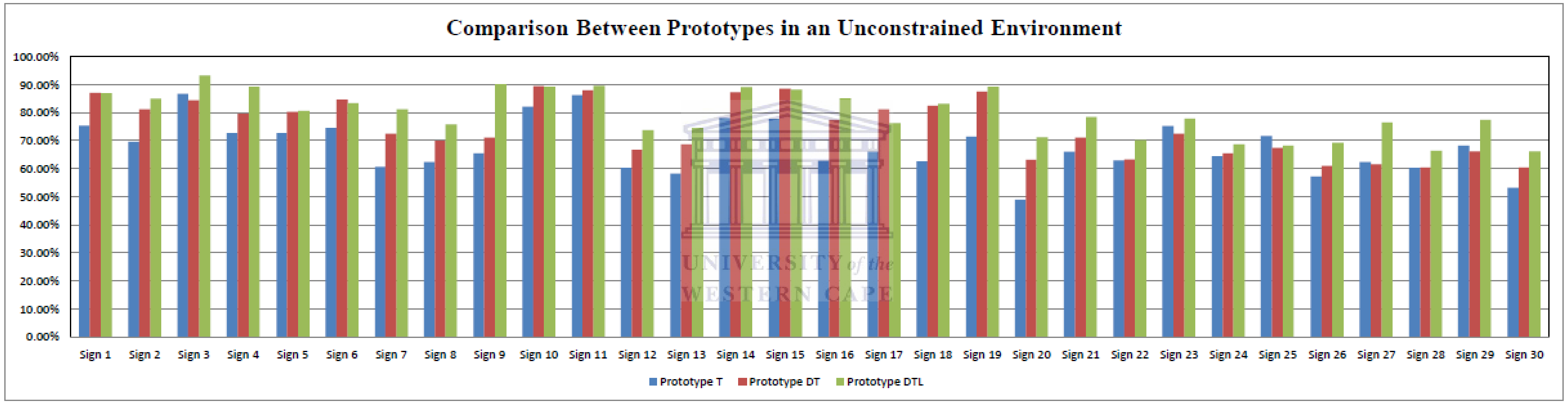


FIGURE D.8: A comparison between the hand-tracking accuracies of each prototype in the unconstrained environments.

Table D.12: A list of the non-significant differences between the results of the different hand detection methods. The difference between the results for the methods not listed in the table are all significant (This list was excluded as it a much longer list). In the table, the combination and _combination column values can be referred to in Table 6.4.

Observation	Combination	_Combination	Estimate	P-value	Adjustment	Adjusted p-value
1	1111	1112	-0.06905	0.0066	Tukey-Kramer	0.704
2	1111	1113	0.001847	0.8736	Tukey-Kramer	1
3	1111	1121	0.07589	0.0014	Tukey-Kramer	0.3181
4	1111	1122	0.04954	0.1122	Tukey-Kramer	0.9999
5	1111	1123	0.04042	0.0943	Tukey-Kramer	0.9997
12	1111	1221	-0.0766	0.0746	Tukey-Kramer	0.9989
13	1111	1222	-0.08228	0.0589	Tukey-Kramer	0.9969
14	1111	1223	-0.06716	0.1172	Tukey-Kramer	0.9999
18	1111	2111	0.09936	0.031	Tukey-Kramer	0.9746
19	1111	2112	-0.1252	0.0056	Tukey-Kramer	0.6636
20	1111	2113	-0.02596	0.4798	Tukey-Kramer	1
27	1111	2211	-0.1464	0.0002	Tukey-Kramer	0.0664
29	1111	2213	0.07951	0.0478	Tukey-Kramer	0.9931
36	1112	1113	0.0709	0.0053	Tukey-Kramer	0.6472
39	1112	1123	0.1095	0.0001	Tukey-Kramer	0.053
46	1112	1221	-0.00755	0.8615	Tukey-Kramer	1
47	1112	1222	-0.01323	0.762	Tukey-Kramer	1
48	1112	1223	0.001885	0.9651	Tukey-Kramer	1
52	1112	2111	0.1684	0.0002	Tukey-Kramer	0.0726
53	1112	2112	-0.05614	0.2141	Tukey-Kramer	1
54	1112	2113	0.04309	0.2534	Tukey-Kramer	1
61	1112	2211	-0.07734	0.0491	Tukey-Kramer	0.9937
62	1112	2212	-0.1538	0.0002	Tukey-Kramer	0.0733
63	1112	2213	0.1486	0.0002	Tukey-Kramer	0.0816
70	1113	1121	0.07404	0.0018	Tukey-Kramer	0.3749
71	1113	1122	0.0477	0.1238	Tukey-Kramer	1
72	1113	1123	0.03857	0.1077	Tukey-Kramer	0.9999

79	1113	1221	-0.07845	0.0685	Tukey-Kramer	0.9984
80	1113	1222	-0.08412	0.0537	Tukey-Kramer	0.9955
81	1113	1223	-0.06901	0.1085	Tukey-Kramer	0.9999
85	1113	2111	0.09751	0.0348	Tukey-Kramer	0.9813
86	1113	2112	-0.127	0.005	Tukey-Kramer	0.633
87	1113	2113	-0.02781	0.4515	Tukey-Kramer	1
94	1113	2211	-0.1482	0.0002	Tukey-Kramer	0.0599
96	1113	2213	0.07766	0.0532	Tukey-Kramer	0.9953
103	1121	1122	-0.02634	0.3603	Tukey-Kramer	1
104	1121	1123	-0.03547	0.0081	Tukey-Kramer	0.7566
105	1121	1131	0.1144	0.0007	Tukey-Kramer	0.1874
106	1121	1132	0.09403	0.005	Tukey-Kramer	0.6334
111	1121	1221	-0.1525	0.0004	Tukey-Kramer	0.122
112	1121	1222	-0.1582	0.0003	Tukey-Kramer	0.0944
113	1121	1223	-0.1431	0.0009	Tukey-Kramer	0.2267
114	1121	1231	0.1294	0.001	Tukey-Kramer	0.25
115	1121	1232	0.1073	0.0069	Tukey-Kramer	0.7169
117	1121	2111	0.02347	0.6073	Tukey-Kramer	1
119	1121	2113	-0.1018	0.0061	Tukey-Kramer	0.683
125	1121	2133	0.08427	0.0169	Tukey-Kramer	0.909
128	1121	2213	0.00362	0.9282	Tukey-Kramer	1
130	1121	2222	0.0825	0.0367	Tukey-Kramer	0.9839
131	1121	2223	0.1002	0.0247	Tukey-Kramer	0.9561
134	1121	2233	0.1373	0.0012	Tukey-Kramer	0.2902
135	1122	1123	-0.00913	0.7525	Tukey-Kramer	1
137	1122	1132	0.1204	0.0007	Tukey-Kramer	0.2046
142	1122	1221	-0.1261	0.0031	Tukey-Kramer	0.5012
143	1122	1222	-0.1318	0.0024	Tukey-Kramer	0.4376
144	1122	1223	-0.1167	0.0062	Tukey-Kramer	0.6889
146	1122	1232	0.1337	0.0008	Tukey-Kramer	0.2208
148	1122	2111	0.04982	0.2687	Tukey-Kramer	1
150	1122	2113	-0.0755	0.0506	Tukey-Kramer	0.9944

156	1122	2133	0.1106	0.0028	Tukey-Kramer	0.4755
159	1122	2213	0.02996	0.4545	Tukey-Kramer	1
161	1122	2222	0.1088	0.0065	Tukey-Kramer	0.702
162	1122	2223	0.1266	0.0039	Tukey-Kramer	0.5618
172	1123	1221	-0.117	0.0064	Tukey-Kramer	0.6971
173	1123	1222	-0.1227	0.0048	Tukey-Kramer	0.6194
174	1123	1223	-0.1076	0.0122	Tukey-Kramer	0.8508
176	1123	1232	0.1428	0.0004	Tukey-Kramer	0.119
178	1123	2111	0.05894	0.1939	Tukey-Kramer	1
179	1123	2112	-0.1656	0.0002	Tukey-Kramer	0.0753
180	1123	2113	-0.06638	0.0764	Tukey-Kramer	0.999
186	1123	2133	0.1197	0.0008	Tukey-Kramer	0.2107
189	1123	2213	0.03909	0.3332	Tukey-Kramer	1
191	1123	2222	0.118	0.0028	Tukey-Kramer	0.4787
192	1123	2223	0.1357	0.0024	Tukey-Kramer	0.4352
196	1131	1132	-0.02035	0.1819	Tukey-Kramer	1
204	1131	1231	0.01499	0.6851	Tukey-Kramer	1
205	1131	1232	-0.00707	0.8508	Tukey-Kramer	1
207	1131	2111	-0.09091	0.0511	Tukey-Kramer	0.9946
213	1131	2131	0.1232	0.0218	Tukey-Kramer	0.9429
214	1131	2132	0.1301	0.0155	Tukey-Kramer	0.8946
215	1131	2133	-0.03011	0.3227	Tukey-Kramer	1
218	1131	2213	-0.1108	0.0039	Tukey-Kramer	0.5647
219	1131	2221	0.09891	0.0425	Tukey-Kramer	0.9898
220	1131	2222	-0.03189	0.3842	Tukey-Kramer	1
221	1131	2223	-0.01415	0.7559	Tukey-Kramer	1
224	1131	2233	0.02291	0.5863	Tukey-Kramer	1
232	1132	1231	0.03534	0.3409	Tukey-Kramer	1
233	1132	1232	0.01328	0.7249	Tukey-Kramer	1
235	1132	2111	-0.07056	0.1269	Tukey-Kramer	1
241	1132	2131	0.1435	0.007	Tukey-Kramer	0.7209
242	1132	2132	0.1504	0.0047	Tukey-Kramer	0.6174

243	1132	2133	-0.00976	0.7537	Tukey-Kramer	1
246	1132	2213	-0.09041	0.0188	Tukey-Kramer	0.9243
247	1132	2221	0.1193	0.0141	Tukey-Kramer	0.8787
248	1132	2222	-0.01154	0.7563	Tukey-Kramer	1
249	1132	2223	0.006202	0.8915	Tukey-Kramer	1
252	1132	2233	0.04326	0.3021	Tukey-Kramer	1
259	1133	1231	-0.03073	0.4005	Tukey-Kramer	1
260	1133	1232	-0.05279	0.1577	Tukey-Kramer	1
261	1133	1233	0.1215	0.0006	Tukey-Kramer	0.1661
262	1133	2111	-0.1366	0.0033	Tukey-Kramer	0.5217
266	1133	2122	0.08092	0.0102	Tukey-Kramer	0.8118
267	1133	2123	0.09648	0.0037	Tukey-Kramer	0.5528
268	1133	2131	0.07746	0.1509	Tukey-Kramer	1
269	1133	2132	0.08438	0.118	Tukey-Kramer	0.9999
270	1133	2133	-0.07583	0.0106	Tukey-Kramer	0.8197
274	1133	2221	0.05319	0.2761	Tukey-Kramer	1
275	1133	2222	-0.07761	0.0324	Tukey-Kramer	0.9772
276	1133	2223	-0.05987	0.1878	Tukey-Kramer	1
279	1133	2233	-0.02281	0.5859	Tukey-Kramer	1
280	1211	1212	-0.09769	0.0003	Tukey-Kramer	0.0956
281	1211	1213	-0.01327	0.3468	Tukey-Kramer	1
289	1211	2112	0.1571	0.0003	Tukey-Kramer	0.1008
298	1211	2212	0.05937	0.0865	Tukey-Kramer	0.9995
306	1212	1213	0.08441	0.0016	Tukey-Kramer	0.3468
347	1213	2212	0.07264	0.036	Tukey-Kramer	0.9831
355	1221	1222	-0.00568	0.8266	Tukey-Kramer	1
356	1221	1223	0.009438	0.5303	Tukey-Kramer	1
361	1221	2112	-0.04858	0.2617	Tukey-Kramer	1
362	1221	2113	0.05064	0.2147	Tukey-Kramer	1
369	1221	2211	-0.06979	0.0933	Tukey-Kramer	0.9997
370	1221	2212	-0.1463	0.0007	Tukey-Kramer	0.2048
378	1222	1223	0.01511	0.5591	Tukey-Kramer	1

383	1222	2112	-0.04291	0.3274	Tukey-Kramer	1
384	1222	2113	0.05632	0.1902	Tukey-Kramer	1
391	1222	2211	-0.06411	0.1363	Tukey-Kramer	1
392	1222	2212	-0.1406	0.0013	Tukey-Kramer	0.2949
404	1223	2112	-0.05802	0.1771	Tukey-Kramer	1
405	1223	2113	0.0412	0.3125	Tukey-Kramer	1
412	1223	2211	-0.07923	0.0572	Tukey-Kramer	0.9965
413	1223	2212	-0.1557	0.0003	Tukey-Kramer	0.107
421	1231	1232	-0.02206	0.2317	Tukey-Kramer	1
423	1231	2111	-0.1059	0.0163	Tukey-Kramer	0.9035
427	1231	2122	0.1117	0.0028	Tukey-Kramer	0.4759
428	1231	2123	0.1272	0.0008	Tukey-Kramer	0.209
429	1231	2131	0.1082	0.0212	Tukey-Kramer	0.9397
430	1231	2132	0.1151	0.0142	Tukey-Kramer	0.88
431	1231	2133	-0.0451	0.1757	Tukey-Kramer	1
435	1231	2221	0.08392	0.0427	Tukey-Kramer	0.9899
436	1231	2222	-0.04688	0.2737	Tukey-Kramer	1
437	1231	2223	-0.02914	0.4242	Tukey-Kramer	1
440	1231	2233	0.007918	0.7897	Tukey-Kramer	1
442	1232	2111	-0.08384	0.055	Tukey-Kramer	0.9959
446	1232	2122	0.1337	0.0004	Tukey-Kramer	0.135
448	1232	2131	0.1302	0.0049	Tukey-Kramer	0.6241
449	1232	2132	0.1372	0.003	Tukey-Kramer	0.4973
450	1232	2133	-0.02304	0.4977	Tukey-Kramer	1
453	1232	2213	-0.1037	0.0004	Tukey-Kramer	0.1286
454	1232	2221	0.106	0.009	Tukey-Kramer	0.7841
455	1232	2222	-0.02482	0.5635	Tukey-Kramer	1
456	1232	2223	-0.00708	0.8449	Tukey-Kramer	1
459	1232	2233	0.02998	0.3319	Tukey-Kramer	1
463	1233	2121	0.1556	0.0004	Tukey-Kramer	0.1304
464	1233	2122	-0.04056	0.2594	Tukey-Kramer	1
465	1233	2123	-0.025	0.4958	Tukey-Kramer	1

466	1233	2131	-0.04402	0.3579	Tukey-Kramer	1
467	1233	2132	-0.0371	0.4385	Tukey-Kramer	1
472	1233	2221	-0.06829	0.1051	Tukey-Kramer	0.9998
479	2111	2113	-0.1253	0.0023	Tukey-Kramer	0.4242
485	2111	2133	0.0608	0.1835	Tukey-Kramer	1
488	2111	2213	-0.01985	0.6431	Tukey-Kramer	1
490	2111	2222	0.05902	0.185	Tukey-Kramer	1
491	2111	2223	0.07676	0.057	Tukey-Kramer	0.9964
494	2111	2233	0.1138	0.0056	Tukey-Kramer	0.6608
495	2112	2113	0.09922	0.0166	Tukey-Kramer	0.9057
502	2112	2211	-0.0212	0.6158	Tukey-Kramer	1
503	2112	2212	-0.0977	0.0189	Tukey-Kramer	0.9252
517	2113	2211	-0.1204	0.0005	Tukey-Kramer	0.1548
519	2113	2213	0.1055	0.0044	Tukey-Kramer	0.5991
537	2121	2231	-0.00937	0.8414	Tukey-Kramer	1
538	2121	2232	0.002554	0.9567	Tukey-Kramer	1
540	2122	2123	0.01556	0.6265	Tukey-Kramer	1
541	2122	2131	-0.00346	0.9462	Tukey-Kramer	1
542	2122	2132	0.00346	0.9463	Tukey-Kramer	1
547	2122	2221	-0.02773	0.5609	Tukey-Kramer	1
549	2122	2223	-0.1408	0.0017	Tukey-Kramer	0.3514
552	2122	2233	-0.1037	0.0131	Tukey-Kramer	0.865
553	2123	2131	-0.01902	0.7021	Tukey-Kramer	1
554	2123	2132	-0.0121	0.8081	Tukey-Kramer	1
559	2123	2221	-0.04329	0.3452	Tukey-Kramer	1
561	2123	2223	-0.1563	0.0004	Tukey-Kramer	0.1411
564	2123	2233	-0.1193	0.0045	Tukey-Kramer	0.6018
565	2131	2132	0.006921	0.3827	Tukey-Kramer	1
566	2131	2133	-0.1533	0.0039	Tukey-Kramer	0.5619
570	2131	2221	-0.02427	0.4146	Tukey-Kramer	1
571	2131	2222	-0.1551	0.0011	Tukey-Kramer	0.2699
572	2131	2223	-0.1373	0.0001	Tukey-Kramer	0.0567

573	2131	2231	0.1903	0.0006	Tukey-Kramer	0.1839
574	2131	2232	0.2022	0.0003	Tukey-Kramer	0.0982
575	2131	2233	-0.1003	0.015	Tukey-Kramer	0.8889
576	2132	2133	-0.1602	0.0026	Tukey-Kramer	0.4565
580	2132	2221	-0.03119	0.2942	Tukey-Kramer	1
581	2132	2222	-0.162	0.0007	Tukey-Kramer	0.1876
583	2132	2231	0.1834	0.001	Tukey-Kramer	0.2554
584	2132	2232	0.1953	0.0005	Tukey-Kramer	0.1453
585	2132	2233	-0.1072	0.0094	Tukey-Kramer	0.7947
588	2133	2213	-0.08065	0.0326	Tukey-Kramer	0.9776
589	2133	2221	0.129	0.0078	Tukey-Kramer	0.7476
590	2133	2222	-0.00178	0.9597	Tukey-Kramer	1
591	2133	2223	0.01596	0.7254	Tukey-Kramer	1
594	2133	2233	0.05302	0.184	Tukey-Kramer	1
595	2211	2212	-0.07649	0.0144	Tukey-Kramer	0.8816
611	2213	2222	0.07888	0.0539	Tukey-Kramer	0.9956
612	2213	2223	0.09662	0.0093	Tukey-Kramer	0.7913
615	2213	2233	0.1337	0.0002	Tukey-Kramer	0.0714
616	2221	2222	-0.1308	0.0052	Tukey-Kramer	0.6432
617	2221	2223	-0.1131	0.001	Tukey-Kramer	0.2456
620	2221	2233	-0.076	0.0435	Tukey-Kramer	0.9905
621	2222	2223	0.01774	0.6958	Tukey-Kramer	1
624	2222	2233	0.05479	0.2217	Tukey-Kramer	1
627	2223	2233	0.03705	0.3181	Tukey-Kramer	1
628	2231	2232	0.01192	0.3661	Tukey-Kramer	1

Appendix E

Publications

This section lists publications originating from parts of work in this thesis:

- I. Achmed, I.M. Venter and P. Eisert, “A Skin-Invariant Modeling Framework”, in *Journal of Visual Communication and Image Representation*, May 2013. (Submitted)
- I. Achmed, I.M. Venter and P. Eisert, “Improved Hand-Tracking Framework with a Recovery Mechanism”, in *Proceedings of the Southern Africa Telecommunication Networks and Applications Conference*, Stellenbosch, South Africa, 1-4 September , pp. 185-190, 2013. (Best Paper Award)
- I. Achmed, I.M. Venter and P. Eisert, “A Framework for Independent Hand Tracking in Unconstrained Environments”, in *Proceedings of the Southern Africa Telecommunications, Networks and Application Conference*, George, South Africa, 2-5 September, pp. 159-164, 2012.
- I. Achmed and I.M. Venter, “A Discriminative Approach to South African Sign Language Recognition from a Monocular 2D View”, in *Proceedings of the Southern Africa Telecommunications, Networks and Application Conference*, East London, South Africa, 4-7 September, 2011.

The following lists presentations resulting from parts of work in this thesis:

- I. Achmed, I.M. Venter and P. Eisert, “Independent hand-tracking from a single 2D view: its application to South African sign language”, *Post-graduate Research Open Day, New Science Auditorium*, University of the Western Cape, Cape Town, South Africa, 30 October, 2013.

- I. Achmed, I.M. Venter, “A discriminative approach in hand-tracking towards the recognition of South African sign language”, in *Design, Development and Research Conference Colloquium*, Cape Peninsula University of Technology, Cape Town, South Africa, 26–27 September, 2011.

Bibliography

[1] I. Achmed, “Upper body pose recognition and estimation towards the translation of South African sign language,” Master’s Thesis, Computer Science, University of the Western Cape, Cape Town, South Africa, December 2010.

[2] I. Achmed and J. Connan, “Upper body pose estimation towards the translation of South African sign language,” in *Proceedings of the Southern Africa Telecommunication Networks and Applications Conference*, Stellenbosch, South Africa, September 2010, pp. 427–432.

[3] I. Achmed, I. Venter, and P. Eisert, “A dynamic skin-invariant modelling framework,” May 2013, submitted for publication. [Online]. Available: http://www.cs.uwc.ac.za/~iachmed/Imran_Achmed_Journal_Article.pdf

[4] I. Achmed, I. M. Venter, and P. Eisert, “A framework for independent hand tracking in unconstrained environments,” in *Proceedings of the Southern Africa Telecommunication Networks and Applications Conference*, George, South Africa, September 2012, pp. 159–164.

[5] M. Agrawal, K. Konolige, and M. R. Blas, “Censure: Center surround extremas for realtime feature detection and matching,” in *European Conference on Computer Vision*. Marseille, France: Springer, 2008, pp. 102–115.

[6] T. Ahonen, A. Hadid, and M. Pietikäinen, “Face recognition with local binary patterns,” in *European Conference on Computer Vision*. Prague, Czech Republic: Springer, May 2004, pp. 469–481.

[7] J. Aker and I. Mbiti, “Mobile phones and economic development in africa,” *The Journal of Economic Perspectives*, vol. 24, no. 3, pp. 207–232, 2010.

[8] A. Alahi, R. Ortiz, and P. Vandergheynst, "Freak: Fast retina keypoint," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, USA, June 2012, pp. 510–517.

[9] A. A. Argyros and M. I. Lourakis, "Real-time tracking of multiple skin-colored objects with a possibly moving camera," in *European Conference on Computer Vision.* Prague, Czech Republic: Springer, May 2004, pp. 368–379.

[10] A. A. Argyros and M. I. Lourakis, "Tracking multiple colored blobs with a moving camera," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 2. San Diego, CA, USA: IEEE, June 2005, p. 1178.

[11] A. Aristidou and J. Lasenby, "Motion capture with constrained inverse kinematics for real-time hand tracking," in *Fourth International Symposium on Communications, Control and Signal Processing (ISCCSP).* IEEE, 2010, pp. 1–5.

[12] L. Auret, "Process monitoring and fault diagnosis using random forests," Ph.D. dissertation, University of Stellenbosch, South Africa, December 2010.

[13] E. Babbie, *The basics of social research.* Cengage Learning, 2008.

[14] D. L. Baggio, S. Emami, D. M. Escrivá, K. Ievgen, N. Mahmood, J. Saragih, and R. Shilkrot, *Mastering OpenCV with Practical Computer Vision Projects.* Packt Publishing, Limited, 2012.

[15] I. Barandiaran, M. Graña, and M. Nieto, "An empirical evaluation of interest point detectors," *Cybernetics and Systems*, vol. 44, no. 2-3, pp. 98–117, 2013.

[16] M. Bauml and R. Stiefelhagen, "Evaluation of local features for person re-identification in image sequences," in *Eight IEEE International Conference on Advanced Video and Signal-Based Surveillance.* IEEE, 2011, pp. 291–296.

[17] H. Bay, T. Tuytelaars, and L. Van Gool, "Surf: Speeded up robust features," in *European Conference on Computer Vision.* Graz, Austria: Springer, May 2006, pp. 404–417.

[18] J. Becker and B. Niehaves, "Epistemological perspectives on is research: a framework for analysing and systematizing epistemological assumptions," *Information Systems Journal*, vol. 17, no. 2, pp. 197–214, 2007.

[19] M. Berbar, “Novel colors correction approaches for natural scenes and skin detection techniques,” *International Journal of Video and Image Processing and Network Security*, vol. 11, no. 2, 2011.

[20] G. Bradski and A. Kaehler, *Learning OpenCV: Computer vision with the OpenCV library.* O’Reilly Media, 2008.

[21] L. Breiman, “Random forests,” *Machine learning*, vol. 45, no. 1, pp. 5–32, 2001.

[22] A. Bryman and E. Bell, *Business research methods.* Oxford University Press, USA, 2007.

[23] P. Buehler, “Automatic learning of British sign language from signed tv broadcasts,” PhD Dissertation, University of Oxford, Oxford, 2010.

[24] M. Calonder, V. Lepetit, C. Strecha, and P. Fua, “BRIEF: Binary robust independent elementary features,” in *European Conference on Computer Vision.* Crete, Greece: Springer, 2010, pp. 778–792.

[25] L. Cerman and V. Hlaváč, “Tracking with context as a semi-supervised learning and labeling problem,” in *21st International Conference on Pattern Recognition.* IEEE, 2012, pp. 2124–2127.

[26] L. Cerman, J. Matas, and V. Hlaváč, “Sputnik tracker: Having a companion improves robustness of the tracker,” in *Image Analysis.* Springer, 2009, pp. 291–300.

[27] R. Cespi, A. Kolb, and M. Lindner, “Hand tracking based on hierarchical clustering of range data,” Centre for Telematics and Information Technology, University of Twente, Technical Report arXiv:1110.5450, October 2011.

[28] X. Chai, Z. Xu, Q. Li, B. Ma, and X. Chen, “Robust hand tracking by integrating appearance, location and depth cues,” in *Proceedings of the Fourth International Conference on Internet Multimedia Computing and Service.* ACM, 2012, pp. 60–65.

[29] C. H. Chan, “Multi-scale local binary pattern histogram for face recognition,” PhD Dissertation, Centre for Vision, Speech and Signal Processing, School of Electronics and Physical Sciences, University of Surrey, September 2008.

[30] C. Chang and C.-J. Lin, *LIBSVM: A library for support vector machines*, 2001, [Online] Software Available at http://www.csie.ntu.edu.tw/~cjlin/libsvm.

[31] A. Cheddad, J. Condell, K. Curran, and P. McKevitt, "A new colour space for skin tone detection," in *Proceedings of the 16th IEEE International Conference on Image Processing.* IEEE, 2009, pp. 497–500.

[32] B.-J. Chen, C.-M. Huang, T.-E. Tseng, and L.-C. Fu, "Robust head and hands tracking with occlusion handling for human machine interaction," in *IEEE/RSJ International Conference on Intelligent Robots and Systems.* IEEE, 2012, pp. 2141–2146.

[33] Q. Chen, "Real-Time Vision-Based Hand Tracking and Gesture Recognition," PhD Dissertation, University of Ottawa, Canada, 2008.

[34] C.-S. Chua, H. Guan, and Y.-K. Ho, "Model-based 3D hand posture estimation from a single 2D image," *Image and Vision Computing*, vol. 20, no. 3, pp. 191–202, 2002.

[35] J. Collis and R. Hussey, *Business research: A practical guide for undergraduate and postgraduate students.* Palgrave Macmillan, 2009.

[36] T. Coogan, G. Awad, J. Han, and A. Sutherland, "Real time hand gesture recognition including hand segmentation and tracking," in *Advances in Visual Computing.* Springer, 2006, pp. 495–504.

[37] N. Cristianini, "Support vector and kernel machines," *Tutorial at the 18th International Conference on Machine Learning*, 2001.

[38] M. Crotty, *The foundations of social research: Meaning and perspective in the research process.* Sage Publications Ltd, 1998.

[39] F. Dadgostar and A. Sarrafzadeh, "A fast real-time skin detector for video sequences," *Image Analysis and Recognition*, pp. 804–811, 2005.

[40] F. Dadgostar and A. Sarrafzadeh, "An adaptive real-time skin detector based on Hue thresholding: A comparison on two motion tracking methods," *Pattern Recognition Letters*, vol. 27, no. 12, pp. 1342–1352, 2006.

[41] S. Davies, "Exploring the views and experiences of early years practitioners with regard to consultation with children under five," Masters Thesis, University of Liverpool (University of Chester), 2006.

[42] M. de La Gorce, N. Paragios, and D. Fleet, "Model-based hand tracking with texture, shading and self-occlusions," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition.* Anchorage, Alaska, USA: IEEE, June 2008, pp. 1–8.

[43] DeafSA, "Deaf federation of South Africa," January 2010, [Online] Available at http://www.deafsa.co.za/index-2.html.

[44] M. Donoser and H. Bischof, "Real time appearance based hand tracking," in *Proceedings of International Conference on Pattern Recognition.* Tampa, Florida, USA: Citeseer, December 2008.

[45] C. Doukim, J. Dargham, and A. Chekima, "Comparison of three colour spaces in skin detection," *Borneo Science*, vol. 24, pp. 75–81, March 2009.

[46] G. Duan, H. Ai, S. Cao, and S. Lao, "Group tracking: exploring mutual relations for multiple object tracking," in *European Conference on Computer Vision.* Firenze, Italy: Springer, October 2012, pp. 129–143.

[47] A. Elgammal, C. Muang, and D. Hu, "Skin detection—a short tutorial," *Encyclopedia of Biometrics, Springer, Verlag*, 2009.

[48] M. Elmezain, A. Al-Hamadi, and B. Michaelis, "A robust method for hand gesture segmentation and recognition using forward spotting scheme in conditional random fields," in *20th International Conference on Pattern Recognition.* IEEE, 2010, pp. 3850–3853.

[49] M. Elmezain, A. Al-Hamadi, R. Niese, and B. Michaelis, "A robust method for hand tracking using mean-shift algorithm and kalman filter in stereo color image sequences," in *Proceedings of the International Conference on Computer Vision, Image and Signal Processing*, 2009, pp. 24–28.

[50] L. Fei-Fei, R. Fergus, and P. Perona, "Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object

categories," *Computer Vision and Image Understanding*, vol. 106, no. 1, pp. 59–70, 2007.

[51] M. Filhol, "Zebedee: A lexical description model for sign language synthesis," Laboratory for Mechanics and Engineering Sciences, Technical Report, 2009.

[52] B. Fitzgerald and D. Howcroft, "Towards dissolution of the IS research debate: from polarization to polarity," *Journal of information technology*, vol. 13, no. 4, pp. 313–326, 1998.

[53] M. Fleck, D. Forsyth, and C. Bregler, "Finding naked people," in *Proceedings of the Fourth European Conference on Computer Vision*, vol. 2. Springer-Verlag, 1996, pp. 593–602.

[54] A. Fogelton, "Real-time hand tracking using modificated flocks of features algorithm," *Information Sciences and Technologies Bulletin of the ACM Slovakia Special Section on Student Research in Informatics and Information Technologies*, vol. 3, no. 2, pp. 1–5, 2011.

[55] V. Frati and D. Prattichizzo, "Using Kinect for hand tracking and rendering in wearable haptics," in *World Haptics Conference*. IEEE, 2011, pp. 317–321.

[56] X. Fu and W. Wei, "Centralized binary patterns embedded with image euclidean distance for facial expression recognition," in *Fourth International Conference on Natural Computation*, vol. 4. Jinan, Shandong, China: IEEE, October 2008, pp. 115–119.

[57] M. D. Gall, W. R. Borg, and J. P. Gall, *Educational research: An introduction* . Longman Publishing, 1996.

[58] S. Gauglitz, T. Höllerer, and M. Turk, "Evaluation of interest point detectors and feature descriptors for visual tracking," *International Journal of Computer Vision*, vol. 94, no. 3, pp. 335–360, 2011.

[59] R. Genuer, J.-M. Poggi, and C. Tuleau-Malot, "Variable selection using random forests," *Pattern Recognition Letters*, vol. 31, no. 14, pp. 2225–2236, 2010.

[60] M. Ghaziasgar, "The use of mobile phones as service-delivery devices in a sign language machine translation system," Master's Thesis, Computer Science, University of the Western Cape, Cape Town, South Africa, June 2010.

[61] M. Glaser and W. Tucker, "Telecommunications bridging between Deaf and hearing users in South Africa," *in Proceedings of the Conference and Workshop on Assistive Technologies for People with Vision and Hearing Impairments*, 2004.

[62] R. Goede, "A framework for the explicit use of specific systems thinking methodologies in data-driven decision support system development," PhD Dissertation, University of Pretoria, November 2005.

[63] R. Gordon Jr, "Ethnologue: Languages of the world," *[Online] Available at http://www.ethnologue.com*, 2005.

[64] H. Grabner, J. Matas, L. Van Gool, and P. Cattin, "Tracking the invisible: Learning where the object might be," in *IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2010, pp. 1285–1292.

[65] R. Gregory, "Design science research and the grounded theory method: Characteristics, differences, and complementary uses," *Proceedings of the 18th European Conference on Information System*, 2010.

[66] Z. Guo, L. Zhang, and D. Zhang, "Rotation invariant texture classification using LBP variance (LBPV) with global matching," *Pattern Recognition*, vol. 43, no. 3, pp. 706–719, 2010.

[67] S. Gupta, "Support vector machines based modelling of concrete strength," *International Journal of Computer Systems Science and Engineering*, vol. 3, no. 1, 2008.

[68] S. Hadfield and R. Bowden, "Generalised pose estimation using depth," in *Trends and Topics in Computer Vision*. Springer, 2012, pp. 312–325.

[69] C. Harris and M. Stephens, "A combined corner and edge detector." in *Fourth Alvey Vision Conference*, vol. 15. Manchester, UK, 1988, p. 50.

[70] O. Henia, M. Hariti, and S. Bouakaz, "A two-step minimization algorithm for model-based hand tracking," in *18th International Conference on Computer Graphics, Visualization and Computer Vision*, Czech Republic, February 2010.

[71] A. Hevner and S. Chatterjee, *Design research in information systems: theory and practice*. Springer, 2010, vol. 22.

[72] A. R. Hevner, S. T. March, J. Park, and S. Ram, “Design science in information systems research,” *Management Information Systems Quarterly*, vol. 28, no. 1, pp. 75–105, 2004.

[73] S. Howard, *Finger talk-South African sign language dictionary.* South Africa: Mondi, 2008.

[74] M. Hrúz, J. Trojanová, and M. Železnỳ, “Local binary pattern based features for sign language recognition,” *Pattern Recognition and Image Analysis*, vol. 22, no. 4, pp. 519–526, 2012.

[75] C. Hsu, C. Chang, and C. Lin, “A practical guide to support vector classification,” National Taiwan University, Technical Report, 2003.

[76] C. Hue, J.-P. Le Cadre, and P. Pérez, “Sequential monte carlo methods for multiple target tracking and data fusion,” *IEEE Transactions on Signal Processing*, vol. 50, no. 2, pp. 309–325, 2002.

[77] J. Hughes, *The philosophy of social research.* Addison-Wesley Longman Ltd, 1990.

[78] L. M. Huglin, “The relationship between personal epistemology and learning style in adult learners,” PhD Dissertation, University of Idaho, 2003.

[79] M. Jones and J. Rehg, “Statistical color models with application to skin detection,” *International Journal of Computer Vision*, vol. 46, no. 1, pp. 81–96, 2002.

[80] P. Kakumanu, S. Makrogiannis, and N. Bourbakis, “A survey of skin-color modeling and detection methods,” *Pattern Recognition*, vol. 40, no. 3, pp. 1106–1122, 2007.

[81] Z. Kalal, “Tracking-learning-detection,” PhD Dissertation, University of Surrey, Guildford, Surrey, September 2010.

[82] W. Kelly, A. Donnellan, and D. Molloy, “A review of skin detection techniques for objectionable images,” *Information Technology and Telecommunications 2007 General Chairs Letter*, p. 40, 2007.

[83] C. Kerdvibulvech and H. Saito, “Model-based hand tracking by chamfer distance and adaptive color learning using particle filter,” *Journal on Image and Video Processing*, vol. 2009, p. 6, 2009.

[84] R. Khan, Z. Khan, M. Aamir, and S. Sattar, “Static filtered skin detection,” *International Journal of Computer Science Issues*, vol. 9, no. 3, pp. 257–261, March 2012.

[85] T.-K. Kim, S.-F. Wong, and R. Cipolla, “Tensor canonical correlation analysis for action classification,” in *IEEE Conference on Computer Vision and Pattern Recognition.* Minneapolis, Minnesota, USA: IEEE, June 2007, pp. 1–8.

[86] M. Kölsch and M. Turk, “Flocks of features for tracking articulated objects,” in *Real-Time Vision for Human-Computer Interaction.* Springer, 2005, pp. 67–83.

[87] P. P. Kumar, P. Vadakkepat, and A. P. Loh, “Hand posture and face recognition using a fuzzy-rough approach,” *International Journal of Humanoid Robotics*, vol. 7, no. 03, pp. 331–356, 2010.

[88] R. Laganière, *OpenCV 2 computer vision application programming cookbook.* Packt Publishing, 2011.

[89] Y. Lao and Y. F. Zheng, “Tracking highly correlated targets through statistical multiplexing,” *Image and Vision Computing*, vol. 29, no. 12, pp. 803–817, 2011.

[90] M. Lee and I. Cohen, “Human upper body pose estimation in static images,” *in Proceedings of the Eighth European Conference on Computer Vision*, pp. 126–138, May 2004.

[91] C. Leistner, “Semi-supervised ensemble methods for computer vision,” PhD Dissertation, Institute for Computer Graphics and Vision, Graz University of Technology, June 2010.

[92] S. Leutenegger, M. Chli, and R. Y. Siegwart, “BRISK: Binary robust invariant scalable keypoints,” in *IEEE International Conference on Computer Vision.* Barcelona, Spain: IEEE, November 2011, pp. 2548–2555.

[93] S. Liao, M. W. Law, and A. C. Chung, “Dominant local binary patterns for texture classification,” *IEEE Transactions on Image Processing*, vol. 18, no. 5, pp. 1107–1118, January 2009.

[94] T. Lindahl, “Study of local binary patterns,” Masters Thesis, Department of Science and Technology, Linköpings University, 2007.

[95] B. Liu, S. Gould, and D. Koller, "Single image depth estimation from predicted semantic labels," in *IEEE Conference on Computer Vision and Pattern Recognition.* San Francisco, USA: IEEE, June 2010, pp. 1253–1260.

[96] H. Liu, Y. Wang, and X. Lu, "A method to choose kernel function and its parameters for support vector machines," in *Proceedings of the 2005 International Conference on Machine Learning and Cybernetics*, vol. 7. Guangzhou, China: IEEE, August 2005, pp. 4277–4280.

[97] H. Liu, W. Cui, and R. Ding, "Robust hand tracking with hough forest and multi-cue flocks of features," in *Advances in Visual Computing.* Springer, 2012, vol. 7432, pp. 458–467.

[98] Y. Liu and P. Zhang, "Hand gesture tracking using particle filter with multiple features," in *Proceedings of the International Symposium on Intelligent Information Systems and Applications*, Qingdao, China, October 2009, pp. 28–30.

[99] D. G. Lowe, "Object recognition from local scale-invariant features," in *Proceedings of the Seventh IEEE International Conference on Computer Vision*, vol. 2. Kerkyra, Corfu, Greece: IEEE, September 1999, pp. 1150–1157.

[100] S. Lu, D. Metaxas, D. Samaras, and J. Oliensis, "Using multiple cues for hand tracking and model refinement," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 2. Madison, Wisconsin: IEEE, June 2003, pp. 443–450.

[101] Y. Luo, L. Li, B. Zhang, and H. Yang, "Video hand tracking algorithm based on hybrid camshift and kalman filter," *Application Research of Computers*, vol. 26, no. 3, pp. 1163–1165, 2009.

[102] Y. Ma, W. Jia, C. Li, J. Yang, Z.-H. Mao, and M. Sun, "Magnetic hand motion tracking system for human-machine interaction," *Electronics letters*, vol. 46, no. 9, pp. 621–623, 2010.

[103] J. MacCormick and M. Isard, "Partitioned sampling, articulated objects, and interface-quality hand tracking," in *Proceedings of the 6th European Conference on Computer Vision.* Dublin, Ireland: Springer, June 2000, pp. 3–19.

[104] E. Maggio and A. Cavallaro, *Video tracking: theory and practice.* Wiley, 2011.

[105] J. Matas, O. Chum, M. Urban, and T. Pajdla, "Robust wide-baseline stereo from maximally stable extremal regions," *Image and Vision Computing*, vol. 22, no. 10, pp. 761–767, 2004.

[106] R. Mattheij and E. Postma, "Feature-based hand detection in visual images," in *Proceedings of TiGeR 2013: The Combined Meeting of the 10th International Gesture Workshop and the 3rd Gesture and Speech in Interaction Conference*, Tilburg, The Netherlands, June 2013.

[107] G. McAllister, S. J. McKenna, and I. W. Ricketts, "Hand tracking for behaviour understanding," *Image and Vision Computing*, vol. 20, no. 12, pp. 827–840, 2002.

[108] O. Miksik and K. Mikolajczyk, "Evaluation of local detectors and descriptors for fast feature matching," in *21st International Conference on Pattern Recognition.* Tsukuba Science City, Japan: IEEE, November 2012, pp. 2681–2684.

[109] A. Mittal, A. Zisserman, and P. H. S. Torr, "Hand detection using multiple proposals," in *Proceedings of the 22nd British Machine Vision Conference*, Scotland, United Kingdom, August 2011.

[110] D. Mohr and G. Zachmann, "Real-time hand tracking for natural and direct interaction," in *Proceedings of the 28th ACM Conference on Human Factors in Computing Systems*, Atlanta, GA, USA, April 2010, pp. 1–6.

[111] D. Mohr, "Model-based high-dimensional pose estimation with application to hand tracking," PhD Dissertation, University of Bremen, Bremen, 2012.

[112] M. Muja and D. G. Lowe, "Fast approximate nearest neighbors with automatic algorithm configuration." in *Proceedings of the International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications*, Lisboa, Portugal, February 2009, pp. 331–340.

[113] K. Nallaperumal, S. Ravi, C. N. K. Babu, R. K. Selvakumar, A. L. Fred, S. Christopher, and S. S. Vinsley, "Skin detection using color pixel classification with application to face detection: A comparative study," in *Proceedings of the International Conference on Computational Intelligence and Multimedia Applications*, vol. 3. Sivakasi, Tamilnadu, India: IEEE, December 2007, pp. 436–441.

[114] P. Neskovic, D. Schuster, and L. N. Cooper, "Biologically inspired recognition system for car detection from real-time video streams," in *Neural Information Processing: Research and Development.* Springer – Verlag, 2004, vol. 152, no. 3, pp. 320–333.

[115] D. T. Nguyen, "Human detection from images and videos," PhD Dissertation, School of Computer Science and Software Engineering, University of Wollongong, 2012.

[116] T. T. Nguyen, N. D. Binh, and H. Bischof, "An active boosting-based learning framework for real-time hand detection," in *Proceedings of the 8th IEEE International Conference on Automatic Face and Gesture Recognition.* Amsterdam, The Netherlands: IEEE, September 2008, pp. 1–6.

[117] B. Niehaves, "Epistemological perspectives on multi-method information systems research," *Proceedings of the 13th European Conference on Information Systems*, pp. 1584–1595, May 2005.

[118] W. Noble, "What is a support vector machine?" *Nature Biotechnology*, vol. 24, no. 12, pp. 1565–1567, 2006.

[119] T. Ojala, M. Pietikainen, and T. Maenpaa, "Multiresolution gray-scale and rotation invariant texture classification with local binary patterns," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 7, pp. 971–987, 2002.

[120] S. Ongkittikul, S. Worrall, and A. Kondoz, "Two hand tracking using colour statistical model with the k-means embedded particle filter for hand gesture recognition," in *7th Computer Information Systems and Industrial Management Applications.* Ostrava, Czech Republic: IEEE, June 2008, pp. 201–206.

[121] T. M. Oshiro, P. S. Perez, and J. A. Baranauskas, "How many trees in a random forest?" in *Proceedings of the Eighth International Conference on Machine Learning and Data Mining in Pattern Recognition*, vol. 7376. Berlin, Germany: Springer, July 2012, pp. 154–168.

[122] V. F. Pamplona, L. A. Fernandes, J. L. Prauchner, L. P. Nedel, and M. M. Oliveira, "The image-based data glove," in *Proceedings of the 10th Symposium on Virtual and Augmented Reality*, Joao Pessoa, Brazil, May 2008, pp. 204–211.

[123] C. Park and S. Lee, "Real-time 3D pointing gesture recognition for mobile robots with cascade hmm and particle filter," *Image and Vision Computing*, vol. 29, no. 1, pp. 51–63, 2011.

[124] S. Park, S. Yu, J. Kim, S. Kim, and S. Lee, "3D hand tracking using kalman filter in depth space," *EURASIP Journal on Advances in Signal Processing*, vol. 2012, no. 1, pp. 1–18, 2012.

[125] N. Petersen and D. Stricker, "Fast hand detection using posture invariant constraints," in *KI 2009: Advances in Artificial Intelligence.* Paderborn, Germany: Springer, September 2009, vol. 5803, pp. 106–113.

[126] M. Pietikäinen, A. Hadid, G. Zhao, and T. Ahonen, *Computer vision using local binary patterns.* Springer, 2011, vol. 40.

[127] V. A. Prisacariu and I. Reid, "3D hand tracking for human computer interaction," *Image and Vision Computing*, vol. 30, no. 3, pp. 236–250, 2012.

[128] S. S. Rautaray and A. Agrawal, "A real time hand tracking system for interactive applications," *International Journal of Computer Applications*, vol. 18, no. 6, pp. 28–33, 2011.

[129] J. Relethford, "Human skin color diversity is highest in sub-Saharan African populations." *Human biology; An International Record of Research*, vol. 72, no. 5, pp. 773–780, October 2000.

[130] Z. Ren, J. Meng, and J. Yuan, "Depth camera based hand gesture recognition and its applications in human-computer-interaction," in *Proceedings of the Eighth International Conference on Information, Communications and Signal Processing.* Singapore: IEEE, December 2011, pp. 1–5.

[131] C. W. Reynolds, "Flocks, herds and schools: A distributed behavioral model," in *ACM SIGGRAPH Computer Graphics*, vol. 21, no. 4. ACM, 1987, pp. 25–34.

[132] J. D. Roode, "Implications for teaching of a process-based research framework for information systems," Orlando, Florida, December 1993.

[133] E. Rosten and T. Drummond, "Machine learning for high-speed corner detection," in *Ninth European Conference on Computer Vision.* Graz, Austria: Springer, May 2006, pp. 430–443.

[134] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, "ORB: An efficient alternative to SIFT or SURF," in *13th IEEE International Conference on Computer Vision.* Barcelona, Spain: IEEE, November 2011, pp. 2564–2571.

[135] D. Rybach, "Appearance-based features for automatic continuous sign language recognition," Master's Thesis, Human Language Technology and Pattern Recognition Group, RWTH Aachen University, Aachen, Germany, June 2006.

[136] A. Saxena, J. Schulte, and A. Ng, "Depth estimation using monocular and stereo cues," in *20th International Joint Conference on Artificial Intelligence*, Hyderabad, India, January 2007, pp. 2197–2203.

[137] A. Schmidt, M. Kraft, F. Michal, and Z. Domagala, "Comparative assessment of point feature detectors and descriptors in the context of robot navigation," *Journal of Automation, Mobile Robotics and Intelligent Systems*, vol. 7, no. 1, pp. 11–20, 2013.

[138] T. Senst, B. Unger, I. Keller, and T. Sikora, "Performance evaluation of feature detection for local optical flow tracking." in *Proceedings of the International Conference on Pattern Recognition Applications and Methods*, Vilamoura, Algarve, Portugal, February 2012, pp. 303–309.

[139] C. Shan, S. Gong, and P. W. McOwan, "Facial expression recognition based on local binary patterns: A comprehensive study," *Image and Vision Computing*, vol. 27, no. 6, pp. 803–816, 2009.

[140] J. Shi and C. Tomasi, "Good features to track," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition.* Seattle, Washington: IEEE, June 1994, pp. 593–600.

[141] M. Shin, K. Chang, and L. Tsap, "Does colorspace transformation make any difference on skin detection?" in *Proceedings of the Sixth IEEE Workshop on Applications of Computer Vision.* Orlando, FL, USA: IEEE Computer Society, December 2002, pp. 275–279.

[142] H. S. Smallman, M. St John, H. M. Oonk, and M. B. Cowen, "Information availability in 2D and 3D displays," *IEEE Computer Graphics and Applications*, vol. 21, no. 5, pp. 51–57, 2001.

[143] A. Sobral, “BGSLibrary: An OpenCV C++ background subtraction library,” in *IX Workshop de Viso Computacional*, Rio de Janeiro, Brazil, Jun 2013.

[144] V. Spruyt, A. Ledda, and S. Geerts, “Real-time multi-colourspace hand segmentation,” in *17th IEEE International Conference on Image Processing*. Hong Kong: IEEE, June 2010, pp. 3117–3120.

[145] C. Stauffer and W. E. L. Grimson, “Adaptive background mixture models for real-time tracking,” in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 2. Collins, CO, USA: IEEE, June 1999.

[146] B. Stenger, A. Thayananthan, P. Torr, and R. Cipolla, “Model-based hand tracking using a hierarchical bayesian filter,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 28, no. 9, pp. 1372–1384, July 2006.

[147] X. Suau, J. Ruiz-Hidalgo, and J. R. Casas, “Real-time head and hand tracking based on 2.5D data,” *IEEE Transactions on Multimedia*, vol. 14, no. 3, pp. 575–585, 2012.

[148] E. Sudderth, M. Mandel, W. Freeman, and A. Willsky, “Visual hand tracking using nonparametric belief propagation,” in *Conference on Computer Vision and Pattern Recognition Workshop*. IEEE, June 2004, pp. 189–189.

[149] W. Tan, C. Chan, P. Yogarajah, and J. Condell, “A fusion approach for efficient human skin detection,” *IEEE Transactions on Industrial Informatics*, vol. 8, no. 1, pp. 138–147, 2012.

[150] X. Tan and B. Triggs, “Enhanced local texture feature sets for face recognition under difficult lighting conditions,” *IEEE Transactions on Image Processing*, vol. 19, no. 6, pp. 1635–1650, 2010.

[151] A. Thangali and S. Sclaroff, “An alignment based similarity measure for hand detection in cluttered sign language video,” in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*. Miami, Florida, USA: IEEE, June 2009, pp. 89–96.

[152] S. Theodoridis and K. Koutroumbas, *Pattern recognition*, 2nd ed. Academic Press-Elsevier, 2003.

[153] M. Travers, *Qualitative research through case studies.* SAGE Publications, Ltd, 2001.

[154] A. Tzotsos, "A support vector machine approach for object based image analysis," *in Proceedings of the First International Conference on Object-Based Image Analysis*, July 2006.

[155] C. M. Utz, "Learning ensembles of Bayesian network structures using random forest techniques," Masters Thesis, School of Computer Science, University of Oklahoma, Oklohoma, USA, 2010.

[156] M. Van den Bergh and L. Van Gool, "Combining RGB and ToF cameras for real-time 3D hand gesture interaction," in *IEEE Workshop on Applications of Computer Vision.* Kona, Hawaii: IEEE, January 2011, pp. 66–72.

[157] V. Vezhnevets, V. Sazonov, and A. Andreeva, "A survey on pixel-based skin color detection techniques," in *Proceedings of the Graphicon*, vol. 85, no. 1, 2003, pp. 85–92.

[158] P. Viola and M. J. Jones, "Robust real-time face detection," *International Journal of Computer Vision*, vol. 57, no. 2, pp. 137–154, 2004.

[159] J. Vom Brocke and C. Buddendick, "Reusable conceptual models – requirements based on the design science research paradigm," in *Proceedings of First International Conference on Design Science Research in Information Systems and Technology*, Claremont, California, February 2006.

[160] J. Wang, P. Neskovic, and L. N. Cooper, "Context-based tracking of object features," in *Proceedings of the IEEE International Joint Conference on Neural Networks*, vol. 3. IEEE, July 2004, pp. 1775–1779.

[161] R. Y. Wang and J. Popović, "Real-time hand-tracking with a color glove," in *ACM Transactions on Graphics*, vol. 28, no. 3. ACM, 2009, p. 63.

[162] J. Wen and Y. Zhan, "Vision-based two hand detection and tracking," in *Proceedings of the Second International Conference on Interaction Sciences: Information Technology, Culture and Human.* Seoul, Korea: ACM, November 2009, pp. 1253–1258.

[163] L. Wen, Z. Cai, Z. Lei, D. Yi, and S. Z. Li, "Online spatio-temporal structural context learning for visual tracking," in *European Conference on Computer Vision.* Firenze, Italy: Springer, October 2012, pp. 716–729.

[164] J. Whitehill, "Automatic real-time facial expression recognition for signed language translation," Master's Thesis, Computer Science, University of the Western Cape, Cape Town, South Africa, May 2006.

[165] K. Wu, E. Otoo, and A. Shoshani, "Optimizing connected component labeling algorithms," in *Medical Imaging 2005: Image Processing*, vol. 5747. SPIE, April 2005, pp. 1965–1976.

[166] B. Xiao, X.-m. Xu, and Q.-P. Mai, "Real-time hand detection and tracking using LBP features," in *Proceedings of the Sixth International Conference on Advanced Data Mining and Applications.* Chongqing, China: Springer, November 2010, pp. 282–289.

[167] W. Xu and E.-J. Lee, "A new NUI method for hand tracking and gesture recognition based on user experience," *International Journal of Security and Its Applications*, vol. 7, no. 2, pp. 149–158, 2013.

[168] Y. Xu, X. Deng, and Y. Jia, "A multi-cue-based human body tracking system," *The 5th International Conference on Computer Vision Systems*, March 2007.

[169] B. Yang and S. Chen, "A comparative study on local binary pattern (LBP) based face recognition: LBP histogram versus LBP image," *Neurocomputing*, vol. 120, pp. 365–379, 2013.

[170] L. Yi, "KernTune: Self-tuning linux kernel performance using support vector machines," Master's Thesis, Computer Science, The University of the Western Cape, Cape Town, South Africa, November 2006.

[171] B. Zarit, B. Super, and F. Quek, "Comparison of five color models in skin pixel classification," in *Proceedings of the International Workshop on Recognition, Analysis, and Tracking of Faces and Gestures in Real-Time Systems.* Corfu, Greece: IEEE Computer Society, September 1999, pp. 58–63.

[172] W. Zhang, S. Shan, W. Gao, X. Chen, and H. Zhang, "Local gabor binary pattern histogram sequence (LGBPHS): A novel non-statistical model for face representation and recognition," in *Proceedings of the Tenth IEEE International Conference on Computer Vision*, vol. 1. Beijing, China: IEEE, October 2005, pp. 786–791.

[173] H. Zhou and T. S. Huang, "Tracking articulated hand motion with eigen dynamics analysis," in *Proceedings of the Ninth IEEE International Conference on Computer Vision*. Nice, France: IEEE, October 2003, pp. 1102–1109.

[174] Z. Zivkovic and F. van der Heijden, "Efficient adaptive density estimation per image pixel for the task of background subtraction," *Pattern recognition letters*, vol. 27, no. 7, pp. 773–780, 2006.

[175] J. A. Zondag, T. Gritti, and V. Jeanne, "Practical study on real-time hand detection," in *Proceedings of the Third International Conference on Affective Computing and Intelligent Interaction and Workshops*. Amsterdam, The Netherlands: IEEE, September 2009, pp. 1–8.