Gesture Recognition by Computer Vision

An Integral Approach

Proefschrift

ter verkrijging van de graad van doctor
aan de Technische Universiteit Delft,
op gezag van de Rector Magnificus prof. dr. ir. J. T. Fokkema,
voorzitter van het College voor Promoties,
in het openbaar te verdedigen op dinsdag 13 oktober 2009 om 15:00 uur
door Jeroen Frederik LICHTENAUER
elektrotechnisch ingenieur
egeboren te Den Haag.
Dit proefschrift is goedgekeurd door de promotor:
Prof. dr. ir. M.J.T. Reinders

Copromotor:
Dr. E.A. Hendriks

Samenstelling promotiecommissie:
Rector Magnificus, voorzitter
Prof. dr. ir. M.J.T. Reinders, Technische Universiteit Delft, promotor
Dr. E.A. Hendriks, Technische Universiteit Delft, copromotor
Prof. dr. ir. L.J. van Vliet, Technische Universiteit Delft
Prof. dr. I. Heynderickx, Technische Universiteit Delft
Prof. dr. J.J. Koenderink, Universiteit Utrecht
Prof. dr. E.O. Postma, Universiteit van Tilburg
Prof. dr. ir. Bart ter Haar Romenij, Technische Universiteit Eindhoven

De afbeelding op de omslag is een uitsnede van het werk ‘Handwandelung’, gemaakt door Alexandra Vallinga © 2006.
Het werk beschreven in dit proefschrift is uitgevoerd in samenwerking met de NSDSK en gedeeltelijk ondersteund door het VSB-fonds.

This work was carried out in the ASCI graduate school.
ASCI dissertation series number assigned series number 185.

ISBN: 978-90-813811-5-4
Uitgever: TU Delft Mediamatica

Copyright © 2009 by J. F. Lichtenauer
All rights reserved. No part of this thesis may be reproduced or transmitted in any form or by any means, electronic, mechanical, photocopying, any information storage or retrieval system, or otherwise, without written permission from the copyright owner.
Gesture Recognition by Computer Vision

An Integral Approach

Voor mijn ouders
Gesture Recognition by Computer Vision

An Integral Approach

Jeroen Frederik Lichtenaur
Summary

The fundamental objective of this Ph.D. thesis is to gain more insight into what is involved in the practical application of a computer vision system, when the conditions of use cannot be controlled completely. The basic assumption is that research on isolated aspects of computer vision often leads to ‘too’ general solutions. That these solutions lack the robustness and accuracy, which could only be achieved by an integral approach of a specific application. Furthermore, an integral approach, and actually trying out a computer vision system in practice, can lead to new insights that can determine the direction of future research in computer vision.

The application for the research in this thesis is automatic sign recognition for feedback in active learning with an electronic learning environment for sign language. The goal of this learning environment is to enlarge the vocabulary of deaf and hard of hearing children, between the age of 3 and 5, in order to facilitate in decreasing a delay in language development. The research has been focussed on a number of aspects that were assumed to have the most influence on the robustness of sign recognition. These were: tracking of movements, the extraction of relevant structure information from an image, skin color detection, including the third dimension of hand locations, dealing with variations of time as well as shape of a sign and reducing the required effort to teach the system to recognize a new sign.

‘Particle filtering’ is a popular method to track hand movement. However, tests with the CONDENSATION algorithm show contradictions in dealing with different situations. When the motion is unpredictable (as is the case with tracking of human hands) a particle filter has difficulty to keep track of the object. It turns out that, under different conditions, different strategies are required to deal with this in the best possible way.

Isophote properties can be used as local abstractions of an image. One advantage of isophote properties is that they are independent of image contrast. In experiments with face detection using isophote properties, the results are superior to using pixels, gradients, or the popular Haar features.

Because face detection requires significant computational cost, and the methods involved are less suitable for detection of hands, it is appealing to detect these body parts on the basis of their color alone. Unfortunately, color behaves less predictable in practical situations, than can be described by a single light-reflection model. Deviations from physical models for reflection are caused by properties and settings of the camera that is used, but also by the combination of different light sources and reflec-
tions. By combining these uncertainties in a more general model, robustness can be obtained in unknown circumstances. Unfortunately, this generalization comes at the price of accuracy in more friendly conditions. To combine robustness with accuracy, we have proposed an adaptive chromatic model, which can use a small set of measurements to model variation of the color of a face, using a bi-modal piecewise linear model in the red/green/blue space.

Sign language takes place in a three-dimensional space, while images only allow measurements in two dimensions. Therefore, we have used stereometry to convert the measured hand locations in the images from two cameras into three-dimensional positions of the hands in space. The experiments show that this richer information does indeed lead to an improvement in sign recognition. Alternatively, the perspective of a single wide-angle camera at a short distance turned out to achieve a comparable improvement. However, the disadvantage of the latter solution is a decreased robustness, because perspective depends highly on the location of a person relative to the camera.

Using dynamic recognition methods, like “Hidden Markov Models” (HMM) or Statistical “Dynamic Time Warping” (SDTW) a sequence of measured features of a person can be recognized as a specific sign. These models are able to deal with differences in tempo, contrary to conventional methods of pattern recognition, which can only deal with a fixed set of features. However, one of the disadvantages of HMM and SDTW is that they assume that what is important for estimating time warping, is equally important to class recognition. Furthermore, they are based on the factorization of probabilities for different time points, preventing the modeling of dependencies between measurements at different time steps. For these reasons, we have proposed to separate time warping and classification into subsequent processing steps. Experiments show a significant improvement over HMM or SDTW alone.

In practice, it is difficult to obtain many examples of signs from different persons, in order to train a recognition system. To make the system robust to small training sets, we let the system make use of sign classes that were already trained with many examples. Here, we assumed that, when a part of the new sign is very similar to a part of a learned sign, its variation can be modeled in the same way. With a single example as the training material, this generalizing system performed comparable to when five examples are used in the regular training method.

From this thesis, it can be concluded that robustness is not only relevant for practical applications of computer vision, but also deserves a place in fundamental research. Combining vantage points from different disciplines, such as physics, machine learning, neuropsychology and human computer interaction, makes sure that all aspects of a computer vision process can be integrally taken into account. With this, more robust solutions can be obtained than with each of the disciplines separately.

Summary of the thesis: “Gesture Recognition by Computer Vision: An Integral Approach”.

# Table of Contents

## Summary

<table>
<thead>
<tr>
<th>1 Introduction</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.1 The Computer Vision Paradigm</td>
<td>2</td>
</tr>
<tr>
<td>1.1.1 Physics</td>
<td>2</td>
</tr>
<tr>
<td>1.1.2 Machine Learning</td>
<td>2</td>
</tr>
<tr>
<td>1.1.3 Neuropsychology</td>
<td>2</td>
</tr>
<tr>
<td>1.1.4 Human Computer Interaction</td>
<td>3</td>
</tr>
<tr>
<td>1.1.5 Aims of This Thesis</td>
<td>3</td>
</tr>
<tr>
<td>1.2 The Application</td>
<td>4</td>
</tr>
<tr>
<td>1.2.1 Related work on Sign Language Recognition by Computer Vision</td>
<td>5</td>
</tr>
<tr>
<td>1.2.2 Discussion</td>
<td>6</td>
</tr>
<tr>
<td>1.3 Thesis Outline</td>
<td>7</td>
</tr>
<tr>
<td>Bibliography</td>
<td>9</td>
</tr>
</tbody>
</table>

## Influence of The Observation Likelihood Function on Particle Filtering Performance in Tracking Applications

| 2.1 Introduction | 12 |
| 2.2 Observation Model | 13 |
| 2.3 Experimental Results | 17 |
| 2.3.1 Single Object Tracking | 17 |
| 2.3.2 Multiple Object Tracking | 18 |
| 2.3.3 Finding a Lost Object | 19 |
| 2.3.4 Discussion | 19 |
| 2.4 Conclusions | 20 |
| Bibliography | 21 |
### 3 Isophote Properties as Features for Object Detection

3.1 Introduction ........................................... 24
3.2 Isophote Orientation and Curvature Features ................. 25
  3.2.1 Direct Isophote Properties .......................... 25
  3.2.2 Gradient Structure Tensor Properties ................. 27
  3.2.3 Orientation Features ................................ 28
3.3 Experimental Results .................................... 29
  3.3.1 Features ............................................ 29
  3.3.2 Data Sets .......................................... 30
  3.3.3 Classifiers ......................................... 30
  3.3.4 Discussion .......................................... 31
3.4 Conclusions and Future Work ................................ 32

**Bibliography** ........................................... 35

### 4 A Shadow Color Subspace Model for Imperfect, Not-Calibrated Cameras

4.1 Introduction ........................................... 38
4.2 Projection Confined Shadow Subspace ........................ 41
4.3 Experiments ............................................ 46
4.4 Conclusions ............................................ 51

**Bibliography** ........................................... 52

### 5 A Self-Calibrating Chrominance Model Applied To Skin Color Detection

5.1 Introduction ........................................... 54
5.2 Related Work .......................................... 55
5.3 Color Detection Method .................................. 55
  5.3.1 Skin Color Appearance Model ......................... 56
  5.3.2 Skin Likelihood ..................................... 58
  5.3.3 Automatic Model Fitting ............................. 59
5.4 Experiment ............................................ 61
5.5 Conclusions and Future Work ................................ 62

**Bibliography** ........................................... 64

### 6 3D versus 2D Pose Information for Recognition of NGT Signs

6.1 Introduction ........................................... 66
6.2 Related Work .......................................... 67
6.3 Sign Detection Method ................................... 67
  6.3.1 Pose Estimation .................................... 68
  6.3.2 Classification ...................................... 69
6.4 Experiments ............................................ 70
# Table of Contents

6.5 Conclusions .................................................. 73
Bibliography .................................................. 74

7 Sign Language Recognition by combining Statistical DTW and Independent Classification 75
7.1 Introduction .................................................. 76
7.2 Related Work .................................................. 78
7.3 Statistical Dynamic Time Warping ............................... 79
7.4 Classification .................................................. 81
   7.4.1 Discriminative Feature (DF) selection ...................... 81
   7.4.2 Combined Discriminative Feature Detectors (CDFD) ....... 81
   7.4.3 Quadratic Classification on DF Fisher Mapping (Q-DFFM) .... 83
7.5 Experiments .................................................. 83
   7.5.1 SDTW Outperforms HMM ................................... 85
   7.5.2 Transition Probabilities are Questionable ................... 86
   7.5.3 Hybrid Approach Best for Single-Model Target Classification 87
   7.5.4 Hybrid Model-Combining Requires Different Approach .... 92
   7.5.5 Multi-class Classification Not Suitable to Detect a Target-Class 93
7.6 Conclusions .................................................. 94
Bibliography .................................................. 95

8 Learning to Recognize a Sign from a Single Example 99
8.1 Introduction .................................................. 100
8.2 Related Work .................................................. 100
8.3 Classification Method ......................................... 101
   8.3.1 Dynamic Time Warping .................................... 102
   8.3.2 Combined Discriminative Feature Detectors ................. 104
8.4 Cross-Generalization Method .................................. 105
8.5 Experiments .................................................. 106
8.6 Conclusions .................................................. 110
Bibliography .................................................. 111

9 Discussion ..................................................... 113
9.1 Tracking ..................................................... 113
9.2 Color ......................................................... 114
9.3 Features ...................................................... 115
9.4 Recognition of Time-Variable Signals ......................... 115
9.5 Human Computer Interaction .................................. 115
A Supplemental Material for Chapter 7  

A.1 Introduction .............................................. 118  
A.2 Robustness CDFD ........................................ 118  
A.3 Dynamic Time Warping (DTW) .......................... 120  
A.4 System Setup ............................................. 121  
A.5 Image Analysis ........................................... 122  
   A.5.1 Single-Camera Tracking ............................. 122  
   A.5.2 Disparity Refinement ............................... 124  
A.6 Features ................................................. 125  
   A.6.1 Pre-Processing ...................................... 126  
   A.6.2 Hand Coordinate System ......................... 126  
   A.6.3 Time-Derivative .................................... 127  
   A.6.4 Hand Speed ........................................ 127  
   A.6.5 Motion Orientation ................................. 128  
   A.6.6 Curvature ........................................... 129  
   A.6.7 Hand Size .......................................... 130  
A.7 LIT Curvature Estimation .............................. 130  
The Bibliography ............................................. 132  
Acknowledgements ........................................... 139
Chapter 1

Introduction

It seems that if a computer could be given eye-sight, the number and flexibility of applications for computer vision would be endless. Therefore, we are currently very fortunate to have fast, inexpensive, high resolution digital cameras and object/face detection, tracking and recognition. But why then, do we still have to tell and prove to a computer who we are, each time we need to log in into the same system, while our friends and authorities can identify us in a glance? Why isn’t there a camera in each traffic light, so we don’t have to wait pointlessly in front of a red light when it is obvious nobody is there to cross the road anyway? And why do we still need human police patrol?

Currently, computer vision is most successfully applied to replace humans in productions lines. For instance, to inspect products for faults or to sort out products with different sizes, shapes or colors. In these applications, many variables can be controlled precisely. Lighting, camera properties, viewpoint, location and motion of the subject can all be estimated or determined within small ranges of uncertainty.

Compared to human vision, computer vision excels at precision, but lacks in robustness under more variable conditions. The difficulty computers have with robustness is mainly due to the poor generalizing performance of current computer vision methods. The complexity of vision, compared to other areas of pattern recognition, lies in the combination of the high dimensionality of visual data (when each pixel is considered an individual measurement), with the high variability of objects and lighting. Some methods work well under the constraints for which they are designed and trained, but they lack the ability to maintain their performance under any other condition. For vision, the space of these ‘other conditions’ is, however, vast, and one definitely will encounter those in practical applications.

The objective of the research that is described in this PhD thesis was to increase robustness of computer vision in the area of automatic sign language recognition.
1.1 The Computer Vision Paradigm

Computer vision is approached from many different disciplines, such as physics, machine learning, neuropsychology or human computer interaction. Although each of these vantage points is aimed at the same result (providing computers with knowledge about visible aspects of their environment), the way to reach it can be completely different.

1.1.1 Physics

A physics approach tries to either have control over circumstances, or to obtain a model for the processes that underly a measured image. The imaging process can be controlled by choosing the most appropriate illumination (intensity, color, orientation, etc.) and camera optics (spectral sensitivity, lens distortion, location, etc.). Variables that cannot be controlled, may be estimated by fitting a parametric model. A complete physical model (of lighting, object and/or capture process) can greatly reduce the dimensionality of the visual information. However, if a model is imprecise or incomplete, it may do more harm than good, since it may not be able to describe actual data. With the large number of variables and complexity of the processes that can underly image data, physically modeling the whole process is (currently) infeasible for many applications.

1.1.2 Machine Learning

Machine learning intends to retrieve as much information as possible, out of the available data. If little is known about the underlying physical processes, machine learning algorithms can automatically retrieve structure from the data. This is very useful for computer vision, since image data is often the combined result of a large number of different physical aspects, too complicated or uncertain to be captured completely within physical models. The aim of machine learning research is, usually, to work with as little a priori knowledge as possible, for as many different scenarios as possible (generality). By having to obtain all knowledge about the underlying structure from the training data, machine learning methods often require a large number of representative examples (possibly including detailed annotations). For many applications, a sufficiently large amount of data can simply not be obtained (within reasonable time or cost).

1.1.3 Neuropsychology

A neuropsychological approach investigates how biology solves the computer vision problem in animals such as cats, primates or humans, with the intention to mimic these processes in computer algorithms. The first step of animal vision (within the retina) can be approximated reasonably well by straightforward (input-output) signal processing methods. However, neurological research has found at least five distinct areas of further processing in the visual cortex of the primate brain (V1 to V5) [1].
1.1. The Computer Vision Paradigm

Between these areas, information does not simply go one way. Feedback paths between the different levels indicate that animal vision is not simply a passive process of classifying visual patterns into objects. On the contrary, ‘seeing’ may prove to be a highly active process. What animals see, may, for a large part, be determined by their high-level understanding of the physical world, combined with the expectations they have about what can be seen in a specific situation, given the complete context. If this hypothesis is true, the performance of human vision may only be approximated when it is backed up by a similar richness of knowledge about the world.

At the same time, focussing only on the visual capabilities of animals, prevents to make use of other visual technologies, that may exceed animal performance in some respects. For instance, most computer vision research is based on cameras that capture the same three color bands for which the human eye is sensitive (‘red’, ‘green’ and ‘blue’). However, technology allows us to measure color at a much higher spectral resolution, or to measure a spectral range far into the ‘invisible’ infrared or ultraviolet. Furthermore, the depth perception of humans is very imprecise, while computers can calculate depth accurately, using triangulation computations or time-of-flight measurements. These technologies provide a richness of information that may reduce complexity, compared to the processes in the primate brain, that need to translate shading and motion into information about object shape.

1.1.4 Human Computer Interaction

A Human Computer Interaction (HCI) vantage point may focus on how to use visual recognition and detection results within a (software) application. Visual information may be combined with other modalities such as audio, buttons and mouse input. The goal of HCI is to achieve maximal ‘usability’. For HCI, computer vision is ‘only’ one of the (many) components to achieve this goal. HCI is able to take into account capabilities and shortcomings of vision algorithms. However, computer vision algorithms may be more general than necessary, while not being robust enough to be useful for a practical application. Instead, highly specialized algorithms may provide enough robustness for a well-bounded application. If more constraints can be set on what does not have to be solved by computer vision, more effort can be spent on obtaining robustness in the actual required functionality. Therefore, computer vision research may be more fruitful, when it is aimed at a specific application.

1.1.5 Aims of This Thesis

Each approach mentioned above misses information and/or opportunities that can be offered by the others. Combining different approaches to gain further improvement, only after each single approach has been completely perfected, is the greedy approach to computer vision. Then, we may end up with perfectly developed, single-disciplinary approaches that are very fragile in many (possibly unforeseen) practical situations, without allowing straightforward combining with other approaches.

Instead, we can choose to combine all disciplines, and find much more robust computer vision solutions for specific applications. With a specific application in mind, the overview from a HCI approach can be used to place computer vision in a
1. Introduction

realistic setting: not using computer vision to solve more problems than necessary. Some problems can be solved much more efficiently by other modalities (buttons, sound, user feedback, etc.). Furthermore, machine learning can be used to only fill the gaps that cannot be covered by physical control or modeling. This could lead to high robustness in the case of uncertainty, at a relatively low training cost. Methodologies found in human visual processing may be combined with technology that is not used by humans, such as range data, active imaging (illumination by light of certain structure and/or spectrum) or a spectral sensitivity with higher resolution and/or broader/different range.

Placing robustness above generality does not necessarily lead us away from generality forever. On the contrary. We may gain tools, knowledge, insights and inspiration from the success of highly specialized solutions, the development and use of diverse hardware, or the experiences provided by the practical use of computer vision. This may help us to more effectively conduct and focus further research, eventually leading to a faster increase of generality, without compromising robustness.

In this scope, our approach was to focus on completing a robust, usable prototype system for a specific application of human behavior analysis. Applying as many of the above mentioned perspectives as possible.

1.2 The Application

Sign languages are the primary means of communication in communities of deaf and hard of hearing people. Although there are many different sign languages, they mostly use the same components: Hand shape, -orientation, -location and -movement, body pose and facial expression. Signs of a sign language can be seen as the specific gestures that are the language's building blocks. Comparable to words in a spoken language.

A child without the ability to hear, in a family without a fluent signer (a person that masters a sign language), does not receive an amount and/or quality of language input that is comparable to a hearing child in a speaking family, or a deaf child in a family with native signers. Children who learn their first language late in life run the risk of suffering the effects of language delay. Therefore, this is a serious risk for many deaf children [2]. Although this problem has been reduced by the use of Cochlear Implants (CI), a CI is not applicable or equally successful in all cases. Therefore, language delay in deaf and hard of hearing children remains a problem to be solved.

An electronic learning environment for deaf children could be used as an extra support to overcome a delayed language acquisition process, in addition to the interaction with teachers, parents and peers. Literature about the importance of active participation in learning is abundant [3, 4, 5, 6]. Therefore, it seems obvious that any sign language learning process should motivate the children to produce the signs themselves. Without the possibility of feedback on sign production, an electronic learning environment would lack an essential learning factor. The first crucial step towards feedback is to sense the child’s sign. Sensors that have to be attached to a child’s hands could compromise usability and the small and various sizes of the hands and fingers of children would make it difficult to use gloves. If the attachment of the
1.2. The Application

sensors would have to involve the guidance of an adult, this would also mean that a child cannot initiate learning on his or her own. Though we cannot be certain what the exact psychological and practical consequences would be, it would be highly preferable if such obtrusions can be avoided completely. Therefore, computer vision is an obvious choice.

1.2.1 Related work on Sign Language Recognition by Computer Vision

Most of the sign language recognition methods proposed in literature have focused on only one or a few aspects at a time, leaving important practical issues unsolved; such as the influence of clothing and background color, non-static or non-uniform backgrounds, inter-personal variation, time-segmentation, large vocabularies, tracking of fast signs or quick response time. Complete systems that solve sign language recognition on all levels, for a general setting, are still absent. There have been a few systems that provide sign language recognition by computer vision with some limited practical functionality. These will be described below.

In 1996, Thad Starner was one of the first to come up with a real-time sign language recognition system [7] that doesn’t require wearing any gloves. The skin color based hand tracking works at 10 frames per second on a 200MHz computer. Sign recognition is done with 4-state Hidden Markov Models (HMM) with a single skip. This low number of states reflects the low resolution in time. The problems of body and face localization, and occlusion of the face by the hands, are avoided by placing the camera looking downwards in the cap of a hat, worn on the signer’s head. The results for this setup are shown to be superior to a frontal view, desk setup. The tracked hand motions only contain 2D image coordinates, missing motions in the direction perpendicular to the image plane. The shapes of the 2D hand blobs are described by moments. Touching or overlapping of the hands is dealt with by replacing the two separate hand shape descriptions with that of their combined skin blob. Using a vocabulary of 40 words, recognition accuracy of separate words is 97.6%. The words are recognized continuously within sentences without the need for word-level segmentation. However, the experiment does not include rejection of silences, unknown signs or fidgets. Furthermore, the experiments consist of training and testing with the same person. It is currently a well-known fact that inter-personal sign variation has a significant negative effect on sign language recognition accuracy [8].

In 2005, Zieren and Kraiss [9] published about a sign language recognition system, based on HMM, that obtains more robust tracking of the hands by considering multiple hypotheses of probable paths. The tracking combines skin segmentation, background modeling, a bio-mechanical body model and template tracking. The drawback of this approach is that the tracking has to be done off-line, after the sign is made. The post-processing takes up to 11 seconds on a 2GHz PC. We expect that this is too long for giving feedback to children. Off-line hand tracking also means that automatic detection of the ending of the sign is not possible. Furthermore, the person-independent recognition rate is only 44.1% for a vocabulary of 221 signs, and 87.8% for a vocabulary of 18 signs.
CopyCat, developed by Brashear et al. [10], is designed as a game for teaching sign language to children of 6 to 11 years old. CopyCat classifies words or sentences as *correct or incorrect*. Recognition relies on the 2D tracking of pink gloves, combined with 3D acceleration measurements from the wireless accelerometers, attached to both gloves. A HMM is trained for each sign. Onset and offset of signing is indicated by the user clicking the mouse button before the start and after the end of signing a sentence. The data set for evaluating the sign recognition accuracy consists of actual signs made during participation in a Wizard of Oz (WOz) version of the game, by children between 9 and 11 years old. The authors report a 86.28% accuracy on signer independent word recognition over all correctly performed sentences, selected manually from all exercises. However, in practice, the incorrect signs that are performed in a specific exercise, depend highly on the exercise itself. It can be expected that a person will be much more likely to confuse very similar signs with the correct sign, than to mistakenly perform totally different signs. Therefore, reported recognition results do not reflect the actual performance during use.

Another Sign Language Tutoring Tool was presented by Aran et al. [11]. Gloves of different colors are used (blue for right- and yellow for left hand) to detect and segment both hands separately. The color models for the gloves are calibrated from samples obtained by manually segmenting the gloves in a number of frames. After this labor-intensive calibration, illumination cannot be changed significantly. Onset and offset of signing is estimated by detecting when the hands enter and leave the camera view, respectively. Therefore, the signer has to let both arms hang down while not signing. The subjects in the experiments are all standing. Besides 2D hand locations and blob descriptions, the authors also extract horizontal, vertical and absolute face movement components from optical flow. They use this to distinguish signs with the same manual component but different head movements. HMM’s of the manual components are used for classification of the set of distinct manual signs; HMM’s of the facial features are used to distinguish between the variants of signs with the same manual form but different head motions. 97.8% accuracy is reported on classifying a sign as 1 out of 19 signs. However, it is not mentioned if the signs in the training- and test sets are from different persons. Interestingly, a sign synthesis method is proposed as well, which converts the measured features into a animated version of the produced sign. This is meant as more detailed feedback. Unfortunately, no details are given on practical user experiences, let alone about the learning aspects of the application.

1.2.2 Discussion

Our primary objective is to achieve robust, real-time sign language recognition, with minimal obtrusion to the user. This excludes wearing gloves, such as in [10] and [11], or a head-mounted camera, such as in [7]. Evidently, person-independent recognition is essential, since the recognition system cannot be trained by signs of the children that still have to learn the signs themselves. Furthermore, recognition should be able to reject unknown and incorrect signs, as well as fidgeting.

The sign language recognition system of Starner [7] did recognize sentences of American Sign Language (ASL) in real-time, without the use of gloves, and even completely without wearables, if used in the frontal-view setup. However, the system
is only tested on the recognition of complete sentences, on the same person on which it was trained. It is doubtful if its accuracy would be sufficient for single-word person-independent recognition. Especially, considering the low frame rate, the 4-state word models and the lack of the third dimension of motion. Furthermore, if the camera is not worn on the person’s head, the problems of overlap of hands and head will have to be dealt with.

1.3 Thesis Outline

Different perspectives on computer vision can be combined, to improve the robustness of a recognition system. The different chapters of this thesis focus on separate aspects that are considered the most critical elements of gesture recognition. Although the aspects differ, they share the relevance of combining different perspectives, as a common theme. Chain 2 starts with an evaluation of object tracking with particle filtering, by controlled experiments with the well-known CONDENSATION algorithm. Particle filtering could be used for gesture recognition by improving the tracking of hands, face or complete body-pose. CONDENSATION is based on the principle of Kalman filtering, where state predictions are made according to a state transition model. CONDENSATION is a pure physics approach. The innovative part of the behavior that cannot be modeled by the physical model is modeled as an independent random variable. In this chapter, we show what can happen under difficult conditions, with a finite set of particles. The experimental results show that the optimal observation likelihood function depends on the situation. This indicates that the context-independent modeling in particle filtering could use more awareness of what is actually going on.

In chapter 3, isophote properties are considered as descriptors of low-level image structure. These properties are functions of Gaussian derivatives, similar to what is also done in the human visual system. In this chapter, we combine these neuropsychological inspired features with machine learning. Applied to face detection, the isophote properties are shown to be superior to pixels, gradients or haar-like features.

A physical model for background color variation is presented in chapter 4. Background modeling could be used to separate a signer in front of the camera from the background. The new model incorporates uncertainty about white-balance or black-point calibration of the camera. These situations are not accounted for in the chrominance models that are commonly used for background modeling. The proposed model may improve results in cases where little is known about the camera properties/characteristics/settings. However, if more can be known about the camera, accuracy could be improved by constraining the model. An automatic approach to model constraintment could be to learn the camera’s physical behavior from the data. That is, to combine a physical approach with a machine learning approach.

This is exactly what is done in chapter 5, where a method is proposed to model skin color based on semi-automatic chrominance-space calibration. The model can achieve a tight skin color classification, when possible, without white-balance or black-point calibration. It also takes into account uncertainty about continuation of the expected color variation for lower and higher intensities of skin color than seen in the calibration.
samples.

In chapter 6, the advantage of depth from stereo in sign language recognition is evaluated. Not surprising, the results show that the additional third spatial dimension of hand location and motion indeed improves performance. However, the circular view of a wide-angle camera at close range also provides implicit depth information in the 2D image coordinates. It is shown that this can also result in improved recognition performance compared to an orthographic image projection (infinite camera distance). From an HCI perspective, this may be exploited in situations where only a single camera can be used.

Chapter 7 proposes a hybrid approach in the use of Hidden Markov Models (HMM). Instead of relying on HMM likelihoods for classification, a more flexible variant, Statistical Dynamic Time Warping (SDTW), is used only to synchronize gestures with the target model. Classification of the fixed-length synchronized feature set is done with two different novel target-class discriminants. The improvements in performance over HMM’s are significant. The hybrid approach is based on the assumption that, although prior information may be used to find the most likely time warp, the ‘likeliness’ of the found warping may not be directly related to the likelihood of the signal class. Furthermore, the properties that can help in finding the time warping may not be equally useful to distinguish between different classes. This departs from a pure Bayesian machine learning approach, where the posterior likelihood is the combined likelihood of all measurable aspects, to a more physical approach, acknowledging that time warping may be the result of processes that are unrelated to signal class. For example, dialect, emotion, distraction, etc.

Pushing the approach proposed in chapter 7 to the limit of small training set size, chapter 8 attempts to use knowledge of sign classes, learned with many examples, to construct a classifier for an unseen sign from only a single example. ‘Cross-generalization’ is inspired by the way how humans generalize knowledge, by associating new problems with familiar problems that are already understood. The results from Dynamic Time Warping (which uses a single example by definition) are exceeded by the proposed cross-generalization method. This shows that cross-generalization is a promising approach to extending the vocabulary of sign language recognition without the need for the construction of large training sets.

In chapter 9, we discuss the insights about different aspects of computer vision, that have been gained.
Bibliography


1. Introduction
Chapter 2

Influence of The Observation Likelihood Function on Particle Filtering Performance in Tracking Applications

©2004 IEEE. Personal use of this material is permitted. However, permission to reprint/republish this material for advertising or promotional purposes or for creating new collective works for resale or redistribution to servers or lists, or to reuse any copyrighted component of this work in other works, must be obtained from the IEEE.

This chapter is based on the article published as "Influence of The Observation Likelihood Function on Particle Filtering Performance in Tracking Applications", by J. F. Lichtenauer, M. J. T. Reinders and E. A. Hendriks in the Proc. of the IEEE Conf. on Automatic Face and Gesture Recognition, pages 767 - 772, Seoul, Korea, May 2004.
Abstract Since the introduction of particle filtering for object tracking, a lot of improvements have been suggested. However, the definition of the observation likelihood function, needed for determining the particle weights, has received little attention. Because particle weights determine how the particles are re-sampled, the likelihood function has a strong influence on the tracking performance. We show experimental results for three different tracking tasks for different parameter values of the assumed observation model. The results show a large influence of the model parameters on the tracking performance. Optimizing the likelihood function can give significant tracking improvement. Different optimal parameter settings are observed for the three different tracking tasks. Consequently, when performing multiple tasks a trade-off must be made for the parameter setting. In practical situations where robust tracking must be achieved with a limited amount of particles, the true observation probability is not always the optimal likelihood function.

2.1 Introduction

Visual object tracking has a very broad area of applications. It can be used for instance in automated surveillance or gesture recognition systems. In these applications speed is often a very important factor. Recently, the rapid increase of computer power has facilitated more practical use of particle filtering, also known as sequential importance (re-)sampling (SIS/SIR) or sequential Monte Carlo (SMC) methods [1, 2, 3, 4]. These methods produce much better tracking performance than, previously often used, Kalman filtering.

In SIR, a weighted set of hypothesized samples of the possible object state, called ‘particles’, are tracked simultaneously. At time $t$ this set consists of $N$ object states $x^{(1)}_t, \ldots, x^{(N)}_t$ and their associated weights $\pi^{(1)}_t, \ldots, \pi^{(N)}_t$. The particle set is a discrete approximation of the posterior distribution of the real object state given the observations up to time $t$:

$$p(x_t | z_{0:t})$$

At the next time step, the particles are re-sampled according to their weights. This is to decrease the number of low-weighted particles and to increase the ones with more ‘potential’.

For SIR to be successful a large number of samples $N$ is needed for two reasons: 1) to get a good approximation of $p(x_t | z_{0:t})$ and 2) to be able to recover from object loss and to find multiple instances if more than one object is visible. However, the size of $N$ has a direct relation with the computational cost and should be kept as low as possible. To increase the efficiency of particle filtering for small $N$, many improvements have been suggested. For instance, hierarchical methods [5, 6] where a coarse to fine approach is used to find the real mode(s) of the object(s) without getting stuck in local optima. Other methods involve more sophisticated re-sampling and/or prediction [7, 8]. However, not much research has been performed on improving the observation model, or likelihood function $p(z|x)$, which is necessary to calculate the particle weights from the observed image. Often no details are given on how the observation model of a particle filtering algorithm is determined. In [9] an optimization method is proposed that is used to optimize the likelihood function with respect to the effective sample size (ESS) and the mean square error (MSE) of the estimated object state. In [10] the observation model parameters are estimated from observations by simplified
2.2 Observation Model

The observation model that we assume here is given by:

\[ p(z|x) \propto \varepsilon + \exp \left( -\frac{[\bar{\alpha}]^2}{2\sigma_o^2} \right), \varepsilon \ll 1 \]  

(2.1)

where \( \varepsilon \) and \( \sigma_o \) are the model parameters and \( \bar{\alpha} \) is the distance measure between the proposed state \( x \) and the observation image \( z \), which will be explained below. The model is shown in fig. 2.1. The near-horizontal part of the model close to zero (\( R_1 \) in the figure) introduces some robustness for differences between the object model and the image due to distortion. Consequently, it prevents large differences between weights of particles close to a true state. The large drop of \( p(z|x) \) between low and high values of (\( R_2 \) in the figure) causes bad samples to be discarded when good samples are present. At high the horizontal part (\( R_3 \) in the figure), caused by \( \varepsilon \), makes approximate maximum likelihood (AML) estimation, which is explained in [11].

The likelihood function determines the re-sampling behavior of the particle filter algorithm and is expected to have influence on the tracking performance. Therefore, in this paper, we present experimental results providing more insight into the influence of the likelihood function on the tracking performance. We distinguish between three tracking tasks: 1) the ability to track a single object moving unpredictably, 2) the ability to track multiple objects simultaneously, and 3) the ability to retrieve the object after tracking loss. We do not regard tracking precision because in visual tracking it is often more important not to lose the object than to have a precise state estimate. For this evaluation we use the CONDENSATION algorithm [3] because it is widely used for comparison and its performance is primarily based on efficient re-sampling. In the supplementary information [12] we have additional information about the experimental setup as well as additional experiments.

Figure 2.1: The assumed observation model. A normal distribution with standard deviation \( \sigma_o \) plus a constant value \( \varepsilon \).
that truly bad samples are weighted equally. This prevents that samples are trapped in local maxima of background clutter when the object is lost. To improve the ability to recover from a lost object, these probabilities should be small but non-zero.

In particle filter algorithms, often edge distances have been used as distance measure [3, 4, 5, 6], which is based on the principle of Chamfer Matching (CM) [7, 13]. Also a combination of edge distance and orientations has been used [14]. However, in our experiments, we use only gradient directions for the distance measure between a hypothesized state and the observation. In this Gradient Direction Matching (GDM) method local directions of maximal gradient at points of the object are compared to the corresponding points in the scene. It is similar to the method applied in [15] where it is used for face detection. The difference is that the authors of [15] also threshold on edge strength to exclude unreliable directions. Using GDM has the advantages that 1) no edge detection and distance transforms need to be performed, that 2) the gradient direction is invariant to contrast and that 3) clutter will have less influence. In cluttered areas, i.e. areas with many (wrong) edges, the edge distance measure is likely to cause strong false matches. In cluttered environments the gradient directions are not more likely to resemble the objects directions than in uniform backgrounds. On the other hand, the disadvantage of GDM is that even uniform backgrounds have noisy gradient directions due to measurement noise, which can accidentally cause false matches even in (near) uniform backgrounds.

Let us define direction $\theta$ as the normal to the local gradient direction, given by:

$$
\theta = \arctan \left( \frac{\partial I(u, v)}{\partial u} / \frac{\partial I(u, v)}{\partial v} \right)
$$

(2.2)

where $I(u, v)$ is the intensity of the image $I$ at location $(u, v)$. The angular distance $\alpha_m(x, z)$ between the angle $\theta_m(x)$ of a point $m$ on the object with state $x$ and the corresponding point in the image $\theta_m(z)$ is computed by

$$
\alpha_m = \min \left( |\theta_m(z) - \theta_m(x)|, \pi - |\theta_m(z) - \theta_m(x)| \right)
$$

(2.3)

Note that $0 \leq \alpha_m \leq \pi/2$, i.e. the sign of the gradients are not used. This allows the background to be darker or to be lighter than the object, which is a useful property for robust tracking of contour edges. However, in cases when there are edges in the object where the sign of the gradient does not change, it would be better to exploit the gradient sign for these edges.

To compute the distance between the object and the scene, the mean of the angular distances $\alpha_{1:M}(x, z)$ is computed:

$$
\bar{\alpha} = \frac{1}{M} \sum_{m=1}^{M} \alpha_m(x, z)
$$

(2.4)

where $M$ is the number of object points that are used. To show the strength of GDM, detection results under severe clutter are shown in fig. 2.2, where the results for CM are also shown for comparison. These are the results of detecting a hand in front of an artificially generated cluttered background, see fig. 2.3. The model of the hand consists of 71 points on the contour of the hand together with the local contour directions
2.2. Observation Model

Figure 2.2: Comparison between Chamfer matching and gradient direction matching. Results are plotted for measurements along a horizontal line of fig. 2.3 where the hand position is located at 500 pixels. This is done using different standard deviations $\sigma_d$ of the Gaussian derivative filters. (a) and (b) show the results of Chamfer matching and Gradient direction matching, respectively, for detecting a hand in an image with random black and white background pixels (see fig. 2.3a). (c) and (d) show the results for a random black and white structured background (see fig. 2.3d).
2. Influence of The Observation Likelihood Function on Particle Filtering Performance in Tracking Applications

Figure 2.3: Hand in front of cluttered backgrounds, left half of the images. In (a) the background contains random black and white pixels. In (d) it consists of random black and white structure. (b) and (e) show the respective results of canny edge detection. (c) and (f) show the absolute gradients directions. These results are obtained with $\sigma_d = 2$.

[12]. The local gradient directions are calculated by using Gaussian derivative filters with standard deviation $\sigma_d$ [12]. In fig. 2.2 the measured distances are plotted for different standard deviations $\sigma_d$ of the derivative filters and different positions along a horizontal line in the corresponding images fig. 2.3a and b, where the correct hand position is at 500 pixels. To have good detection capacity, the distance at 500 pixels must be distinctively smaller than anywhere else.

When a background with random black and white pixels is used (fig. 2.3a-c), the numerous edges due to the background noise (see fig. 2.3b) cause a small distance for CM in the background for small $\sigma_d$, making the object undetectable. GDM is less influenced by the noise at small $\sigma_d$. In fact, it is affected by the noise approximately the same at all scales. At larger scales, the background is smoothed by the Gaussian derivative filter, eventually leaving the edges around the hand as the only ones, which makes the hand very well detectable by CM. However, the hand shape is severely distorted by the large-scale filters, causing the distance for GDM to increase.

The experiment is repeated with a background of larger random black and white segments [12], see fig. 2.3d-f. A small amount of random noise is also added to the background to prevent the segments to be completely uniform. Now both distance measures perform well at small scales. It can be seen that GDM produces very consistent results for the background measurements. They are always centered about $0.25\pi$ with approximately the same variance. CM, on the other hand, results in highly fluctuating values. This makes it harder to model the observation probability for CM than for GDM.
2.3. Experimental Results

![Figure 2.4](image)

Figure 2.4: Tracked object (a) and its edges (b) detected by Canny edge detection with $\sigma_d = 4$. At 76 selected edge locations (c). The directions are determined with a derivative filter with $\sigma_d = 1$. The background (512x512 pixels) consists of random black and white pixels as in fig. 2.3a.

2.3 Experimental Results

In the CONDENSATION algorithm, we used a constant velocity motion model, given by

$$x_t = 2x_{t-1} - x_{t-2} + n$$

where $x$ is a vector of the horizontal and vertical coordinates of the object center and $n$ is a vector of independent normally distributed random variables with $\sigma_n = 5$ pixels.

The tracked object is the head of Lena at 150x150 pixels, as shown in figure 2.4. The background consists of 512x512 random black and white pixels as in fig. 2.4a and is static during each experiment. The object model consists of a collection of $M=76$ edge point locations with corresponding values for $\theta_m$. Using only directions at edges of the object increases robustness against noise. Chen et al. [16] have shown that - provided there are no cast shadows or specular reflections - the direction of the gradient is insensitive to illumination direction and the robustness is higher at surfaces with high curvature.

2.3.1 Single Object Tracking

For single object tracking, all particles should stay close to the object state. When this object state corresponds to the minimum value for $\alpha$, the best tracking will be achieved if only the best matching sample is re-sampled. For our observation model, this will happen when $\sigma_o$ is close to zero and $\varepsilon$ equals zero, because then strong exponential behavior causes the largest weight to be significantly larger than all others. This is evaluated by tracking a single object following a circular trajectory with a radius of 150 pixels. In this case, $N = 100$ samples are used.

Because the motion model does not correctly predict circular motion, tracking relies on re-sampling. To test the influence of the likelihood function on the re-sampling, the speed of the object is gradually increased until the object is lost. Object loss is detected when the position of the best matching sample is not within a radius of 10 pixels.
2. Influence of The Observation Likelihood Function on Particle Filtering Performance in Tracking Applications

Figure 2.5: Tracking performance for different values of $\sigma$ and $\epsilon$. The results for tracking a single moving object (a), tracking two static objects simultaneously (b) and finding a lost object (c). All results are averaged over 50 experiments.

...around the actual object position. The average speed (50 experiments) at which the object is lost is plotted for different values of $\sigma_o$ and $\epsilon$ in Fig. 5a. When $\sigma_o$ is too large bad matches are not weighted low enough and will also be re-sampled, which causes an easy drift from the actual object state. Consequently, the most important location is sampled by less particles. This behavior can be seen in Fig 5a as the decrease for large values of $\sigma_o$. The same happens when $\epsilon$ is too large. In that case, the probability will level out too soon, causing (near) equal weights for bad and good matches. However, the point at which this happens depends on $\sigma_o$, resulting in a change of position of the steep slope at the left side as a function of $\sigma_o$. Finally, the experiments indeed show that the best tracking performance is achieved with $\epsilon = 0$ and $\sigma_o$ is small, see [12].

2.3.2 Multiple Object Tracking

If more then one object needs to be tracked, the samples must be re-sampled around all separate objects. Since there is a ‘competition’ between the objects, there is a non-zero probability that at a certain time $t$ no samples are re-sampled near one of the objects. In that case we consider the object being lost. It can only be retrieved if a sample around one of the other objects accidentally hits the lost object.

To test the influence of the observation model on tracking multiple objects, two objects are placed 300 pixels apart at fixed positions. At initialization, 50 samples are sampled around each object position. The tracking algorithm is run until there are no samples left within a radius of 10 pixels around one of the objects. Fig. 2.5b shows the results.

To reduce the possibility of losing an object, the weights of samples around all objects must be approximately equal. When $\sigma_o$ is too small, the differences between the weights will be very large, causing re-sampling of only the very best matching sample(s). This increases the possibility of losing one of the objects, which can be observed in fig. 2.5b. The time at which one object is lost decreases for smaller $\sigma_o$. The decrease for larger $\sigma_o$ and very large $\epsilon$ is again due to the drifting of samples.
2.3. Experimental Results

2.3.3 Finding a Lost Object

When the object is lost completely, it is desirable to recover as quickly as possible. In a tracker that only uses condensation this can only happen if the samples spread out across the entire state space until the object is found. To lock on the object again, the sample that has found it must be re-sampled as much as possible.

The performance of finding a lost object is evaluated by initializing samples 300 pixels away from the object position, which is fixed during the search. In this experiment \( N = 500 \) samples were used. The tracking algorithm is run until the particles have locked on to the object or 25 time steps have passed without an object lock. An object lock is detected when the best matching sample is within a radius of 5 pixels from the object during more than 5 subsequent time steps to be sure the ‘hit’ is not just a free moving particle passing by. Samples that leave the image area are re-initialized with velocity 0 at uniform distributed random locations in the image.

The average runtimes are shown in fig 2.5c. Two ridges, indicating optimal values, can be seen. One is parallel to the \( \epsilon \) axis. The other, sharper and higher, is curved and runs from low \( \sigma_o \) and \( \epsilon \) to high \( \sigma_o \) and \( \epsilon \). The first one corresponds to an observation model that is nearly Gaussian, i.e. the contribution of \( \epsilon \) is insignificant. If \( \sigma_o \) of the observation model is too small, the particles get stuck in local maxima of the background and do not propagate across the entire space. If \( \sigma_o \) is too large, the particles will not lock on to the object when it is found. This trade-off determines the local optimum for \( \sigma_o \). When \( \epsilon \) is too large, the particles drift anyway. The second ridge corresponds to the combinations of \( \sigma_o \) and \( \epsilon \) that level out the observation model at the best point. It should be noted that when all matches are on the flat part of the curve, the samples have near equal weights and propagate freely. When one or more samples get a sufficiently good match these samples get a much higher weight, causing all samples to be re-sampled near the object within the next few time instances. For the best results, the observation probability is equal for all samples in the background, to prevent getting trapped in local maxima, and much higher for samples near the object.

2.3.4 Discussion

The results show that choosing the parameters of the observation likelihood function determines the performance of tracking to a large extend. Consequently, optimizing the likelihood function can give significant improvement. Furthermore, the three tracking tasks show different optima for the likelihood function parameters. This implies that generally a trade-off has to be made that depends on the desired tracking tasks. This is different from the theory of particle filtering where the likelihood function is defined as the true observation probability. The reason for this is the limited amount of particles that are used in practical tracking applications. The true posterior probability distribution \( p(\mathbf{x}_t|z_{0:t}) \) cannot be reconstructed completely by the finite particle set. Consequently, a choice has to be made where to focus the attention on. In these experiments we have used a pre-defined \( \sigma_n \). However, this parameter also has influence on tracking performance. The influence of \( \sigma_n \) on the optimal likelihood function remains for future research.
Although these experiments were conducted using CONDENSATION, we expect that these conclusions also apply to other methods of particle filtering that rely on weighted re-sampling.

2.4 Conclusions

In this paper, we have proposed a parameterized likelihood function based on gradient direction matching. We have presented experimental results showing the influence of the observation model parameters on object tracking by CONDENSATION, where we distinguished between three different tasks of object tracking. The results show a significant dependency of performance on the model parameters. Hence, performance can be improved significantly by optimizing the observation model. However, the optimal observation model parameters are different for all three tasks. A trade-off has to be made between tracking properties. This also implies that the true observation probability is not always the optimal likelihood function. This is because, for robust tracking, the limited amount of particles used in practice must be concentrated as much as possible on the most important areas instead of approximating the complete posterior distribution.
Bibliography


2. Influence of The Observation Likelihood Function on Particle Filtering Performance in Tracking Applications


Chapter 3

Isophote Properties as Features for Object Detection

©2005 IEEE. Personal use of this material is permitted. However, permission to reprint/republish this material for advertising or promotional purposes or for creating new collective works for resale or redistribution to servers or lists, or to reuse any copyrighted component of this work in other works, must be obtained from the IEEE.

Abstract Usually, object detection is performed directly on (normalized) gray values or gray primitives like gradients or Haar-like features. In that case the learning of relationships between gray primitives, that describe the structure of the object, is the complete responsibility of the classifier. We propose to apply more knowledge about the image structure in the preprocessing step, by computing local isophote directions and curvatures, in order to supply the classifier with much more informative image structure features. However, a periodic feature space, like orientation, is unsuited for common classification methods. Therefore, we split orientation into two more suitable components. Experiments show that the isophote features result in better detection performance than intensities, gradients or Haar-like features.

3.1 Introduction

In order to evaluate the presence of an object in an image, relevant and robust properties of the image must be extracted that can be processed to classify or compare objects. One of the most popular features for object detection has been information about the edges, as used by the well known Chamfer Matching technique [1] and the face recognition method in [2]. Edges contain information about the shape, but they cannot describe smooth surfaces. The shape of these surfaces is visible only because of shading and reflections [3]. Moreover, highly curved surfaces have approximately constant orientation of shading under varying lighting directions [4]. Isophotes follow constant intensity and therefore follow object shape both around edges as well as smooth surfaces. As such they are closed curves within the image.

A totally differentiable curve can be completely described at any point \( a \) on the curve by the Taylor expansion \( \alpha(s) \) of the curve parameterized by arc length \( s \):

\[
\alpha(s) = \sum_{n=0}^{\infty} \frac{\alpha^{(n)}(a)}{n!}(s - a)^n
\]  

(3.1)

Here, \( \alpha(s) \) is a two dimensional vector of the spatial coordinates of the curve at position \( s \), \( a \) is the point on the curve where all curve derivatives are measured and \( \alpha^{(n)}(a) \) is the \( n \)th derivative of \( \alpha \) at point \( a \). Isophotes are not necessarily totally differentiable, however, we will only use the first two derivatives and assume that these exist:

\[
\tilde{\alpha}(s) = \alpha(a) + \alpha'(a)(s - a) + \frac{1}{2}\alpha''(a)(s - a)^2 + R
\]  

(3.2)

where \( \alpha'(a) \) is the tangent vector at \( a \), \( \alpha''(a) \) is directly related to curvature \( \kappa \) and \( R \) contains all higher order terms that are discarded when only direction and curvature are used. We further assume that the tangent and curvature change smoothly over the curve. This implies that isophotes can be described by a sparse set of directions and curvatures. Isophote direction and curvature can be computed directly from gray images [5].

Isophote properties have been used for object detection before. Froba and Kublbeck have used isophote orientation as features for face detection in [6] where they computed an average face model and used angular distances to obtain a similarity measure.
3.2 Isophote Orientation and Curvature Features

Freeman and Roth have used orientation histograms for hand gesture recognition [7]. Ravela and Hanson have used histograms of both isophote orientations and curvatures to compare two faces [8] and Maintz et al. [9] have used curvature features to detect ridges in CT/MR images of the human brain. Recently, Froba and Ernst [10] have used the Census Transform [11] as features for face detection. This transform also captures local image structure information. It can distinguish 511 possible local shapes. Apart from detection, isophotes have also been used for image segmentation [12].

Instead of computing isophote (or histogram) similarities to an (average) object model or using an exhaustive amount of structure features, we propose to use both orientations and curvatures directly as features for training a classifier. To make orientation suitable for classification it is further decomposed into a symmetric and a binary feature. Furthermore, we include a different approach for computing the isophote properties, using gradient structure tensor smoothing. We evaluate the performance of isophote orientation and curvature as object descriptors by applying them to face detection, since face detection is a well studied object detection problem for which a lot of experimental data and results are available.

3.2 Isophote Orientation and Curvature Features

An important parameter for calculating isophote properties is the scale \( \sigma_s \) which defines the detail of the structure that is described by the isophotes. Given \( \sigma_s \), there are two distinct methods to compute the isophote properties. We shall refer to these methods as the ‘direct isophote’ (I-) and ‘structure tensor’ (T-) method respectively.

3.2.1 Direct Isophote Properties

In the direct method, regularized first and second order derivatives are applied directly at scale \( \sigma_s \). The local isophote direction \( \phi_i \), is given by

\[
\phi_i = \text{arg}(D_y - jD_x), \phi_i \in [0, 2\pi)
\]  

(3.3)

where \( D_x \) and \( D_y \) are the first order derivatives of the image in horizontal and vertical direction respectively. In the experiments presented in this paper, the derivatives are calculated using Gaussian regularization with \( \sigma_s = 1.5 \) pixels. \( \phi_i \) is directed along the isophote in the direction that keeps the brighter side at the right. On a uniformly colored surface, the brighter side depends on the illumination direction. This can cause a \( \pi \) rad ambiguity, making \( \phi_i \) bimodal. Also around ridges \( \phi_i \) flips \( \pi \) rad, causing multimodality when the image is not perfectly registered. Because multimodal classes are more difficult to classify using standard methods, the sign is split from the direction:

\[
\theta_i = \phi_i \text{(mod } \pi), \theta_i \in [0, \pi)
\]

(3.4)

\[
\gamma_i = \begin{cases} 
1, & D_x \geq 0 \\
-1, & D_x < 0 
\end{cases}
\]

(3.5)

\( \theta_i \) and \( \gamma_i \) are shown in figure 3.1 (b) and (e), respectively, for concentric circles with Gaussian noise.
Curvature, $\kappa = 1/r$, is defined as the rate of change of direction along a curved line, with $r$ the radius of a circle that has identical curvature. Local isophote curvature $\kappa_i$ can be computed in an image according to [5]

$$\kappa_i = \frac{d\theta_i}{ds} = \frac{-(D_y^2 D_{xx} - 2 D_y D_x D_{xy} + D_x^2 D_{yy})}{(D_y^2 + D_x^2)^{3/2}}$$

The sign of $\kappa_i$ depends on the intensity at the outer side of the curve. It is positive for a brighter outer side. To prevent multi-modal features, we separate the sign, $\varphi_i$, from the curvature:

$$\tilde{\kappa}_i = |\kappa_i|$$
$$\varphi_i = \begin{cases} 1, & \kappa_i \geq 0 \\ -1, & \kappa_i < 0 \end{cases}$$

$\tilde{\kappa}_i$ and $\varphi_i$ are shown in figure 3.1 (c) and (d), respectively, for concentric circles with Gaussian noise.

The difficulty with using curvature as a feature is that it can take on any value between $-\infty$ and $\infty$ and curvature difference is not proportional to similarity. Classifiers generally have great difficulty with such features. Therefore, we first transform the feature space of $\tilde{\kappa}_i$ to a space where it is more uniformly distributed, by mapping
3.2. Isophote Orientation and Curvature Features

3.2.1 Orientation and Curvature Features

\[ \kappa_i \] with its Cumulative Distribution Function (CDF) for i.i.d. Gaussian noise \( F_X(\kappa_i) \):

\[ \hat{\kappa} = F_X(\kappa) = \int_{-\infty}^{\kappa} f_X(\kappa)dx \]  

\( f_X(\kappa_i) \) is estimated by computing \( \kappa_i \) over an image with 300x300 Gaussian distributed pixels.

3.2.2 Gradient Structure Tensor Properties

As can be seen in figure 3.1 (b, c), the isophote orientation and curvature suffer from singularities. This is because they are not defined for pixels with zero gradient. A solution is to use the orientation tensor representation, as explained in [5]. In this approach, \( D_x \) and \( D_y \) are first computed at a small scale, from which the gradient tensor \( G \) is computed. The tensor components are smoothed over a neighborhood, obtaining the average tensor \( \bar{G} \), called the Gradient Structure Tensor (GST):

\[ \bar{G} = \left[ \begin{array}{cc} G_{11} & G_{12} \\ G_{21} & G_{22} \end{array} \right] = \left[ \begin{array}{cc} D_x^2 & D_x D_y \\ D_x D_y & D_y^2 \end{array} \right] \]

where the bar (\( \bar{} \)) denotes the result after applying a smoothing operator with scale \( \sigma_s \).

In the experiments, the small-scale horizontal and vertical derivatives are computed by convolution with \( [\frac{1}{2} \ 0 \ \frac{-1}{2}] \) and \( [\frac{-1}{2} \ 0 \ \frac{1}{2}]^T \), respectively. Tensor smoothing is performed with a Gaussian filter with \( \sigma_s = 1.5 \). The smooth GST can be used to compute isophote properties with much less singularities. GST orientation \( \theta_T \) is calculated by

\[ \theta_T = \frac{1}{2} \arctan \left( \frac{2\bar{G}_{12}}{\bar{G}_{11} - \bar{G}_{22}} \right) + \frac{1}{2} \pi, \theta_T \in (0, \pi] \]  

The result is shown in figure 3.1 (f). GST curvature \( \kappa_T \) is calculated by [5]

\[ \kappa_T = -\cos(\theta_T) \frac{\partial \theta_T}{\partial x} - \sin(\theta_T) \frac{\partial \theta_T}{\partial y} \]
3. Isophote Properties as Features for Object Detection

Figure 3.3: Comparison of orientation representations. Resulting ROC curves for the testset that is explained in section 3.3

with

\[
\frac{d\theta_T}{dx} = \Re \left\{ \frac{1}{2} j e^{-j2\theta_T} \left( \frac{\partial \cos(2\theta_T)}{\partial x} + j \frac{\partial \sin(2\theta_T)}{\partial x} \right) \right\}
\]

\[
\frac{d\theta_T}{dy} = \Re \left\{ \frac{1}{2} j e^{-j2\theta_T} \left( \frac{\partial \cos(2\theta_T)}{\partial y} + j \frac{\partial \sin(2\theta_T)}{\partial y} \right) \right\}
\]

where \(\Re(c)\) is the real part of \(c\). \(\partial \sin(\theta_T)\) and \(\partial \cos(\theta_T)\) to \(x\) and \(y\) are calculated by convolution with \([\frac{1}{2} 0 -\frac{1}{2}]\) and \([-\frac{1}{2} 0 \frac{1}{2}]\T\), respectively. \(\tilde{\kappa}_T = |\kappa_T|\) and \(\varphi_T\), see figure 3.1 (g, h), are also separated to prevent multi-modal features. The positive sign of \(\kappa_T\) now corresponds to curves that have their outer side directed towards the right side of the image, as can be seen from the signs in fig. 3.1 (h). Also \(\tilde{\kappa}_T\) is transformed by its CDF in Gaussian noise using equation 3.9 to obtain \(\hat{\kappa}_T\) as a feature.

3.2.3 Orientation Features

The orientation \(\theta\) is discontinuous with a jump at every \(\pi\) rad. This property is not well suited for classification because it will split classes where they cross \(\theta = \pi\). To reduce this problem the orientation can be represented by two features. Three different representations are shown in figure 3.2: double orientation (a) where the discontinuities are at different positions, vector representation (b) and orientation magnitude and sign (OM and OS) (c) computed by

\[
OM = |\theta/\pi - 0.5|
\]

\[
OS = \begin{cases} 
0, & \theta < \frac{1}{2}\pi \\
1, & \theta \geq \frac{1}{2}\pi
\end{cases}
\]
3.3 Experimental Results

We will compare isophote features to pixel, gradient and Haar-like features, all computed after histogram equalization, while the isophote features are computed without histogram equalization since they are invariant to contrast.

3.3.1 Features

The features sets that will be compared are shown in table 3.1. The Haar-like features, as used in [13], are computed at approximately the same scale as the other features. The filter sizes for the horizontal and vertical filters are 2 by 4 pixels. The size of the diagonal filter is 4 by 4. Because these filters are even-sized and the face patches are odd-sized, the center results are averaged to obtain a symmetrical odd-sized feature where OM is symmetric and OS indicates the side of the symmetric OM. Note that OM corresponds to ‘horizontalness’.

To select the best representation, an experiment is performed similar to the experiments explained in section 3.3. A feature set was obtained by concatenating the features resulting from the direct isophote and GST orientations. The best ROC curves of three different classifiers are shown in figure 3.3. The vector and magnitude/sign representations provide the best results. Furthermore, the fact that OM features are different from OS features can give this representation an advantage when it is used in combination with other features, since maybe only either OM or OS is interesting to combine with certain other features, while the two components of the vector representation do not have any distinct property to offer over the other one. Therefore, we have used the OM/OS orientation representation in the experiments of section 3.3.

Table 3.1: Feature set names and descriptions

<table>
<thead>
<tr>
<th>Feature set name</th>
<th>Set size</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Illumination:</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>NL361</td>
<td>9x9x9x91</td>
<td>All Normalized histogram equalized Luminance values</td>
</tr>
<tr>
<td>NL81</td>
<td>9x9x9x81</td>
<td>Grid of pixels selected after Gaussian smoothing with standard deviation std. of 3 pixels and histogram equalization</td>
</tr>
<tr>
<td><strong>Gradient:</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>GH</td>
<td>9x9x9x81</td>
<td>Horizontal gradient magnitudes from the histogram-equalized face using filtering with Gaussian derivatives, std. 1.5 pixels</td>
</tr>
<tr>
<td>GV</td>
<td>9x9x9x81</td>
<td>Same as GH but vertical</td>
</tr>
<tr>
<td>G</td>
<td>9x9x9x81</td>
<td>$G = \sqrt{GH^2 + GV^2}$, used instead of GH and GV in the experiments with all feature sets</td>
</tr>
<tr>
<td><strong>Haar-like features:</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>H2H</td>
<td>9x9x9x81</td>
<td>Horizontal differences</td>
</tr>
<tr>
<td>H2V</td>
<td>9x9x9x81</td>
<td>Vertical differences</td>
</tr>
<tr>
<td>H3H</td>
<td>9x9x9x81</td>
<td>Horizontal peak filter</td>
</tr>
<tr>
<td>H3V</td>
<td>9x9x9x81</td>
<td>Vertical peak filter</td>
</tr>
<tr>
<td>H</td>
<td>9x9x9x81</td>
<td>Diagonal filter</td>
</tr>
<tr>
<td><strong>Isophote features:</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>IDS</td>
<td>9x9x9x81</td>
<td>Isophote Direction Sign ($\gamma_i$)</td>
</tr>
<tr>
<td>IOM</td>
<td>9x9x9x81</td>
<td>Isophote Orientation Magnitude derived from $\theta_i$</td>
</tr>
<tr>
<td>IOS</td>
<td>9x9x9x81</td>
<td>Isophote Orientation Sign derived from $\theta_i$</td>
</tr>
<tr>
<td>TOM</td>
<td>9x9x9x81</td>
<td>GST Orientation Magnitude derived from $\theta_T$</td>
</tr>
<tr>
<td>TOS</td>
<td>9x9x9x81</td>
<td>GST Orientation Sign derived from $\theta_T$</td>
</tr>
<tr>
<td>IC</td>
<td>9x9x9x81</td>
<td>Normalized Isophote Curvature $\epsilon_i$</td>
</tr>
<tr>
<td>ICS</td>
<td>9x9x9x81</td>
<td>Normalized GST Curvature $\epsilon_T$</td>
</tr>
<tr>
<td>TC</td>
<td>9x9x9x81</td>
<td>Normalized GST Curvature Sign $\epsilon_T$</td>
</tr>
<tr>
<td>TCS</td>
<td>9x9x9x81</td>
<td>GST Curvature Sign $\epsilon_T$</td>
</tr>
</tbody>
</table>
set. With each of the five filters a feature set of 9x9 values is obtained from a normalized 19x19 image patch. Note that Haar-like features usually also include longer versions of the filters. These are omitted here, as they are equivalent to combinations of the short filters.

3.3.2 Data Sets

The databases used in the experiments are shown in figure 3.5. The face examples that are used for training and feature selection are taken from the Yale Face Database B [14]. This database consists of images of 10 different subjects taken under a discrete set of different angles and illuminations. To obtain more robustness to rotations, the images were randomly rotated by a uniform distribution between -20 and 20 degrees. The rotated images were re-scaled and face patches of 19 by 19 pixels were cut out and finally mirrored to obtain more samples. Faces that were close to the border of the image were left out. One part of these samples was used for training and the other for feature set selection. For testing we used the CMU Test Set 1 [15], which consists of images of scenes containing one or more (near) frontal human faces. The faces were re-scaled and cut out to obtain a total of 471 face patches. As non-face examples image patches were obtained at different scales from images of the Corel image database that did not contain any faces. 10,000 of the selected patches were the ones that looked most similar to a face according to a face classifier using quadratic Bayes classification on a combination of isophote features and luminance values.

3.3.3 Classifiers

All features are normalized to have standard deviation 1 over the entire training set. Three different classifiers were used: the linear Fisher discriminant, quadratic Bayes (Normal-based) and unbounded Bayes (Parzen density). See [16] for more details on these classification methods. With the quadratic classifier Principle Component Analysis (PCA) is performed on the face class features (similar to [17]) to select the most important eigenvectors that, together, contribute to 99% of the total variance. The Parzen density classifier is not practical since the classification is very slow but it has good performance on complex, non-linear class distributions. Note that these are
3.3. Experimental Results

Figure 3.5: Datasets used in the experiments. By randomly drawing from two face (F) sets and one non-face (NF) set, three non-overlapping datasets are obtained. The gray arrow denotes selection based on high face detector output.

- faces: 5,590 from Yale
- non-faces: 24,162 random
- 10,000 difficult
- training set: 2,500 F, 5,000 NF
- feature selection: 3,090 F, 15,000 NF
- test set: 471 F, 14,162 NF
- faces: 471 from CMU
- non-faces: 10,290,384 from Corel

single-stage classifiers, while in practical situations a cascade of classifiers combined with boosting, like described in [13], is applied to obtain real-time speed. Since we want to evaluate the features themselves, speed is not regarded in these experiments.

To select an optimal feature set, a feature-set selection procedure is followed. By forward selection, at each step the feature set is added to the existing set that minimizes the area above the ROC curve until there is no feature set left that results in a decrease of the ROC area. The PCA procedure before the Normal-based classifier training was applied after combining the selected feature sets.

The results in table 3.2 show the performance on the feature selection data set and the test set. The feature types of the experiments correspond to the type of feature sets that the feature selection procedure was limited to. See table 3.1 for more details on the feature sets. In this way, the luminance, gradient, Haar-like and isophote feature sets are tested individually. The selected feature sets are shown in the order of selection. The ‘all features’ experiments exclude NL361, and GH and GV are replaced by G (see table 3.1). The resulting ROC curves of the combined sets are shown in figure 3.6. These results are for the classifier that resulted in the smallest area above the curve.

3.3.4 Discussion

The isophote properties result in better classification (smaller area above ROC curve) than the normalized luminance values, gradients or Haar-like features, for all three classifiers. The combination of all features resulted in a slightly better classification over the selection set, but on the test set the best result was obtained with the isophote properties alone, indicating that isophote properties generalize better. For the classifiers using all features, most of the selected sets were isophote properties. This indicates that the isophote properties capture the most important information about the structure of the face and luminance and gradient magnitudes are less essential.

There is no clear preference between GST and direct isophote features, though. With all three classifiers, pairs of similar features for the two different approaches
3. Isophote Properties as Features for Object Detection

Table 3.2: Feature selection and classification results. The selected feature sets are in the order in which they were selected. For the Normal-based classifier the second number of features is the number of principle components. The area’s above the ROC curves are computed both for the feature selection dataset and the test set.

<table>
<thead>
<tr>
<th>Feature</th>
<th>Selected sets of applicable otherwise all used sets</th>
<th>ROC test</th>
<th>ROC normal-based:</th>
<th>ROC test</th>
<th>ROC normal-based:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Normal-based:</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>NL361</td>
<td>[NL361]</td>
<td>0.0133</td>
<td>0.116</td>
<td>0.0133</td>
<td>0.116</td>
</tr>
<tr>
<td>NL81</td>
<td>[NL81]</td>
<td>0.0258</td>
<td>0.115</td>
<td>0.0258</td>
<td>0.115</td>
</tr>
<tr>
<td>Gradient</td>
<td>[Gradient]</td>
<td>0.0210</td>
<td>0.206</td>
<td>0.0210</td>
<td>0.206</td>
</tr>
<tr>
<td>Haar-like</td>
<td>[Haar-like]</td>
<td>0.0317</td>
<td>0.263</td>
<td>0.0317</td>
<td>0.263</td>
</tr>
<tr>
<td>Isophote</td>
<td>[Isophote]</td>
<td>5.81 × 10⁻⁴</td>
<td>0.0323</td>
<td>5.81 × 10⁻⁴</td>
<td>0.0323</td>
</tr>
<tr>
<td>Fisher:</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>NL361</td>
<td>[NL361]</td>
<td>0.0578</td>
<td>0.112</td>
<td>0.0578</td>
<td>0.112</td>
</tr>
<tr>
<td>NL81</td>
<td>[NL81]</td>
<td>0.0989</td>
<td>0.112</td>
<td>0.0989</td>
<td>0.112</td>
</tr>
<tr>
<td>Gradient</td>
<td>[Gradient]</td>
<td>0.0394</td>
<td>0.0543</td>
<td>0.0394</td>
<td>0.0543</td>
</tr>
<tr>
<td>Haar-like</td>
<td>[Haar-like]</td>
<td>0.0904</td>
<td>0.0654</td>
<td>0.0904</td>
<td>0.0654</td>
</tr>
<tr>
<td>Isophote</td>
<td>[Isophote]</td>
<td>1.09 × 10⁻³</td>
<td>0.0494</td>
<td>1.09 × 10⁻³</td>
<td>0.0494</td>
</tr>
<tr>
<td>Parzen:</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>NL361</td>
<td>[NL361]</td>
<td>0.0066</td>
<td>0.045</td>
<td>0.0066</td>
<td>0.045</td>
</tr>
<tr>
<td>NL81</td>
<td>[NL81]</td>
<td>0.0322</td>
<td>0.248</td>
<td>0.0322</td>
<td>0.248</td>
</tr>
<tr>
<td>Gradient</td>
<td>[Gradient]</td>
<td>0.0248</td>
<td>0.180</td>
<td>0.0248</td>
<td>0.180</td>
</tr>
<tr>
<td>Haar-like</td>
<td>[Haar-like]</td>
<td>1.33 × 10⁻³</td>
<td>0.188</td>
<td>1.33 × 10⁻³</td>
<td>0.188</td>
</tr>
<tr>
<td>Isophote</td>
<td>[Isophote]</td>
<td>1.88 × 10⁻⁸</td>
<td>0.0454</td>
<td>1.88 × 10⁻⁸</td>
<td>0.0454</td>
</tr>
<tr>
<td>all features</td>
<td>[all features]</td>
<td>7.96 × 10⁻⁵</td>
<td>0.0543</td>
<td>7.96 × 10⁻⁵</td>
<td>0.0543</td>
</tr>
</tbody>
</table>

are combined to improve performance, suggesting that the two approaches capture different structural information. From the Haar-like features, only two or three sets were selected for Normal-based and Parzen classification, while much more sets were selected from the isophote properties. Apparently, the Haar-like features have more redundancy than the isophote features. The Parzen classifier nearly always outperforms the other two classifiers on the feature selection set but not on the test set. This is because Parzen density is more complex, hence more sensitive to ‘over-training’ and, therefore, does not generalize well.

### 3.4 Conclusions and Future Work

We proposed to use a combination of isophote properties to obtain a compact and descriptive representation of image structure that is invariant to contrast and robust to illumination changes. Furthermore, to make orientation features suitable for classification they are separated into a symmetrical and a binary feature. We applied it to face detection and compared the isophote features to Haar-like features, gradients and normalized luminance values, which are often used for face detection. The experiments show a better classification performance, especially when applied to a separate test set, which indicates better generalization. The direct and the GST approach for obtaining isophote features supplement each other, indicating that they capture different information about the object structure. Only single-scale-features were used here, while all features can also be computed at other scales to obtain even better classification performance. In these experiments, speed is not taken into account. The Haar-like features can be computed efficiently using the integral image, as explained in [13], while the isophote features in this paper are computed using Gaussian filtering, trigonometric
3.4. Conclusions and Future Work

Figure 3.6: ROC curves. (a) at the end of feature set selection, (b) classification on the different test set. The results are for the classifier that resulted in the smallest area above the curve.
and modulo calculations, which slow down the computation significantly. One possibility is to compute the isophote properties from an image pyramid with only a few scales and then apply nearest neighbour interpolation on the closest scale(s) to obtain the features on scales in between. The best application for isophote features seems to be object tracking, where the approximate scale of the object can be predicted from the previous observations. A multi-scale search needs to be performed only at initialization and features need to be re-computed only where the local image content has changed.
Bibliography


Chapter 4

A Shadow Color Subspace Model for Imperfect, Not-Calibrated Cameras

Abstract Color-based detection is usually done in chromatic or HSV color space, for which camera calibration is necessary and where noise is brightness dependent. This makes it hard to obtain a color model that is completely invariant to lighting. Therefore, we propose a new method to model shadows, consisting of a piecewise linear RGB subspace. Shadow invariant color change is classified by computationally inexpensive inner products of RGB vectors. Unlike other shadow-invariant color change detection methods, this method does not need RGB origin alignment or thresholds to discard unreliable colors in dark or low-saturated regions. Furthermore, relaxation of the constraints of camera calibration and linearity is controllable, which makes the method useful for controlled as well as uncontrolled conditions. We demonstrate our method on a difficult foreground detection example, that shows that the RGB shadow subspace model outperforms detection by hue or normalized rgb, even after using a totally different camera, without having to adapt the parameters of our method.

4.1 Introduction

Color is widely used in computer vision. This is mainly because colors are often very discriminative features of objects. However, from the start of computer vision research on, shadows have posed a great threat to robust color-based detection schemes. Shadows change the observed color that is reflected from an object in an, often, unpredictable manner. Furthermore, when the light is captured by a digital camera with a non-linear, and/or even unknown, light sensitivity curve, the shadow problem becomes even more complex. A summary of different algorithms for shadow detection can be found in [1].

We are interested in modeling shadows in the context of foreground detection, where the foreground objects cast shadows on the background, causing undesired false positives. Particularly, we want to use foreground detection to detect the skin of a person in a moderately controlled indoor scene for the application of hand gesture recognition. The procedure of skin color detection is shown in figure 4.1. In an initialization procedure where the user hides his/her hands (e.g. behind the back), a global (the same for the entire image) skin color lookup table is built, where the color of the person’s clothing is excluded, see (a). Note that any color that is excluded from this global model potentially impoverishes the skin color model, so we do not want to exclude any colors unnecessarily. Because we assume that background colors remain at a fixed position (local), we definitely do not want to exclude them from the global model, thus any false positives during the acquisition of the person’s clothing colors must be prevented. False negatives are less of a problem here, since a person’s clothes often contain only a few different colors, hence only small parts of the clothing can provide enough samples. During skin color detection, as shown in (b), we want to exclude background color locally, at pixel level, for which we perform a color-based foreground detection.

The foreground detection used here must be invariant to shadows, because otherwise skin colored background would be falsely detected as skin when a shadow is cast over it. But we want to confine the background color model as much as possible to reduce the possibility of completely losing hands in front of a skin colored
4.1. Introduction

Figure 4.1: Skin color detection using a global skin color lookup table without clothing colors and a local background color model. (a) Initialization: Skin and clothing colors are extracted to form a skin lookup table where clothing colors are excluded. (b) Skin color detection procedure.

background. When the background color can be confined to only a small part of the skin color model, parts of the hands can still be retrieved in front of it. Therefore, we need an accurate model for the set of colors a specific background pixel could take on, when it would be subject to reduced (blocked) illumination. The model should be as tight as possible, but general enough to work under a variety of unknown conditions. Re-calibration when our gesture recognition system is used in a new environment is undesirable. Therefore, it should work without precise color calibration.

Different methods already exist for brightness invariant color description. A well known color model that is widely used for color detection is the Hue, Saturation, Value (HSV) description of color [2]. By discarding the ‘Value’ component, the description is theoretically invariant to illumination. Many researchers have applied this method for color detection [3, 4]. Because shadows also change Saturation when the camera is not calibrated, often mainly only the hue component is used for color detection.
Because hue is not defined for unsaturated colors, this method fails for (nearly) black, gray and white objects.

Others have used chromatic color descriptions derived directly from RGB space [5, 6], or using color derivatives instead of absolute colors [7]. These work better for unsaturated colors but still fail for dark colors, because then noise, ill-calibration or inhomogeneous illumination has a large influence on chromatic color. An approach which overcomes both instabilities is proposed in [8].

A strict assumption that all these illumination-invariants share is that the actual color origin is aligned with the origin of RGB space. When this assumption does not hold, a large decrease of intensity, caused by a shadow, will still cause a large change of the invariant. This makes a calibration procedure necessary. But a camera cannot be calibrated beforehand, because the real color origin depends on the camera settings (exposure, gain, brightness, etc.) and the settings need to be adapted to every specific situation in order to keep the captured colors within the limited RGB space.

Other attempts to obtain an illumination invariant color description have combined multiple pixels [9, 10] to model color relationships between pixels instead of exact color. In our application we want to detect background color change at pixel-level to aid skin color detection. Furthermore, total illumination invariance is not necessary for foreground detection when the illumination does not change during operation. Besides, more invariance means less discriminative power. The main problem we need to solve is the miss-alignment of the color origin that has a negative effect on shadow robustness of current single-pixel brightness invariant color descriptions.

Figure 4.2 shows how much the color origin can vary between situations. These are the scatter plots of the RGB colors of five different uniformly colored balls (red, blue, green, yellow and orange) that are captured by two different cameras with a regular office lighting. The cameras were configured by manually changing the settings until a perceptually clear color image on a CRT monitor was obtained. Because of the round shape of the balls, pixels of the same color under a range of illuminations are captured at the same time. Therefore, the scatter plots show the path that each object color follows when it is subject to shadow. In (a) the shading paths run parallel to the diagonal, corresponding to a real color origin in $(-\infty, -\infty, -\infty)$, while (b) shows an origin somewhere on the positive side of the diagonal. Furthermore, because the color space is tilted due to incorrect white-balance, the blue color is on the diagonal, corresponding to grey. These examples clearly show that either the camera must be calibrated, or the chrominance thresholds must be adapted to the chrominance spread that is introduced by the offset of the real color origin. An automatic way to adjust these thresholds is proposed in [4]. An even more difficult problem is to model the non-linear change of the shading path when one of the RGB channels becomes zero, as can be seen with all colors in (a). Even when the path follows a constant hue, a slanted intersection of the hue component with one of the RGB planes will cause a change in hue that is not modeled by the classical chrominance models.

We propose a shadow model in RGB space that incorporates variation of the real color origin. When the real origin has negative RGB value, our model also describes how shadows follow the RGB side-planes when any of the RGB channels gets zero. The advantage of using RGB space is that we can apply a margin in all directions,
based on the standard deviation of the additive noise in the respective directions.

Our assumptions are:
1) Cast shadows do not increase RGB values of the background.
2) Shadow does not significantly change the real hue of reflected colors.
3) Image noise is additive in RGB space.
4) The real color origin lies close to the diagonal of RGB space between \((-\infty, -\infty, -\infty)\) and the maximum brightness value.
5) The Euclidean distance between the diagonal of RGB space and measured object color is a constant or strictly increasing function of illumination brightness.

Assumption 1) can be regarded as generally valid. Assumption 2) will be violated when assumption 4) does not hold or when illumination of the scene is inhomogeneous, causing shadows to block only a certain color, hence changing the color of the light that illuminates the object, or when a surface has both specular and diffuse reflection and these two components are not reduced equally by shadow. Assumption 3) is generally regarded as true for thermal noise in most CCD cameras. Note that we do not assume equal standard deviation for the three color channels. Assumption 4) is the main assumption that our method is based upon. It is much more relaxed than the usual assumption of total origin alignment, but we still assume there is no off-diagonal origin offset. Experiments have shown that this is a reasonable assumption, but otherwise, an approximation of the offset should be estimated automatically to correct it. Assumption 5) will be violated when a maximum color channel value is reached, either by over-exposure of the CCD chip or hitting the maximum representation value of the channel (typically, 255 for a 24 bit color image). When a channel maximum is hit, the color will be pushed back towards the diagonal when brightness increases. This can be prevented by setting the exposure, gain and brightness of the camera at a sufficiently low level.

4.2 Projection Confined Shadow Subspace

Our shadow model defines the subspace of RGB color space that contains the possible colors that a (background) pixel with color vector \(\vec{x}\) can take on when it is subject to cast shadow, taking into account uncertainty about the camera properties and settings. If a color lies inside this subspace of the background color, it is regarded unchanged or changed by shadow, and the respective pixel is classified as background. If a color is outside the subspace, its difference with the background color is assumed not to be due to shadow and the respective pixel is classified as foreground.

The shadow subspace model is nothing more than a translation of our assumptions in Sec. 4.1 to RGB space limits. By approximating these limits by a piece-wise linear convex hull, the shadow subspace can be completely enclosed by a set of planes. This
Figure 4.2: RGB scatter plots of colored balls under office lighting, including the projections onto an equal-brightness plane. (a) Captured by a Philips ToUcam Pro II USB webcam, where the paths of the colors under different shading seem to run parallel, corresponding to a real color origin of \( O = (\infty, \infty, \infty) \). (b) Captured by a Videre DCAM-L firewire camera, where the paths seem to meet somewhere on the positive side of the diagonal, near the point \((20, 20, 20)\).
4.2. Projection Confined Shadow Subspace

is shown in figure 4.3 (a). Since the inner product with the unit vector normal to a plane is equal to the norm of the projection onto that normal, hence, the distance to the plane, computationally inexpensive inner products with the unit normals can be used to determine if a color is on the ‘inner’ or ‘outer’ side of each of the planes. The decision of foreground classification is made by a cascade of boolean evaluations of the vector products, making the method even more computationally efficient. If any of these products exceeds a threshold, the pixel is classified as foreground. If all projections are below the thresholds, the color is classified as background.

To decide if a color \( \vec{y} \) is in- or outside the Projection Confined Shadow Subspace (PCSS), denoted by \( D_1 = 0 \) and \( D_1 = 1 \) respectively, the following sequence of decisions D1-D7 are performed, of which the planes belonging to D1-D5 are also indicated in figure 4.3 (a):

**Decision D1:**
Following assumption 1), the separate color channels are compared to the background color to detect a color increase larger than a noise threshold. Furthermore, colors with negative R, G or B value are somehow mapped to one of the side-planes of RGB space. When this is also the plane that the background color model intersects, these colors will be modeled differently. To detect this occurrence, we define the ordered color channels by \( C_1, C_2 \) and \( C_3 \) as the lowest, the middle and the highest channel of the RGB value \( \vec{x} \) of the background, respectively. Note that these are not the values but the indices of the three color channels. When \( C_1 y \) is zero, the side-plane model of the background is evaluated instead of the 3D model, which will be explained further on.

\[
D = D_1 = \begin{cases} 
1, & R_y > R_x + t_R \\
& G_y > G_x + t_G \\
& B_y > B_x + t_B \\
D6, & C_1 y = 0 \\
D2, & otherwise 
\end{cases}
\]  

(4.1)

Where \( \vec{y} \) is the measured color, \( R, G \) and \( B \) are the red, green and blue values and \( t_R, t_G \) and \( t_B \) the noise thresholds for the respective color channels.

**Decision D2:**
The basis of the PCSS model is a triangle between \( \vec{x} \) and the maximum and minimum expected real color origin locations on the diagonal, \( \vec{O}_{max} \) and \( \vec{O}_{min} \) respectively, see figure 4.3. Building on assumption 2), we assume that the background color stays within a margin from both sides of this hue plane, illustrated by the thickness of the PCSS hull in (a). Note that, because of using a planar model, also in-plane-curved paths are modeled intrinsically, as long as assumption 5) is not violated. Figure 4.2 shows that some paths can be slightly curved, but the projections do not clearly show any curving, implying that indeed only the saturation component is curved, not the hue. Exceeding the maximal distance from the basis plane (denoted by D2) is tested.
Figure 4.3: The PCSS model in RGB space. (a) The hull of the PCSS, defined by noise margins around the basis triangle. It is confined by D1 and planes D2 to D5. (b) The basis PCSS triangle and its 2D side-plane model, both colored in dark gray. (c) The 2D side-plane model in the C2, C3 plane.
4.2. Projection Confined Shadow Subspace

by the inner product with the normal to the plane, \( \hat{n}_2 \):

\[
\begin{align*}
\vec{n}_2 &= \vec{x} \times \hat{v}_{\text{diag}} \\
\hat{n}_2 &= \frac{\vec{n}_2}{|\vec{n}_2|} \\
D_2 &= \begin{cases} 
1, & |\vec{y}\hat{n}_2^\top| > t_n|\hat{n}_2N| \\
D_3, & \text{otherwise}
\end{cases}
\end{align*}
\]  

(4.2)

Where \( \times \) denotes the cross product, \( \hat{v}_{\text{diag}} \) the unit vector in the direction of the diagonal of RGB space, \( t_n \) the noise threshold, and \( N \) the \( 3 \times 3 \) noise covariance matrix. Robustness to violation of assumptions 2) and 4) will also be handled by the noise threshold.

**Decision D3:**

Distance from the plane through \( \vec{x} \) and \( \vec{O}_{\min} \) (denoted by D3) is tested by the inner product with plane normal \( \hat{n}_3 \):

\[
\begin{align*}
\vec{n}_3 &= (\vec{x} - \vec{O}_{\min}) \times \hat{n}_2 \\
\hat{n}_3 &= \frac{\vec{n}_3}{|\vec{n}_3|} \\
D_3 &= \begin{cases} 
1, & \vec{y}\hat{n}_3^\top > \vec{x}\hat{n}_3^\top + t_n|\hat{n}_3N| \\
D_4, & \text{otherwise}
\end{cases}
\end{align*}
\]  

(4.3)

When \( \vec{O}_{\min} = (\infty, \infty, \infty) \), an in-plane unit vector perpendicular to the diagonal must be computed instead of equation 4.4:

\[
\hat{n}_3 = \vec{x}A
\]  

(4.6)

Where \( A \) is the projection matrix for the plane through 0 perpendicular to the diagonal.

**Decision D4:**

Since we have really observed shadow paths parallel to the diagonal in our experiments, we will keep \( \vec{O}_{\min} \) at \((\infty, \infty, \infty)\), to be general enough for our camera. Therefore, the plane through \( \vec{O}_{\max} \) and \( \vec{O}_{\min} \) is parallel to the previous plane and its normal is equal to \( -\hat{n}_3 \):

\[
\begin{align*}
D_4 &= \begin{cases} 
1, & \vec{y}\hat{n}_3^\top < -t_n|\hat{n}_3N| \\
D_5, & \text{otherwise}
\end{cases}
\end{align*}
\]  

(4.7)

**Decision D5:**

The last plane, through \( \vec{x} \) and \( \vec{O}_{\max} \), with normal \( \hat{n}_5 \):

\[
\begin{align*}
\vec{n}_5 &= \hat{n}_2 \times (\vec{x} - \vec{O}_{\max}) \\
\hat{n}_5 &= \frac{\vec{n}_5}{|\vec{n}_5|} \\
D_5 &= \begin{cases} 
1, & \vec{y}\hat{n}_5^\top > \vec{x}\hat{n}_5^\top + t_n|\hat{n}_5N| \\
0, & \text{otherwise}
\end{cases}
\end{align*}
\]  

(4.9)

**Decision D6:**

An exception is made when \( C_{\vec{y}} \) is zero, as explained above. In this case, a similar
method is applied in the 2D plane of the other two color channels, see also figure 4.3 (c). First, the intersection \( \vec{x}_{2D} \) of the line from \( \vec{O}_{\text{min}} \) to \( \vec{x} \) with the side-plane, is calculated by:

\[
\vec{x}_{2D} = \frac{C^{\vec{p}}}{C_1 - C_1^{\vec{p}}} \vec{x}_p + \frac{-C^{\vec{x}}}{C_1 - C_1^{\vec{x}}} \vec{O}_{\text{min}}
\]

(4.10)

\[
\vec{x}_p = \vec{x}_\hat{n}_3^\top \hat{n}_3
\]

(4.11)

\( \hat{n}_6 \) is obtained by projecting \( \vec{x}_{2D} \) onto \( \hat{n}(C_2, C_3) \), a unit vector in the \( C_2, C_3 \) plane that is perpendicular to this plane’s diagonal:

\[
\hat{n}_6 = \vec{x}_{2D} \hat{n}_6^\top (C_2, C_3) \hat{n}(C_2, C_3)
\]

(4.12)

\[
\hat{n}_6 = \hat{n}_6 / |\hat{n}_6|
\]

This is because the \( \vec{O}_{\text{min}} \) is chosen at \((-\infty, -\infty, -\infty)\).

\[
D_{6} = \begin{cases} 
1, & \vec{y}_{\hat{n}_6} > \vec{x}_{2D} \hat{n}_6^\top + t_n |\hat{n}_6 N| \\
0, & \text{otherwise}
\end{cases}
\]

(4.13)

**Decision D7:**

Also the other side of the triangle must be tested, which runs from 0 to \( \vec{x}_{2D} \):

\[
\hat{n}_7 = \vec{x}_{2D} \times \hat{n}_{\text{diag}2D} \times \vec{x}_{2D}
\]

(4.14)

\[
\hat{n}_7 = \hat{n}_7 / |\hat{n}_7|
\]

\[
D_{7} = \begin{cases} 
1, & \vec{y}_{\hat{n}_7} > \vec{x}_{2D} \hat{n}_7^\top + t_n |\hat{n}_7 N| \\
0, & \text{otherwise}
\end{cases}
\]

(4.15)

Where \( \hat{n}_{\text{diag}2D} \) is the unit vector in the direction of the diagonal of the \( C_2, C_3 \) plane.

In figure 4.4 the basis triangle and its 2D version are shown for the maximum color of the image of the red ball, together with the scatter plot, of figure 4.2 (a). This example shows how well the model fits real-world data from a poorly configured camera.

### 4.3 Experiments

We have tested the PCSS model in a difficult foreground detection example, shown in figure 4.5. The actual test images were much larger (640x480 pixels), but they are cropped to show only the subject. The same scene with the same lighting is captured with two different cameras, with and without a person. The lighting consist of a mixture of office lighting and a differently colored spot light. With this mixture of multiple differently colored and directed illumination sources and clothing of similar color as the background, it is hard to perform foreground detection purely based on color. No calibration is performed, but the cameras are configured by hand to make
Figure 4.4: The scatter plot of a red ball with its 3D and 2D PCSS triangles. Because shadow follows a path perpendicular to the RGB diagonal, the streak exactly follows the edge of our PCSS model.
Figure 4.5: Foreground detection results with two different cameras. For fair comparison, the result of step D1 in PCSS is added to all other results. (a-f) Results using the Philips ToUcam after tuning the threshold of each method. (g-l) Results using the Videre DCAM with the same thresholds as above, and (m-n) after re-tuning the thresholds for the hue and rg methods.
4.3. Experiments

the image perceptually good. This example will show how well our proposed method can cope with difficult situations where our assumptions might not even hold.

We have compared our model to Euclidean distance in RGB space, hue and Euclidean distance in normalized red and green space (rg). Since the third channel of normalized color is completely dependent on the other two (the three components add up to 1), and the blue channel was more noisy, we have omitted normalized blue. For the ease of comparison, the standard deviation of the noise is assumed equal in the red, green and blue channels. This leaves only one free noise parameter for all the methods that we compare, hence we can use this parameter to compare ROC curves. For hue and rg we have applied a fixed threshold on intensity of 50 to discard unreliable chromaticity of dark colors. These are all identified as background to prevent many false positives in dark regions. We have not applied a threshold on low saturation in the hue detection method, because that would result in not being able to detect change from or to gray color at all, which is unacceptable for our application as walls often appear gray. Moreover, because we do not calibrate the camera, gray in the image could correspond to any real color, depending on the white balance. The rg method should not have problems with low saturated colors.

The thresholds \( t_R, t_G \) and \( t_B \) in equation 4.1 are all set to 35. To make a fair comparison, the foreground pixels resulting from D1 are also added to the result of all other methods. The maximum origin of the PCSS was experimentally set to \( \vec{O}_{max} = (15, 15, 15) \).

Foreground detection results using the first camera are shown in figure 4.5 (a-f). The noise threshold of each method was tuned to detect as much of the foreground as possible with only a small amount of false positives in the background. PCSS detects more of the foreground than the other methods, with less false positives. Furthermore, it is the only result where all of the skin is correctly detected as foreground, which is especially important for the application of skin color detection.

The results of using the same thresholds with the other camera is shown in (g-l). Hue and rg now detect hardly anything additional to what is detected by the threshold on color increase, while PCSS still correctly detects the skin of the face and some other parts of the clothes. The improved results after re-tuning the thresholds for hue and rg are shown in (m-n). The PCSS result could not be improved by re-tuning the threshold. This is an advantage of PCSS over the other methods, since it makes color detection much more flexible. Even after re-tuning, the hue method still cannot detect the face as foreground and the rg method also misses a part of the neck, while they both already have more false positives in the background than PCSS. Not surprisingly, the RGB Euclidean distance method always falsely detects shadow that is cast on the background while detecting only little of the real foreground.

The Receiver Operating Characteristic (ROC) curves for these two situations are shown in figure 4.6. These curves show performance for a range of operating points by plotting false positive rate against true positive rate for different thresholds. The foreground truth is obtained by hand-drawing the person’s contour. False positive rate is defined as the number of background pixels falsely detected as foreground, divided by the total number of true background pixels. True positive rate is the number of correctly detected foreground pixels divided by the total number of true foreground
Figure 4.6: ROC curves of the foreground detection example. (a) Philips ToUcam Pro II USB webcam, (b) Videre DCAM-L firewire camera
4.4. Conclusions

pixels. Smaller distance to the upper left side of the ROC graph corresponds to better detection performance. In (a) the performance of PCSS is always equal to or better than the other methods. Surprisingly, rg works even worse than RGB. This is because chromatic colors of the foreground are very similar to the background. Hue works better because it is more discriminating for low saturated colors. These colors are always close in rg space but not in hue. In the left side of (b) PCSS works slightly worse than rg and RGB. This is because of a violation of assumption 5) on the wall behind the head. In the background image, at some pixels one or more of the three color channels have been clipped to the maximum value. When shadow decreases the RGB background values and the color ‘enters’ the RGB space, the distance to the diagonal (related to saturation) will increase. This effect is visible in the PCSS segmentation result in figure 4.5 (l) as the small blob connected to the left side of the head. This blob will grow some more when the noise threshold is set lower. Although this is an important shortcoming of the PCSS model, the problem can automatically be prevented by setting the camera’s brightness and/or gain low enough to keep all the RGB values in the background below maximum.

4.4 Conclusions

We have proposed a flexible, robust and computationally efficient shadow model consisting of a piece-wise linear RGB subspace. Using this model, color change that is not caused by shadow can be detected by a cascade of, at most, four inner products with the RGB color. It can be used as a method for background color comparison in the presence of cast shadows. Contrary to existing methods, the model takes into account miss-alignment of the color origin, caused by calibration inaccuracy. It can be configured for application in controlled (precise calibration) or uncontrolled (no calibration) situations. The experiments of foreground detection in an uncontrolled situation show significantly improved performance over the classical RGB, hue and normalized rg color detection methods. Furthermore, the optimal parameter settings did not have to be changed for a different camera, as opposed to the other methods that were tested.
Bibliography


Chapter 5

A Self-Calibrating Chrominance Model Applied To Skin Color Detection

Abstract In case of the absence of a calibration procedure, or when there exists a color difference between direct and ambient light, standard chrominance models are not completely brightness invariant. Therefore, they cannot provide the best space for robust color modeling. Instead of using a fixed chrominance model, our method estimates the actual dependency between color appearance and brightness. This is done by fitting a linear function to a small set of color samples. In the resulting self-calibrated chromatic space, orthogonal to this line, the color distribution is modeled as a 2D Gaussian distribution. The method is applied to skin detection, where the face provides the initialization samples to detect the skin of hands and arms. A comparison with fixed chrominance models shows an overall improvement and also an increased reliability of detection performance in different environments.

5.1 Introduction

Color is an important property of many objects. Therefore, it has been of great interest to researchers in the field of image analysis since the introduction of digital color images. However, the lack of constancy of color between different lightings, camera equipment and settings has challenged researchers ever since. To reduce the problem of color variation between different scenarios, color can be modeled adaptively. But even in the same scenario, color can change due to changing lighting conditions. A significant amount of research has been conducted to find functions of color that are invariant to illumination change. The most prominent functions that are commonly used are the chromatic color models. E.g. normalized rgb, YUV, YCrCb, HSV or CIELAB. In these color spaces, the brightness factor is isolated from the chromatic representation of color. This facilitates adaptive color modeling that is invariant to changes in illumination brightness.

However, brightness invariance is not guaranteed in these models. All these chromatic color space conversions are based on certain assumptions about color appearance in RBG space. Normalized rgb, HSV and CIELAB assume that black is represented by \([R, G, B] = [0, 0, 0]\) and, as a result, all colors meet each other at this point when their brightness is reduced. Contradictory, YUV and YCrCb assume that a brightness change results in a change of color parallel to the diagonal of RGB space. Furthermore, HSV assumes that gray (unsaturated) colors satisfy \(R = G = B\) (correct white balance) and CIELAB even needs a completely calibrated RGB (XYZ) space.

Violation of these assumptions, e.g. due to incorrect white balance, non-ideal camera sensitivity or settings or heterogeneous illumination, can severely degrade performance of color analysis methods based on these color spaces. This was shown by Martinkauppi et al. [1], who have tested robustness of different skin detection methods under large changes of conditions. To our knowledge, the validity in real-world digital imaging situations of the chrominance models that are often used for these methods, and the consequences of violating the model assumptions, have never been studied directly.

If the color spectrum of the illumination is homogeneous, the assumptions can be satisfied by a calibration procedure. However, in many real-world applications, such
a procedure is not desirable because it takes extra time and effort. It may not even be possible, e.g. in case of diversity of unknown cameras, non-expert users or processing of video previously recorded under non-ideal circumstances. Furthermore, calibration may not be effective, because of a difference between spectra of direct and ambient light, resulting in correlation between chrominance and brightness.

To avoid the shortcomings of chromatic models in real-world scenarios, we don’t want to rely on constraints on white balance, origin offset or correlation between brightness and chrominance. Therefore, we take the principle of chromatic color representation one step further, by presenting our Adaptive Chrominance Space (ACS) method that adaptively finds a linear function that minimizes correlation between brightness and the chrominance representation of the appearance of a specific object color.

Our application for this color model is hand gesture recognition. Skin color is modeled from samples obtained from face detection and applied to detect and track a person’s hands. Furthermore, two models are combined, fitted to samples from the left and right part of the face, respectively. This increases robustness to variation in illumination spectra from different horizontal directions.

Section 5.2 contains a summary of related research on skin color detection, section 5.3 describes our method, for which experimental results are provided in section 5.4. Our conclusions are given in section 5.5.

5.2 Related Work

Many methods for skin color detection have been proposed in the literature. Surveys can be found in [2, 3]. Most methods learn a general skin color distribution from a database of images [4, 5, 6], e.g. a selection from the internet. This results in very general models that take into account variation of camera, camera settings and illumination. However, because the variation of skin color appearance between different situations is so large, these general models lack precision to distinguish between real skin and skin-colored objects. This greatly restricts the reliability of skin segmentation, since the false positive rate will be high when other colors in the image are close to skin color.

To overcome this problem, some methods adapt to the specific situation by learning the skin color distribution from samples taken from the same video [7, 8, 9, 10, 11], often combined with a prior skin color distribution learned from a database. The problem with skin model adaptation is that it is difficult to obtain a large and representative sample set of skin color from a video automatically. Adaptive methods do not generalize well to other skin regions in the video if they lack a (realistic) chrominance model.

5.3 Color Detection Method

Instead of learning a general skin color distribution that generalizes too much, or relying on a chrominance model based on rigid, non-realistic assumptions, we propose
5. A Self-Calibrating Chrominance Model Applied To Skin Color Detection

Figure 5.1: RGB scatter plots of colored balls under office lighting without calibration. The dotted line indicates the diagonal of RGB space.

to use a general model of skin color variation for not-calibrated camera’s with fixed settings and non-changing illumination. We use an automatic procedure that fits this model to a small and noisy sample set.

The general model of skin color variation is explained in paragraph 5.3.1. Paragraph 5.3.2 explains how a similarity measure for skin color can be computed using this model, and 5.3.3 describes how the model is fitted to the sample set.

5.3.1 Skin Color Appearance Model

We define $x$ as a point at a specific image pixel location, corresponding to a point on the surface of an object with object color $\xi$ and $\theta$ as the spectrum of the illumination source. Our appearance model of skin assumes that the appearance in RGB space $\vec{C}_{RGB}(x, \xi, \theta)$ of color $\xi$ is a linear function $\vec{\ell}(x, \xi, \theta)$ of reflected light intensity $I(x)$ plus some independent random zero-mean noise vector $\vec{\eta}(x)$ (assuming Lambertian reflection):

$$
\vec{C}_{RGB}(x, \xi, \theta) = \vec{\ell}(x, \xi, \theta) + \vec{\eta}(x),
$$

(5.1)

$$
\vec{\ell}(x, \xi, \theta) := \vec{C}_{RGB0} + I(x)\vec{c}(\xi, \theta).
$$

(5.2)

where $\vec{c}(\xi, \theta)$ is a color vector with unit intensity and $\vec{C}_{RGB0}$ is the (unknown) calibration point for black. $I(x)$ depends on both the light source intensity and the sur-
5.3. Color Detection Method

Figure 5.2: Appearance model of skin color $p(C_{RGB}|skin)$ according to the multiplicative noise model, represented by an isosurface.

face tangent, of which the latter depends on $x$ in a non-deterministic way. For brief notation, $C_{RGB}(x, \xi, \theta)$ is denoted by $\vec{C}_{RGB}$ in the remainder of this work, silently assuming a specific object color, illumination and image pixel coordinates.

For fixed $\xi$ and $\theta$, $\vec{l}(x, \xi, \theta)$ is a straight line in RGB space. If the camera is not calibrated, $\vec{l}(x, \xi, \theta)$ does not have to pass through the origin. An example image without calibration is shown in figure 5.1 (a). This is an image of uniformly colored balls captured by a webcam under office lighting. Although the image quality looks acceptable, the color distributions of the balls are not at all directed towards the origin, as would have been the case if black was calibrated at $\vec{C}_{RGB0} = [0, 0, 0]^T$.

For specific $\xi$ and $\theta$, the dimensionality of the appearance model can be reduced from three to two dimensions by projecting the RGB values $\vec{C}_{RGB}$ onto a plane perpendicular to $\vec{l}(x, \xi, \theta)$:

$$\vec{C}_S = [\hat{S}_1, \hat{S}_2]^T \vec{C}_{RGB}.$$  \hspace{1cm} (5.3)

$[\hat{S}_1, \hat{S}_2, \hat{S}_3]$ is the orthonormal basis of the adapted color space, which is a rotated RGB coordinate system with $\hat{S}_3$ in the direction of $\vec{l}(x, \xi, \theta)$, corresponding to the luminance axis, and $[\hat{S}_1, \hat{S}_2]$ spanning the perpendicular plane. The latter can be seen as an intensity-invariant chrominance space for one specific object color in a specific situation (camera, calibration, illumination, etc.), referred to as 'Adaptive Chrominance Space' (ACS). The distribution of an object color in ACS will be modeled by a Gaussian distribution with mean $\mu_S$ and $2 \times 2$ covariance matrix $\Sigma_S$. The Mahalanobis distance $D_S$ to the closest point on the line $\vec{l}(x, \xi, \theta)$ is computed by

$$D_S = \sqrt{(\vec{C}_S - \mu_S)^T \Sigma_S^{-1} (\vec{C}_S - \mu_S)}.$$  \hspace{1cm} (5.4)
Because of the Gaussian approximation perpendicular to the luminance axis, the skin color model becomes an infinite elliptic cylinder in RGB space. To save computational load, $\hat{S}_1$ and $\hat{S}_2$ can be chosen in the directions of the eigenvectors of $\Sigma_S$ and divided by the square roots of the respective eigenvalues, leaving the 2x2 identity matrix instead of $\Sigma_S$.

### 5.3.2 Skin Likelihood

Computation of skin likelihood at image position $x$ is performed according to Bayes’ theorem:

$$p(\text{skin}|\vec{C}_{RGB}) = \frac{\int p(\vec{C}_{RGB}|\text{skin})p(\text{skin})}{p(\vec{C}_{RGB})}$$  \hspace{1cm} (5.5)

Where $\text{skin}$ is the event that the image really contains skin at the measured location.

The prior probability density of $p(\vec{C}_{RGB})$ can be marginalized by

$$p(\vec{C}_{RGB}) = \int_{-\infty}^{\infty} p(\vec{C}_{RGB}|I = \alpha)p(I = \alpha)\delta(\alpha - I)d\alpha = p(\vec{C}_{RGB}|I = \beta)p(I = \beta),$$  \hspace{1cm} (5.6)

because $p(\vec{C}_{RGB})$ is zero for $\alpha \neq I$. $\beta$ is equal to the light intensity of ($\vec{C}_{RGB}$), calculated by

$$I = (C_R - C_{R0}) + (C_G - C_{G0}) + (C_B - C_{B0}).$$  \hspace{1cm} (5.7)

Since the real value of black $\vec{C}_{RGB0}$ is unknown, we approximate it by

$$\vec{C}_{RGB0} = \vec{C}_{G0} = \vec{C}_{B0} = \min\{C_R(X) \cup C_G(X) \cup C_B(X)\}$$  \hspace{1cm} (5.8)

where $C_R$, $C_G$, $C_B$ are the red, green and blue values, respectively, of $\vec{C}_{RGB}$ and $X$ is the total set of pixels available, measured in the specific situation. Here we assume that the black origin is on the diagonal of RGB space, which is a reasonable assumption considering the imaging processes of most digital cameras. Unfortunately, in most situations the real value of $\vec{C}_{RGB0}$ is negative and the lower color values are clipped to 0, resulting in a large estimation error in $\vec{C}_{RGB0}$.

The probability density of the distance to the RGB diagonal at the brightness plane $C_R + C_G + C_B = I$ is assumed constant and non-zero inside positive RGB space, but zero outside (uniformly distributed saturation). The integral of $p(\vec{C}_{RGB}|I = \beta)$ over all possible values of $\vec{C}_{RGB}$ for which $I = \beta$ must be equal to 1. This set is a plane perpendicular to the diagonal of RGB space, which is an equilateral triangle with its corners at $\vec{C}_{RGB} = \{[0, 0, \beta]^T, [\beta, 0, 0]^T, [0, \beta, 0]^T\}$. Using the area $A(\beta)$ of this triangle and assuming that the prior probability of brightness $p(I = \beta)$ is constant,

$$p(\vec{C}_{RGB}) \propto 1/A(\beta) = 2/(\sqrt{3}\beta^2).$$  \hspace{1cm} (5.9)

We choose to model uncertainty about the actual color of skin $\xi$, but to neglect the additive noise $\vec{η}(x)$. This results in a conical shape with its tip at $\vec{C}_{RGB0}$, centered
5.3. Color Detection Method

around \( \vec{\ell}(x, \xi, \theta) \) corresponding to the mean \( \xi \) and \( \theta \), shown in figure 5.2. In this case, deviation from the line \( \vec{\ell}(x, \xi, \theta) \) is only due to variation of skin or illumination color. The existing appearance model can easily be modified by normalizing the Mahalanobis distance of equation 5.4 by the brightness \( I \), similar to computing normalized rgb:

\[
\hat{D}_S(x) = \frac{D_S}{I}.
\]  

(5.10)

When \( p(\vec{C}_{RGB}|\text{skin}) \), in equation 5.5, is computed with \( \hat{D}_S \) instead of \( D_S \), the prior (equation 5.9) has to be normalized by \( 1/I^2 \), hence becomes a constant. The log likelihood becomes:

\[
\ln\{p(\text{skin}|\vec{C}_{RGB})\} \propto -\frac{(\vec{C}_S - \mu_S)^T \Sigma^{-1}_S (\vec{C}_S - \mu_S)}{I^2}.
\]  

(5.11)

Note that, if \( \vec{C}_{RGB0} = [0, 0, 0]^T \) (i.e. in case of correct calibration), this approach is very similar to a skin color model in Hue/Saturation space or two dimensions of the normalized rgb space.

5.3.3 Automatic Model Fitting

The automatic Adaptive Chrominance Space adaptation procedure attempts to find the best fit of the model to a sample set taken from a specific situation. To obtain positive RGB skin samples for model fitting, a face detection method could be used. Pixels from the left and right side of the sample area are modeled separately. Resulting in two separate ACS models. This is because the left and right side of the face are often illuminated with a different \( \theta \), causing a different \( \vec{\ell}(x, \xi, \theta) \). By modeling both sides of the face separately, the combined model can detect skin illuminated by different light sources.

Skin samples for which at least one of the color channels has an intensity of 0 or 255 are removed. This is because these samples might have been projected from outside the RGB space, leading to a skewed color representation. Note that modeling these projected values is possible, as described in our previous work [12], however, it is beyond the scope of this work.

The fitting procedure is based on the assumption that the main axis of the positive sample distribution corresponds to \( \vec{\ell}(x, \xi, \theta) \). This is usually the case, since the area of the face, used for sampling, contains both dark and light skin values (e.g. due to the nose and its curved shape). Even in case of very uniform illumination, some shading will still be present. Otherwise, one would not see any shape of the face other than the eyes and mouth of a person. These small nuances of skin color are enough to show a main axis in the direction of intensity in RGB space. The main axis is found by line fitting, using RANSAC [13]. However, the search of RANSAC is constrained by assumptions about shadowed colors, following [12]. Pairs of points for which one of the two can never be a shaded color of the other, are discarded. The regular least square fitting method does not work because of the inhomogeneous distribution of samples along the line. Therefore, the best result of the RANSAC
5. A Self-Calibrating Chrominance Model Applied To Skin Color Detection

Figure 5.3: Some images used for the experiments. The area inside the white rectangle was used for modeling skin color.

search is refined by computing the means $m_L$ and $m_H$ of two subsets of the inlier samples: First, the samples are projected on the initially fitted line. $m_L$ is the mean of the samples with the lowest 0.1 quantile of projection values, $m_H$ is computed from samples in the highest 0.1 quantile. From the line between $m_L$ and $m_H$, $\hat{S}_3$ is derived and two perpendicular vectors $\hat{S}_1$ and $\hat{S}_2$ can be determined. After projecting the positive samples onto the AC space spanned by $[\hat{S}_1, \hat{S}_2]$, the mean and covariance of the distribution are estimated by a fixed number of EM iterations, where the 0.9 best fitting samples are used to compute the parameters of the next iteration. The latter is to discard outliers.

To get a combined likelihood of the ACS model for the left and the right side of the face, the two are combined by taking the maximum of the two log-likelihoods, computed with normalized inverse covariance matrices $\hat{\Sigma}_S^{-1}$:

$$\hat{\Sigma}_S^{-1} = \frac{\Sigma_S^{-1}}{\sqrt{|\Sigma_S^{-1}|}}. \quad (5.12)$$

The measured main axis of the skin color distribution is not always the result from the appearance model in equation 5.1. For example, when multiple light sources are present, the transition from one light source to the other could contain correlation between $I(x)$ and $\theta$. In these situations, there is no guarantee that the appearance model will continue in the same direction, outside of the intensity range of the training samples. In order to be more robust against these exceptions, also a more conservative Hybrid ACS (HACS) method is considered. This model is split in three parts: a middle part for $I(m_L) < I < I(m_H)$, identical to the normal ACS model, and a lower and higher part, for $I < I(m_L)$ and $I > I(m_H)$, respectively, that assume that $\vec{\ell}(x, \xi, \theta)$ continues from the ends of middle part into the directions corresponding to the estimated $\hat{C}_{RGB0}$ from equation 5.8, but using the same covariance as the middle part.
Table 5.1: Area Under the Curve of the ROC for FP rate \( \leq 0.1 \).

<table>
<thead>
<tr>
<th>Image</th>
<th>HS</th>
<th>rgb</th>
<th>CrCb</th>
<th>RGB</th>
</tr>
</thead>
<tbody>
<tr>
<td>Studio</td>
<td>0.099</td>
<td>0.094</td>
<td>0.094</td>
<td>0.094</td>
</tr>
<tr>
<td>Office</td>
<td>0.095</td>
<td>0.090</td>
<td>0.084</td>
<td>0.084</td>
</tr>
<tr>
<td>Hallway</td>
<td>0.055</td>
<td>0.047</td>
<td>0.056</td>
<td>0.054</td>
</tr>
<tr>
<td>Sunset</td>
<td>0.063</td>
<td>0.056</td>
<td>0.063</td>
<td>0.063</td>
</tr>
<tr>
<td>Restaurant</td>
<td>0.085</td>
<td>0.072</td>
<td>0.080</td>
<td>0.079</td>
</tr>
<tr>
<td>Window</td>
<td>0.076</td>
<td>0.084</td>
<td>0.080</td>
<td>0.077</td>
</tr>
<tr>
<td>Living room</td>
<td>0.078</td>
<td>0.057</td>
<td>0.057</td>
<td>0.057</td>
</tr>
<tr>
<td>Glass wall</td>
<td>0.071</td>
<td>0.082</td>
<td>0.082</td>
<td>0.082</td>
</tr>
<tr>
<td>Table room</td>
<td>0.065</td>
<td>0.061</td>
<td>0.060</td>
<td>0.060</td>
</tr>
<tr>
<td>Average</td>
<td>0.076</td>
<td>0.071</td>
<td>0.071</td>
<td>0.071</td>
</tr>
<tr>
<td>Deviation</td>
<td>0.009</td>
<td>0.010</td>
<td>0.010</td>
<td>0.010</td>
</tr>
</tbody>
</table>

5.4 Experiment

We have evaluated our ACS method by applying it to 10 different photos of different persons in different environments (see figure 5.4). Most photos were taken with a digital photo camera, except for ‘Studio’, which was captured with a firewire video camera. This camera was calibrated to have \( \vec{C}_{\text{RGB}0} = [0, 0, 0]^T \) and homogeneous illumination was used (i.e. ideal circumstances for most color models). The photo cameras used for the other photos automatically adjusted white balance and intensity for each photo. All photos were filtered using a 3x3 median filter, to reduce noise and erroneous colors around sharp edges. A rectangle was manually annotated at each face, to extract the training color samples. These rectangles are shown in the photos of figure 5.4. The left and right sides of these rectangles were used for training the left and right ACS models, respectively. To remove remaining non-skin pixels from the training samples (e.g. from eyes and mouth), the training is done in two steps. After training the initial ACS models on all respective training pixels, the pixels with the lowest 30% of likelihoods for the respective ACS model are removed from the rectangle and a 3x3 morphological closing is applied on the total remaining pixel mask. The ACS models are re-trained using only the samples selected with this skin mask. For a fair comparison, also the other color models are trained on these filtered samples.

Training and testing of the skin color model was done separately for each image, to evaluate the average performance of ACS over different circumstances. The true positive (TP) rate (number of pixels correctly detected as skin divided by the total number of tested skin pixels) is computed only from the bare skin of the hands and arms in the same image, which was also manually annotated. The false positive (FP) rate (number of pixels wrongly detected as skin divided by the total number of tested non-skin pixels) was computed from all areas in the same photo not containing any bare skin or hair. Note that, although training and test samples were extracted from the same image, they correspond to physically different parts of the scene, subject to possibly different illumination and captured with different parts of the camera sensor. This results in a better separation between training and test set than when they only differ in time.

Four different color spaces were compared to ACS: Hue-Saturation (HS) space, normalized Red-Green (rg) space, CrCb space and RGB space. Histograms of the face samples were used as the skin likelihood models. For the first three color spaces, the
histogram sizes are $100 \times 100$ bins and for RGB space the histogram was $100 \times 100 \times 100$ bins. The reason for using histograms is because histograms do not assume any type of distribution. Because histograms do not generalize as well as parameterized distributions, the histograms are all smoothed with a Gaussian kernel of 21 bins in all dimensions, with standard deviation 1. HACS and HSV color models and skin detection for the image 'Window' are shown in figure 5.4.

To get an objective and comparable measurement of performance, the Area Underneath the Curve (AUC) is computed for the Receiver Operation Characteristic (ROC) curves of all methods and photos. Only the part of the curve with small FP rate is relevant, since too many false positives will make it impossible to separate hands from the rest of the image. Therefore, only the AUC for FP rates between 0 and 0.1 is used for comparison. The results are shown in table 5.1 (the closer to 0.1, the better). The ACS method shows the best mean AUC, however, HACS has the best average ranking over the test images. Furthermore, these results show that the relative performance between all color models greatly depends on the situation. The low standard deviation of the HACS results indicates the high robustness of the method. Although the other methods have higher results in some specific images, they also have significantly lower results for other images. Surprisingly, the RGB model outperforms the HS, rg and CrCb models, contrary to what could be expected from modeling skin color with prior knowledge about intensity invariance. The violation of the assumptions of these models in everyday situations clearly has a negative effect on their performance.

### 5.5 Conclusions and Future Work

We have proposed an adaptive chrominance model and automatic fitting procedure that can be used to detect skin color more accurately than the compared methods when no color space calibration is performed and/or heterogeneous illumination is present. This makes our (H)ACS model especially useful for real-world applications. Besides a better overall skin detection performance, HACS also showed a lower standard deviation between different situations, while the other methods showed more unstable results.

Further improvements of the model are possible. First of all, over- or under-saturated colors can be accounted for and assumptions about the model orientation and shape outside of the intensity range of the training samples can be improved. Furthermore, many improvements are possible on the prior probability model of background color. A histogram of the complete image and/or an off-line image database could be used to exclude colors with a high prior probability.
5.5. Conclusions and Future Work

Figure 5.4: Results for the image ‘Window’. The upper figure shows detection of the main axes of the left (black dots, solid line) and right (blue dots, dashed line) side of the face. All hulls in RGB space correspond to skin color using a threshold that detects 75% of the training (face) samples.
Bibliography


Chapter 6

3D versus 2D Pose Information for Recognition of NGT Signs

This chapter is based on the paper published as “3D versus 2D pose information for recognition of NGT signs”, by J. F. Lichtenauer, E. A. Hendriks and M. J. T. Reinders in the Proc. 27th Symposium on Information Theory in the Benelux, June 8-9 2006.
Abstract In this article, we evaluate the improvement in sign recognition that can be achieved with 3D tracking, compared to recognition with image plane tracking. Experiments are shown using a pair of stereo cameras, from which 3D positions of the segmented hands are estimated by triangulation between the left and right cameras. A sign classifier is trained on a set of 37 different NGT signs using 2D and 3D features. The results show that 3D features improve sign detection results, especially when 3D features are only used when their relevance is known beforehand. Furthermore, we show the positive effect of perspective distortion at close range.

6.1 Introduction

Recently, hand gesture recognition has gained a lot of interest as a means for Human Computer Interaction (HCI). Hand gestures are believed to be an important human communication modality and are, therefore, essential for natural HCI. We are developing an electronic learning environment (ELo), for learning a vocabulary of signs of the Dutch sign language (NGT), in collaboration with the Dutch Foundation for the Deaf and Hard of Hearing Child (NSDSK) [1]. To provide feedback to the child about its performance of producing a sign, a visual detection system is necessary that distinguishes between a specific, correctly performed sign and any other gesture.

![Figure 6.1: The Electronic Learning Environment ELo.](image)

The setup is schematically shown in figure 6.1. The user sits in front of a table with a touch screen and a camera. The camera has a wide angle lens (100 degrees) placed at eye-level, and the touch screen is slanted under an angle that prevents the screen from occluding the hands to the camera.

The system requirements are: 1) It must work with different and mixed lighting sources and backgrounds; 2) Person independence. The users are children making the signs for the first time; 3) Immediate response; 4) Adjustable ‘strictness’, to adapt to the experience level of a user; 5) Invariance to valid sign variation. A lot of NGT signs contain periodic hand motion or shape change, e.g. repetitive rotations, moving hands repeatedly up and down or left and right. The number of periods, as well as the starting and ending positions or shape of the hands is not defined for these signs.

With this in mind, we have developed a sign detection method that is robust to inter personal variance, adapts itself to the lighting conditions by learning skin color
6.2 Related Work

from color samples of the face, and works in real time (25 frames per second). It has a detection threshold to adjust signer accuracy and can deal with sign variations due to periodicity. To further improve the performance of the system, a second camera could be added for 3D tracking. In this article, we investigate the usefulness for sign detection of this extra depth information.

First, related work is described in section 6.2, our method is explained in section 6.3 and the results are shown in 6.4. Finally, conclusions about depth in sign detection are given in section 6.5.

6.2 Related Work

Many different methods for visual gesture recognition have been investigated. The larger part of recent work is based on Hidden Markov Models [2, 3] and some on sequential Markov Models [4] or gesture templates [5]. Most gesture recognition methods use only a single camera.

Instead of using HMMs, we have chosen a direct feature-matching approach, letting the person indicate the start and end of a sign by placing hands on the table. This makes very fast detection possible without losing accuracy.

The system that is most similar to our application is that of Lee et al. [6]. They have developed a sign language learning application for deaf children that recognizes one sign out of a set of signs. In our case, however, we want to detect if the child has produced the right sign, not the most similar of a specific set.

In our system, the hands are tracked based on skin color. For detecting skin in color images, many different color models have been proposed. Some assume a prior distribution of skin color and are trained off-line on a large database of images containing skin. Others are adaptive to the conditions [7, 8]. There are also methods using a combination of both [9]. The distributions are mostly either Gaussian (mixture) models in RGB or a 2D chrominance color space or a color histogram/lookup table. A good survey of skin detection methods is given by Vezhnevets et al. [10]. Because in our application the face is visible most of the time, face detection can be used to learn the skin color and obtain more precise and robust skin detection than possible with a prior skin model. This is done before, e.g. by Fritsch et al. [11].

Our adaptive skin detection method is different from existing methods because it models left and right face color separately with a unimodal distribution and is based on a novel color distribution model. Fitting a parameterized model generalizes better than filling a histogram with only the skin colors visible in the face.

6.3 Sign Detection Method

The complete sign detection system that we propose is shown in figure 6.2. When the user produces the sign, the hands and head are visually tracked and their features are measured. Afterwards, a classifier compares the measured features with the model for the correct sign and provides a confidence measure of correct production on which a threshold can be set to approve or disapprove the produced sign.
6.3.1 Pose Estimation

The positions of hands and head are estimated by subsequent detection of skin colored blobs and assigning them to the head and left and right hand, according to their size and position.

Adaptive skin color modeling: To determine how skin color appears under specific conditions (camera settings and lighting), the skin model is (re-)determined by a separate initialization procedure. Using face and motion detection, positive and negative skin examples are sampled from the video.

Skin detection: Skin detection is performed according to [12]. Skin color is modeled by a 2D Gaussian perpendicular to the main direction of the distribution of the positive skin samples in RGB space, which is found by RANSAC. During experiments, the distribution of skin pixels often turned out to be bimodal, caused by multiple light sources (e.g., daylight through the window together with interior lighting and/or reflection). Therefore, we modeled both modalities separately instead of assuming unimodality. Since the difference of illumination direction is mainly due to the surface longitude of the face, the two skin color modalities can be largely separated by modeling the color samples of the left and right side of the face separately. A more detailed description of this color model can be found in [12]. Skin is detected by hysteresis thresholding skin color dissimilarity. For robustness reasons, the high and low thresholds of both color models are made dependent on fractions of the dissimilarities of the positive and negative skin samples. The final skin pixels are found by applying a morphological closing on the union of positive detections with the left and right face color models.

Figure 6.2: Sign detection algorithm. (a) Block diagram, (b),(c) Skin segmentation and hand tracking example.
6.3. Sign Detection Method

Start/end detection: Knowing the start and end time of a produced sign makes it possible to be invariant for gesture duration. The motion between start and end can be normalized by warping it to the same length as a reference model. With the current system the sign has to be started and ended explicitly by putting both hands on the table. A moving hand is detected by thresholds on the total number of skin pixels and the number of skin pixels in motion in the image areas where the hands are expected to be put on the table.

Tracking: Tracking the hands and head is done by following their respective blobs over consecutive frames. They are initialized by size and position using the three largest skin blobs in the image and tracked by finding the nearest blob in the next frame. During occlusion with each other, left and right hand share the same blob until they split again. The left and right blob are assigned to the left and right hand, respectively, which means crossing hands will confuse tracking. During occlusion with the head, the last free position of a hand blob is maintained until a new blob re-appears close to this position and is assigned to the lost hand. The blobs are re-initialized when there is occlusion during more than 20 consecutive frames.

Depth: For 3D hand localization, the skin-based hand tracking is performed in both frames from a stereo camera pair. The found pixel locations of the hands and head are mapped onto a virtual camera model consisting of two pinhole lenses that are completely parallel. Using this calibrated model, the distance of the hands and head are estimated by triangulation.

6.3.2 Classification

A detector for an electronic sign learning application must be able to detect correct production of a sign, even when the exact motion trajectory can vary in case of periodicity. Pose features, related to the shape, movement and change of shape of the hand and head blobs can be used for detection. These properties are measured in each frame. The mean position of the head over the non-occluded frames of the gesture is used as the origin for the positions of the hands.

Synchronization: First, the features are linearly warped to a normalized length of 100 time frames using nearest-neighbor interpolation. The time-derivative features are scaled accordingly. Subsequently, a local feature synchronization (LFS) is applied to the gestures by minimizing the feature distance to a reference sign over a time window of $2w + 1$ frames:

$$\hat{F}_q(f, t) = F_q(f, \tau(t))$$
$$\tau(t) = \arg \min_{j \in [-w, t+w]} |F_r(f, t) - F_q(f, j)|$$

Where $F_r(f, t)$ and $F_q(f, t)$ are the feature values of the time-normalized reference and query sign, respectively, for feature type $f$ at time frame $t$.

Training: For each sign $s$, a two-class classifier is trained on a set of positive and negative (other signs) examples. First, a reference sign (prototype) is selected for the respective classifier. The reference sign is not used for training, but only for synchronization. Because accurate tracking is often hindered by occlusions, the sign with the lowest number of frames with occlusion is used as the reference sign.
For each recorded sign, the total number of features equals the number of measured properties \( \times \) 100 frames. Because of this large number each feature is independently modeled by a 1D Gaussian with mean \( \mu(s, f, t) \) and standard standard deviation \( \sigma(s, f, t) \), estimated from the positive training set. For each time-conditional feature, a weak classifier is constructed by setting a threshold at \( \lambda_w = \eta_f \sigma(s, f, t) \) on \( |\hat{F}_q(f, t) - \mu(s, f, t)| \). Noisy features are excluded by a threshold on their false positive rate on the negative training set. The weak classifiers are combined by their total rate of positive detections \( \nu(F_q, s) \). The threshold \( \lambda_t(s) \) for sign detection with a classifier trained on one person is derived from the desired false positive rate. Because it is possible to make an NGT sign both left- or right-handed, also the mirrored features \( \nu(F'_q, s) \) are classified. After normalization with \( \lambda_t(s) \), the classification ‘score’ becomes:

\[
\hat{\nu}(F_q, s) = \max(\nu(F_q, s), \nu(F'_q, s)) / \lambda_t(s).
\] (6.3)

The total pre-processing and classification of one sign takes 70 milliseconds on a 1.5GHz pentium pc.

![Figure 6.3: Experimental setup and results. (a) The stereo camera pair is placed at a large baseline of 0.5m, at 1.3m from the table edge. (b) The effect of perspective distortion on sign detection, tested by varying the distance of a virtual frontal pinhole camera.](image)

### 6.4 Experiments

Our sign detection method, using the features shown in table 6.1 and a LFS window \( w = 15 \), is applied to a set of 37 different NGT signs, each performed 8 times by a single person. Hence, the total data set size is 296 sign examples. The images are captured at 640 by 480 pixels at 25 frames per second from both cameras and down-scaled to 160 by 120 pixels for real-time skin color segmentation. The experiment
Table 6.1: Features used for sign classification

<table>
<thead>
<tr>
<th>left &amp; right hand 2D:</th>
<th>left &amp; right hand 3D:</th>
</tr>
</thead>
<tbody>
<tr>
<td>horizontal position relative to head</td>
<td>forward horizontal distance from head</td>
</tr>
<tr>
<td>vertical position relative to head</td>
<td>forward horizontal velocity</td>
</tr>
<tr>
<td>horizontal velocity</td>
<td>forward horizontal velocity</td>
</tr>
<tr>
<td>vertical velocity</td>
<td></td>
</tr>
</tbody>
</table>

consisted of an 8-fold cross validation: each of the eight examples is used as the positive test sign and the other 7 for training. The training and test examples of the remaining 36 signs are used as the negative training and test examples, respectively, for each sign.

In this way, for each sign in each cross validation step, a separate sign-specific two class classifier (a sign detector) is trained, from which the area under the ROC curve (AUC) can be estimated as a performance measure that can be averaged over all experiments. These values are shown in table 6.2. Because only a few (or even zero) false positives occurred for true positive rates smaller than one, the AUC values are not a very reliable measure. Therefore, we have also compared the average likelihoods of the positive signs to those of the negative signs. A reliable detector should have a high ratio between positive and negative likelihoods.

Three different feature types are compared: 3D features, where the perspective distortion is completely corrected and depth is included; 2D features, identical to the 3D features except for leaving out the depth; and 2D features that contain perspective camera distortion. For the last features, the result is shown in table 6.2 only for the camera distance with the best result: 35 cm from the table edge. All features are derived from the real positions, estimated from depth. In this way, the error of depth estimation has an equal effect on all features. The perspective transformations are applied to a virtual pinhole camera (wide angle of 180 degrees) positioned on a line through the middle between the stereo camera pair.

Some of the tested NGT signs have depth as an explicit part of the motion or the position of the hands. For other signs, depth is not particularly a distinguishing feature. This is indicated in the last column of table 6.2. For instance ‘treinspoor’ (railway track) and ‘paadje’ (path) consists of a significant forward motion of one or both hands, and the signs ‘mensen’ (people) and ‘schaap’ (sheep) are made touching the body, instead of in the regular sign space in front of the body. Furthermore, ‘schaap’ also contains a small repeating motion forwards. The results show that these signs benefit the most from the depth features, but the performance for other signs is sometimes even degraded because of adding redundant depth features. Because we know beforehand which signs explicitly contain depth, depth features only have to be used for these ones. The result of this method (2D/3D) (shown at the bottom of 6.2) is the best detection performance compared to all other methods.

When sign recognition is performed without depth estimation, it is important to know the consequence of perspective distortion on the recognition performance. The estimations of the real 3D locations in our experiments made it possible to convert the results to any virtual camera point. The average sign performances for a range of
Table 6.2: Results showing the average Area Under the Curve (AUC) of the ROC and the ratio between the average likelihood of the positive and negative signs

<table>
<thead>
<tr>
<th></th>
<th>2D, no perspective</th>
<th>2D, close perspective</th>
<th>3D</th>
<th>explicit depth</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>AUC</td>
<td>Relative likelihood</td>
<td>AUC</td>
<td>Relative likelihood</td>
</tr>
<tr>
<td>cijfers</td>
<td>1.000</td>
<td>1.0900</td>
<td>0.9295</td>
<td>1.7427</td>
</tr>
<tr>
<td>dinourmet</td>
<td>1.000</td>
<td>1.6314</td>
<td>1.000</td>
<td>1.2891</td>
</tr>
<tr>
<td>erwet</td>
<td>1.1000</td>
<td>1.4713</td>
<td>1.000</td>
<td>1.6899</td>
</tr>
<tr>
<td>erziet</td>
<td>1.1000</td>
<td>1.2670</td>
<td>1.000</td>
<td>1.2855</td>
</tr>
<tr>
<td>gudokus</td>
<td>1.0000</td>
<td>2.0390</td>
<td>1.000</td>
<td>2.2317</td>
</tr>
<tr>
<td>huppe</td>
<td>1.0000</td>
<td>1.5334</td>
<td>1.000</td>
<td>1.8893</td>
</tr>
<tr>
<td>is</td>
<td>0.9560</td>
<td>1.2842</td>
<td>1.000</td>
<td>1.7359</td>
</tr>
<tr>
<td>kashel</td>
<td>1.0000</td>
<td>2.1425</td>
<td>1.000</td>
<td>2.2812</td>
</tr>
<tr>
<td>lavage</td>
<td>0.9560</td>
<td>1.2943</td>
<td>1.000</td>
<td>1.6143</td>
</tr>
<tr>
<td>kool</td>
<td>1.0000</td>
<td>1.7128</td>
<td>1.000</td>
<td>2.3990</td>
</tr>
<tr>
<td>kookbeek</td>
<td>0.9565</td>
<td>1.3290</td>
<td>0.9826</td>
<td>1.9115</td>
</tr>
<tr>
<td>kule</td>
<td>1.0000</td>
<td>2.1806</td>
<td>1.000</td>
<td>1.9779</td>
</tr>
<tr>
<td>kitchen</td>
<td>1.0000</td>
<td>2.1345</td>
<td>1.000</td>
<td>2.3984</td>
</tr>
<tr>
<td>lipi</td>
<td>0.9560</td>
<td>1.0049</td>
<td>1.000</td>
<td>2.3176</td>
</tr>
<tr>
<td>maturas</td>
<td>1.0000</td>
<td>1.7872</td>
<td>1.000</td>
<td>1.8458</td>
</tr>
<tr>
<td>mepursumus</td>
<td>1.0000</td>
<td>1.2890</td>
<td>1.000</td>
<td>1.2902</td>
</tr>
<tr>
<td>melk</td>
<td>0.9560</td>
<td>1.0049</td>
<td>1.000</td>
<td>1.5130</td>
</tr>
<tr>
<td>mezen</td>
<td>0.9565</td>
<td>1.7273</td>
<td>0.9565</td>
<td>1.7139</td>
</tr>
<tr>
<td>mus</td>
<td>1.0000</td>
<td>1.3867</td>
<td>0.8438</td>
<td>1.2222</td>
</tr>
<tr>
<td>paude</td>
<td>0.9560</td>
<td>1.5346</td>
<td>0.9565</td>
<td>2.5746</td>
</tr>
<tr>
<td>puzzel</td>
<td>1.0000</td>
<td>1.4745</td>
<td>0.9826</td>
<td>1.9887</td>
</tr>
<tr>
<td>mupraper</td>
<td>0.9560</td>
<td>1.0341</td>
<td>1.000</td>
<td>1.8672</td>
</tr>
<tr>
<td>schaud</td>
<td>0.9560</td>
<td>1.3899</td>
<td>1.000</td>
<td>1.7612</td>
</tr>
<tr>
<td>schulaw</td>
<td>0.9560</td>
<td>1.7602</td>
<td>0.9565</td>
<td>1.7862</td>
</tr>
<tr>
<td>schulit</td>
<td>1.0000</td>
<td>1.6321</td>
<td>1.000</td>
<td>1.8389</td>
</tr>
<tr>
<td>spegelleest</td>
<td>1.0000</td>
<td>2.8300</td>
<td>1.000</td>
<td>2.1221</td>
</tr>
<tr>
<td>weightroom</td>
<td>1.0000</td>
<td>1.2899</td>
<td>1.000</td>
<td>2.7980</td>
</tr>
<tr>
<td>stel</td>
<td>1.0000</td>
<td>1.4084</td>
<td>1.000</td>
<td>1.7399</td>
</tr>
<tr>
<td>saint</td>
<td>1.0000</td>
<td>1.7773</td>
<td>1.000</td>
<td>1.7599</td>
</tr>
<tr>
<td>sake</td>
<td>1.0000</td>
<td>2.0384</td>
<td>1.000</td>
<td>2.1811</td>
</tr>
<tr>
<td>weלוטsחט</td>
<td>1.0000</td>
<td>2.0817</td>
<td>1.000</td>
<td>2.4013</td>
</tr>
<tr>
<td>viscer</td>
<td>1.0000</td>
<td>2.0093</td>
<td>1.000</td>
<td>2.2309</td>
</tr>
<tr>
<td>vezugm</td>
<td>0.9560</td>
<td>1.7798</td>
<td>0.9565</td>
<td>1.7934</td>
</tr>
<tr>
<td>web</td>
<td>1.0000</td>
<td>1.4794</td>
<td>0.9757</td>
<td>1.4224</td>
</tr>
<tr>
<td>wak</td>
<td>1.0000</td>
<td>1.7223</td>
<td>0.9565</td>
<td>2.0218</td>
</tr>
<tr>
<td>school</td>
<td>1.0000</td>
<td>1.7979</td>
<td>0.9565</td>
<td>1.7981</td>
</tr>
<tr>
<td>tiberon</td>
<td>0.9722</td>
<td>1.8408</td>
<td>1.000</td>
<td>1.8779</td>
</tr>
<tr>
<td>average</td>
<td>0.9562</td>
<td>1.7594</td>
<td>0.9587</td>
<td>1.9214</td>
</tr>
</tbody>
</table>
virtual distances are plotted in figure 6.3. These results show that the distortion has little influence when the camera is placed further than one meter. When the camera is placed closer, the performance increases, especially for signs with explicit depth. This is probably because the perspective distortion causes depth information to be present in the 2D features. However, although the highest ratio of likelihoods (at 35cm) is even better than using 3D features only, the average AUC is lower (see table 6.2). This might be because the standard deviation of likelihoods is also higher. Furthermore, a camera placed so close will probably be very sensitive to a change in the position of the person.

6.5 Conclusions

We have compared two dimensional to three dimensional feature sets to detect NGT sign language signs. We found out that the extra depth dimension results in improved results for signs where motion in the third dimension is an explicit distinguishing property of the sign. However, this advantage is not always present for signs where depth is not an explicit property. When depth is not relevant, the extra redundant depth features can even decrease detection performance. When depth features are only used for detection of signs with explicit depth, the overall improvement is even larger. We also found out that when the camera is placed closer than one meter, the perspective distortion can actually improve detection performance with 2D features. At close proximity, the projective imaging transformation has the consequence that depth information is present in the measured 2D features.
Bibliography


Chapter 7

Sign Language Recognition by combining Statistical DTW and Independent Classification

©2008 IEEE. Personal use of this material is permitted. However, permission to reprint/republish this material for advertising or promotional purposes or for creating new collective works for resale or redistribution to servers or lists, or to reuse any copyrighted component of this work in other works, must be obtained from the IEEE.

Abstract To recognize speech, handwriting or sign language, many hybrid approaches have been proposed that combine Dynamic Time Warping (DTW) or Hidden Markov Models (HMM) with discriminative classifiers. However, all methods rely directly on the likelihood models of DTW/HMM. We hypothesize that time warping and classification should be separated because of conflicting likelihood modeling demands. To overcome these restrictions, we propose to use Statistical DTW (SDTW) only for time warping, while classifying the warped features with a different method. Two novel statistical classifiers are proposed (CDFD and Q-DFFM), both using a selection of discriminative features (DF), and are shown to outperform HMM and SDTW. However, we have found that combining likelihoods of multiple models in a second classification stage degrades performance of the proposed classifiers, while improving performance with HMM and SDTW. A proof-of-concept experiment, combining DFFM mappings of multiple SDTW models with SDTW likelihoods, shows that also for model-combining, hybrid classification can provide significant improvement over SDTW. Although recognition is mainly based on 3D hand motion features, these results can be expected to generalize to recognition with more detailed measurements such as hand/body pose and facial expression.

7.1 Introduction

Time-variable signals like speech, handwriting, hand-gestures and body movements cannot be compared in a Euclidean space directly, because of misalignments in time. Therefore, automatic recognition of these signals is not straightforward. In recent years, successful methods for speech recognition have been thankfully borrowed and adapted for sign language recognition. However, this has been done without questioning the exact linguistic role of the dynamics in sign language, or possible conflicts between optimality in time-synchronization and class-discrimination. Therefore, in this article, we explore the consequences and benefits of separating time-synchronization from classification in sign language recognition. The downside of this separation is that any information about relative timings is lost. The advantage of separate classification on synchronized features is that it allows the use of standard classification methods with possibly higher discriminative performance.

Dynamic Time Warping (DTW) and Hidden Markov Models (HMM) are two methods that simultaneously align signals and compute a likelihood of similarity. Therefore, they both have been applied successfully to recognize speech [1, 2, 3, 4], online [5] or off-line handwriting. Currently, they are also the mostly used methods for recognition of gestures [6, 7, 8, 9]. Over the years, DTW has lost some interest because HMM is able to statistically model a set of samples to generalize better, while DTW is an exemplar-based matching procedure, hence usually requires matching with a plurality of prototypes to get comparable performance, resulting in a higher computational load. Recently, however, Bahlmann and Burkhardt have shown that also DTW can be applied to train a statistical model, using ‘Statistical Dynamic Time Warping’ (SDTW) [5], achieving higher performance than HMM.

Since Bahlmann and Burkhardt have applied SDTW to on-line handwriting recog-
nition in [5], which can be seen as a two-dimensional gesture recognition problem, it can be expected that an improvement over HMM can also be expected when SDTW is applied to sign language recognition. Our results show that this is indeed the case. However, we further improve upon SDTW, based on our main proposition:

**Proposition 1** The maximized likelihood that results in the optimal signal warping is not the optimal conditional likelihood estimation of the signal class.

This proposition is supported by the following lemma's:

**Lemma 1.** Transition probabilities in SDTW and HMM represent prior probabilities on path shape, which is necessary for warping in case of noisy or ambiguous observation likelihoods.

**Lemma 2.** When the meaning of a sequence has invariance to time distortion, the class-conditional probability estimate of a signal should exclude path shape likelihoods.

Lemma 1 argues that transition probabilities should be applied to find the best warp of a signal, while lemma 2 implies that they should not be used in classification of signals with invariance to time-distortions. Furthermore, warping may benefit from cues that are the same for each sign, e.g. the transition from rest to movement at the onset of a sign and from movement to rest at the end. Such cues can be highly informative for warping, but completely uninformative for classification, at the same time. To reduce the dimensionality and the influence of noise, parts that are irrelevant for the meaning of a sign are often best discarded from classification.

While most spoken languages can be regarded as one-dimensional signals (sequences of audio patterns), sign languages make use of a combination of multiple cues that can be sequential (like in speech), but also parallel, consisting of different aspects/dimensions [10, 11]. The most commonly used dimensions are: hand shape/orientation, changes in hand shape/orientation, hand location, movements of hand locations, hand-hand touching, hand-body touching (mostly specific locations on the face), lip movements, facial expression and torso-/shoulder pose and -movements. Furthermore, in many cases, context is essential to uniquely define the meaning of a sign.

Regardless of which components of sign language are considered, they are all part of a dynamic process, as is speech. However, that does not necessarily mean that the dynamical aspects of sign language have the same behavior and play the same linguistic role as dynamics in spoken languages. At least three important distinctions have to be taken into account. First of all, the one-dimensionality of speech makes it sequential in nature. (Relative) timing and speed of a sequence of phonemes convey a lot of the meaning in a word. On the contrary, sign language is composed of many parallel components. Because of this richness in dimensionality, it is possible to vary speed and timing significantly without changing the message [10, 12]. Secondly, for most signs, only a subset of the degrees of freedom are important. However, this can be a different set for different signs. Furthermore, motion path features like motion orientation and curvature are not defined during a stand-still, which will result in extremely noisy values. Thirdly, the moments before the actual stroke of a sign (the preparation), after finishing the sign (retraction) or in between different signs or parts of signs (transition), are not essential for recognition. These parts are either irrelevant or redundant.
7. Sign Language Recognition by combining Statistical DTW and Independent Classification

[13]. However, they cannot simply be detected and excluded like silences in speech or pen-off periods in handwriting. The above can be summarized by the following observations:

**Observation 1.** Because of the high dimensionality of sign language, time is relatively less important for the meaning of a sign than it is for a spoken word or written letter.

**Observation 2.** Signs in sign language are defined on (different) subsets of a person’s degrees of freedom and can vary greatly on the other dimensions without any change of meaning.

**Observation 3.** Preparations, retractions and transitions in sign language cannot be removed beforehand, unlike silences in speech or pen-off periods in handwriting.

If Observation 1 is true, it may result in larger deviations of gesture speed and timing, and it also implies that the consequences of Lemma 2 have to be considered. Observation 2 implies that sign-specific feature selection would be necessary for good discrimination, which would have to be done after synchronization, just like removal of the irrelevant or redundant segments indicated by observation 3.

The important consequence of proposition 1 is that warping and classification of time-variable signals should be regarded as two distinct problems, instead of naively incorporating it into one integral Bayesian model. Observation 1 to 3 imply that this holds in particular for sign language. Therefore, we use SDTW only to warp a signal onto a reference model, and regard the time-normalized signal as a fixed-size feature set. To remove irrelevant and redundant parts and dimensions, we apply robust statistics to select only discriminative features. The proposed method is computationally attractive, as time warping is solved by dynamic programming, and the classification step is even significantly less costly. Our experiments are limited to hand motion trajectories and apparent hand-size change in isolated signs. This is because these are the few components that the current state of the art in human motion analysis allows to track in reasonably soft-constrained situations without manual initialization of tracking. Therefore, they are currently the most relevant properties for practical applications. We assume that if information about (relative) timing can be disregarded for classification when only these properties are used, this will certainly be the case if even more parallel aspects (e.g. detailed hand/body pose and facial expression) are considered, which would be inevitable to obtain perfect recognition [14].

7.2 Related Work

We are not the first to combine a variable-time signal match, like DTW/HMM, with fixed-vector-size mappings or classifiers, in order to improve results. Previous approaches can be roughly divided into methods that apply DTW/HMM on mappings of the fixed-size measurement vectors of all time-frames (to get a more informative observation likelihood) and methods that use the results of a fixed number of different DTW/HMM evaluations as the input of a second-stage classifier. In [15] a Multi Layer Perceptron provides estimates of the emission probabilities for all phonemes of speech, subsequently used for matching a HMM. In [16] a Neural Network classifies the measurements of separate frames into a first and second guess of a speech phoneme, and a DTW match uses the phoneme matches with a template word as a
7.3. Statistical Dynamic Time Warping

Statistical Dynamic Time Warping (SDTW) was first introduced in [5]. A description of Dynamic Time Warping is given in section A.3 of appendix A. DTW compares each test signal \( t = [t_1, \ldots, t_N] \) to a stored reference \( r = [r_1, \ldots, r_N] \). The difference between SDTW and normal DTW is that, instead of comparing a test signal \( t \) to a reference signal \( r \), the reference \( \mathcal{R} = [\mathcal{R}_1, \ldots, \mathcal{R}_N] \) in SDTW is not a signal but a statistical model consisting of a Normal distribution for each time point \( j \) with mean \( \mu_j \) and covariance matrix \( \Sigma_j \) and transition probabilities \( \alpha_j(\Delta\phi) : \mathcal{R}_j = \{ \mu_j, \Sigma_j, \alpha_j(\Delta\phi) \} \). \( \Delta\phi \in \mathbb{P} \) is a transition of the warping path to a point with state \( j \), where \( \mathbb{P} \) are the possible transitions from \((\phi_t(n-1), \phi_R(n-1))\) to \((\phi_t(n), \phi_R(n))\).

distance measure. In [17], the measurements for a frame of a gestured command, recorded by a camera, are converted into a probability estimate of each state, by a Radial Basis Function network. The resulting state emission probabilities are used for an HMM. In [18], Chinese sign language is measured with data gloves. Signs that are not well separated by HMM alone, are classified in an extra recognition step by a Support Vector Machine (SVM) using a DTW kernel. In [19], a sequential HMM is trained for each hand gesture measured from two cameras. The HMM match result is split into five components, which are used as features for a multi-class SVM classifier, trained by applying one HMM to all training gestures. The final classification is obtained by majority voting of the results of the HMM/SVM pairs for all gesture classes. A similar approach is chosen in [20] to classify online hand writing characters. Instead of using HMM, here SDTW is used as a kernel for SVM.

The above works confirm that results can be improved over HMM/DTW alone. However, all methods have relied directly on the likelihoods obtained from DTW or HMM. Instead, we consider DTW/HMM primarily as a registration method. We use the complete set of registered features as a richer sign representation instead of, or in addition to the outputs of HMM/DTW. This approach may even be combined with mappings of input vectors per frame as well, although this is beyond the scope of this article.

Alon et al. [21] have proposed Dynamic Space Time Warping (DSTW), which considers multiple possible 2D hand locations in each frame. This reduces the consequences of imperfect tracking. Although our experiments use single-hypothesis 3D tracking, the principle of separating warping and classification may easily be extended to DSTW. One advantage of our approach is that the negative influence of irrelevant variations in preparations, transitions and retractions can be reduced by applying feature selection on the registered feature set. Instead, Yang, Sarker and Loeding [22] have resolved this problem by including a separate model with constant distance, that is fitted to sequences that do not fit well to any known sign. The disadvantage is that it introduces the possibility of falsely inserting the transition model in the place of a sign that differs more from its model than is accounted for.

7.3 Statistical Dynamic Time Warping

Statistical DTW (STDW) was first introduced in [5]. A description of Dynamic Time Warping is given in section A.3 of appendix A. DTW compares each test signal \( t = [t_1, \ldots, t_N] \) to a stored reference \( r = [r_1, \ldots, r_N] \). The difference between SDTW and normal DTW is that, instead of comparing a test signal \( t \) to a reference signal \( r \), the reference \( \mathcal{R} = [\mathcal{R}_1, \ldots, \mathcal{R}_N] \) in SDTW is not a signal but a statistical model consisting of a Normal distribution for each time point \( j \) with mean \( \mu_j \) and covariance matrix \( \Sigma_j \) and transition probabilities \( \alpha_j(\Delta\phi) : \mathcal{R}_j = \{ \mu_j, \Sigma_j, \alpha_j(\Delta\phi) \} \). \( \Delta\phi \in \mathbb{P} \) is a transition of the warping path to a point with state \( j \), where \( \mathbb{P} \) are the possible transitions from \((\phi_t(n-1), \phi_R(n-1))\) to \((\phi_t(n), \phi_R(n))\).
7. Sign Language Recognition by combining Statistical DTW and Independent Classification

The matching cost $C^*(t, R)$ is defined by:

$$C^{(*)}_{\Phi}(t, R) = \frac{\sum_{n=1}^{N} d(t_{\phi_t(n)}, R_{\phi_R(n)})w(P(n))}{\sum_{n=1}^{N} w(P(n))}$$  \hspace{1cm} (7.1)$$

$$C^*(t, R) = C_{\Phi^{(*)}}(t, R) = \min_{\Phi} C^{(*)}_{\Phi}(t, R).$$  \hspace{1cm} (7.2)$$

Where $\Phi = \{\phi_t(1), ..., \phi_t(N_{\phi}), \phi_r(1), ..., \phi_r(N_{\phi})\}$ are the steps of the path through the 2D correspondence matrix $C(t, R)$ of the time frames of $t$ and $R$. $P(n)$ is a transition step between two subsequent points of the path $\Phi$ through $C(t, R)$ and $w(P(n))$ is a function that assigns a weight to a transition type. $C^*(t, R)$ is also used as the final match cost. Some constraints of the path are implied to confine the minimization procedure to practical results. The most common constraints are that the path starts in $\Phi(1) = (1, 1)$ and ends in $\Phi(N_{\phi}) = (N_t, N_r)$, and the set of possible transitions $P$ is limited to

$$P(n) = [(\phi_t(n + 1) - \phi_t(n)), (\phi_r(n + 1) - \phi_r(n))]$$

$$\in \{[0, 1], [1, 0], [1, 1]\},$$  \hspace{1cm} (7.3)$$

corresponding to horizontal, vertical and diagonal steps, respectively. The distance function $d(t_i, R_j)$ is defined as the inverse log likelihood:

$$d(t_i, R_j) = \frac{1}{2} \left( \ln(2\pi \Sigma_j) + (t_i - \mu_j)^T \Sigma_j^{-1} (t_i - \mu_j) \right)$$

$$- \ln(\alpha_j(\Delta \phi)),$$  \hspace{1cm} (7.4)$$

Equation 7.2 is approximated efficiently using dynamic programming, by the omission of the denominator in equation 7.1 in the choice of sub-paths. The denominator is applied only on the finally chosen path. The transition weighting function $w(P(n))$ can be chosen so that all possible sub-paths leading to one location in $C(t, R)$ have an equal sum of weights: unbiased. In that case, the approximation of equation 7.1 by dynamic programming is exact. In [4, 3] it is explained how to obtain such unbiased weighting functions. We will use the, mostly used, biased method that assigns $w = 1$ to all three transition types. Instead of a solution of equation 7.2, this weighting gives preference to more diagonal, shorter paths. According to Lemma 1, a bias towards more linear paths may actually be an advantage, as it acts as a path shape prior in case of noisy measurements.

The biased SDTW, as defined above, is equivalent to a forward HMM with self-transitions and no skips, to which ‘null-transitions’ are added. The null-transitions can be used to step to a next state (or the same state) without advancing in time. This allows unlimited compression of the model in time. The damage of a missing part can be limited to the missing part only (like one less repetition of a repetitive motion or an extremely high signing speed causing a significant reduction of time points), while with a regular HMM an observation is assigned only once to any state instance, causing the left-out part of the trained HMM to steal-away observations belonging to other, surrounding states, and limiting the amount that a sign can be compressed in time.
An SDTW model $R$ is trained on a set of examples by iteratively warping all training samples with an initial model $R$ and re-estimating each $\mu_j$, $\Sigma_j$ and $\alpha_j(\Delta\phi)$ from the aligned observations [5]. Note that, similar to a Markov Model, the transition probabilities $\alpha_j(\Delta\phi)$ at step $n$ only depend on $\phi(n)$ and the previous $\phi(n-1)$. However, a gap or insertion in a sign $t$ (e.g., less or more repetitions in a repetitive motion) requires a number of subsequent repetitions of a time frame of $t$ or a state of $R$, respectively, while otherwise steps in both are required (more or less diagonal path). Therefore, the memory-less assumption of transitions does not hold for gaps and insertions.

### 7.4 Classification

(S)DTW (or fitting an HMM) finds the best hidden sequence of a specific model in another sign by maximizing the likelihood of the observation over possible time-synchronizations. A limitation of a SDTW/HMM likelihood model is that the observation likelihood is modeled independently per state/frame, usually by mixtures of Gaussians, and only inter-frame dependence of observation likelihoods is considered. Furthermore, the same features types are used for all signs and frames, even though the relevance of these measurements may vary significantly between signs and frames. Our proposed CDFD classifier, explained in paragraph 7.4.2, not only applies feature selection, but also uses an alternative likelihood model that overcomes shortcomings of the independence assumption. In paragraph 7.4.3 we propose another classifier (Q-DFFM) that works in the joint feature space of all selected features of all frames together. But first, the next paragraph describes a robust method to discard non-discriminative features, that is used in both proposed classifiers.

#### 7.4.1 Discriminative Feature (DF) selection

Following Observation 2 and 3, we expect that recognition would greatly benefit from leaving out segments and dimensions completely from classification, if they are irrelevant or do not differ between sign classes. The features used for classification are reduced to a set of discriminative features by a robust statistical test. A feature $f_j(m)$ of type $m$ (see table I in S-IV) corresponding to the synchronized time frame $j$ is selected for classification only if the middle 50% of its distribution (between the 0.25 and 0.75 quantile) over the set $\chi_p$ of training examples of the correct sign (positive examples) has an overlap of less than 25% with the distribution of the set $\chi_n$ of training examples of incorrect signs (negative examples).

#### 7.4.2 Combined Discriminative Feature Detectors (CDFD)

After feature selection, a relatively large number of features still remains (around 500 selected out of 1900), while it is difficult to obtain a large multi-singer training set (variation of a single signer does not generalize well to others). Because of the curse of dimensionality, we assume independence between features. The classification is
7. Sign Language Recognition by combining Statistical DTW and Independent Classification

Based on assuming an independent Normal distribution \( L(\hat{t}, \mathcal{R}, j, m) \) of each feature type \( m \) in reference frame \( j \) (1-dimensional):

\[
L(\hat{t}, \mathcal{R}, j, m) = \ln \left\{ p(\hat{t}_j(m)|\mathcal{R}_j(m)) \right\} = -\frac{1}{2} \left( \ln(2\pi\sigma^2_j(m)) + \frac{(\hat{t}_j(m) - \mu_j(m))^2}{\sigma^2_j(m)} \right)
\]  

(7.5)

Usually, the feature log likelihoods, computed by equation 7.5, would be naively combined by their sum. However, this would result in a low likelihood of a sloppy but completely correct sign. Using a strictly statistical approach, this problem can only be solved by accounting for dependence between frames, which is difficult with a small training set. To overcome this problem, CDFD first converts the feature likelihood distributions to piece-wise uniform functions, which can be seen as Feature Detectors (FD):

\[
q(\hat{t}, \mathcal{R}, j, m) = \begin{cases} 
1, & \text{for } L(\hat{t}, \mathcal{R}, j, m) \geq T_j(m) - T_g \\
0, & \text{for } L(\hat{t}, \mathcal{R}, j, m) < T_j(m) - T_g
\end{cases}
\]  

(7.6)

where \( T_g \) is the gauge parameter that will determine the operating point of the final classifier and \( T_j(m) \) is the calibrated threshold that accepts 90% of the positive training data for a particular feature at \( T_g = 0 \). The choice of 90% as the calibration point is a trade-off between generalizing (include the complete range of allowed variation of a feature) and the expected reliability of the training set (reject outliers). The training set contains tracking errors and signs that were not performed well enough to be correct. We assume that these errors are below 10% for all selected features. Excluding the outer 10% of the positive distribution should eliminate the influence of these outliers in determining the default boundary of allowed variation. We have added an experiment in section A.2 of appendix A that shows the sensitivity to the choice of the acceptance rate for \( T_j(m) \). Decreasing the acceptance rate with 10% results in a decrease of approximately 0.5% in the partial Area Under the Receiving Operator Characteristic (ROC) curve between 0 and 0.1 false positive rate (pAUC0.1).

With \( q(\hat{t}, \mathcal{R}, j, m) \), all outliers outside of the allowable variation are penalized equally with a score of 0, no matter how great their distance to the mean feature value. Likewise, all variation inside the allowed interval gets the same score of 1. This makes it possible to accept sloppy but completely correct signs (e.g. signs that are made smaller than usual), while rejecting incorrect signs that are very similar to a subset of the feature models (e.g. incomplete signs).

The classifier output is generated by:

\[
Q_\mathcal{R}(t) = \sum_{j=1}^{N_\mathcal{R}} \sum_{m=1}^{N_m} s_j(m)q(\hat{t}, \mathcal{R}, j, m).
\]  

(7.7)

Where \( s_j(m) \) is 1 for selected features (from section 7.4.1) and 0 otherwise, and \( N_m \) is the number of feature types, equal to 25 (table I in S-IV). A sign is classified by:

\[
C_\mathcal{R}(t) = \begin{cases} 
correct, & Q_\mathcal{R}(t) \geq T_C \\
incorrect, & Q_\mathcal{R}(t) < T_C
\end{cases}
\]  

(7.8)
where $T_C$ is the value that classifies 50% of the positive training set correctly at $T_g = 0$ (median of $Q_R$). $T_C$ determines the fraction in time that a sign needs to be correct, hence allows for some tracking errors or small errors or hesitations in making the correct sign. Because in our application, a sign has to be made correct from beginning to end, $T_C$ is kept fixed. Instead, $T_g$ determines the allowed variation and is adapted to the allowed sloppy-ness (the operating point).

7.4.3 Quadratic Classification on DF Fisher Mapping (Q-DFFM)

Instead of combining independent feature detectors, Q-DFFM estimates a statistical model of a sign class that includes dependencies between features and time frames. To overcome the curse of dimensionality, the dimensionality of the Discriminative Feature (DF) set is reduced by Fisher mapping [23], which is a form of Linear Discriminant Analysis (LDA). The final classifier should distinguish between only two classes (‘correct’ and ‘incorrect’). However, the incorrect class of the training set is composed of many different signs classes. This fact can be exploited by finding a set of projections of DF that optimally separates all different sign classes. It can be expected that such a mapped space captures information that is generally useful to distinguish different sign classes. The Fisher mapping attempts to find the best linear separation between each class and the other classes. Projecting the initial feature space onto the separating directions for all classes results in a $N_C - 1$ dimensional feature space, with $N_C$ the total number of sign classes in the training set. When it is not possible to separate all classes with the provided measurements, some of the Fisher dimensions will be useless. Therefore, only the most discriminating $N_F \leq N_C - 1$ dimensions are used. Note that any $N_F < N_C - 1$ results in loss of optimality for separating all classes [24]. In [24] a method is proposed to regain optimality. This may lead to better performance, although our dimensionality reduction is meant for non-linear separation of the target class instead of linear separation of all classes.

The entire training set is mapped by the (subset of the) Fisher mapping, on which a quadratic (Gaussian) two-class classifier is trained. The target sign class is one of the two classes, while all examples of other sign classes in the training set are merged into a single ‘background’ class. The likelihood ratio between the two estimated Gaussian distributions is the final classifier.

7.5 Experiments

Sign classification is evaluated on a set of 120 different signs of the Dutch Sign Language (DSL), each performed by 75 different persons. The images are captured at 640x480 pixels and 25 frames per second. Most sign examples contain partial occlusions of hands with each other or with the face/neck. A description of the system setup is given in section A.4 of appendix A.

The data that are used for recognition consist of estimates of 3D locations of both hands over time, plus the size of the segmented hands in the image. They are measured from the images of a calibrated stereo camera. The image analysis procedures to extract the measurements from the camera output are described in appendix A,
section A.5 and the supplementary video "3DTracking.avi" in [25]. From these measurements, a set of 25 higher level features are extracted. 9 measurements consist of hand coordinates relative to the face or to the other hand, 14 describe the motion of the hands and 2 correspond to the size changes of the segmented hands. To reduce variation due to signer speed, the features corresponding to change are soft-thresholded above. As the average sign length is around 3 seconds, or 75 frames, the total amount of features can be more than 1500 per sign. The features obtained for each video frame are described in S-IV. Note that hand motion features are only a small fraction of all the cues that define meaning in sign language. Other cues (like hand shape, facial expression and context) are often static during a sign stroke. The dynamic motion features are expected to be more affected by Observation 1 than other features. It is assumed that better modelling of dynamic components alone will also lead to improved performance when other cues are considered as well.

We consider two different scenario’s of sign language recognition: target-class (two-class) and multi-class classification. In target-class classification, the target class is the correct sign, while examples of other signs form a second (background) class. Although multi-class classification is mostly chosen for research, it is not practically feasible when the number of classes becomes large, or multiple classes cannot be distinguished based on the measured features (full overlap). Furthermore, a rejection step for unknown classes is often omitted. In practice, rejection of unknown classes (incorrect gestures that can be anything) is one of the most important requirements. Moreover, discriminating between known sign-classes is sometimes even undesirable in case of full overlap in the measured features. Forcing an algorithm to discriminate between two indistinguishable signs will deteriorate recognition performance for both.

A target-class classification experiment consists of 120 5-fold cross-validations. One test run consists of training and testing a classifier that should distinguish one specific sign-class (denoted as the ‘positive’ class) from any other sign or gesture (negative class). Because the purpose of target-class classification is to reject unseen classes, the 120 signs classes were split into 96 training classes and 24 test classes. Only the target-class has examples both in the training set (60 examples) and the test set (15 examples). Whenever a target class is one of the assigned training classes, only the remaining 95 non-target classes are used for training, and when the target-class is one of the assigned test classes, only the remaining 23 non-target classes are used for testing. For Q-DFFM, the number of dimensions \( N_F \) of the Fisher mapping is optimized by maximizing the average partial Area Under the Receiver Operating Characteristic (ROC) Curve between 0 an 0.1 false positive rate (pAUC\(_{0.1}\)) of a 6 fold cross-validation on the training set.

Note that the classifier is tested only on negative classes that it has never seen before. Unlike multi-class classification, the performance in this test will not necessarily decrease with a larger number of classes, as the classifier is tested as a one-against-all (two-class) classifier. The performance may even increase if more classes are added to the negative training set, as it will improve generalization.

To test target-class classification in MCL space, the samples of the negative training classes for each model were also split into 60 training samples and 15 test samples, just like the positive set, to prevent the positive test samples of a target class being used
Table 7.1: A comparison between standard HMM and SDTW by pAUC

<table>
<thead>
<tr>
<th></th>
<th>Description</th>
<th>pAUC 0.1</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>40 state HMM</td>
<td>84.61%</td>
</tr>
<tr>
<td>b</td>
<td>40 state SDTW trained as HMM</td>
<td>87.57%</td>
</tr>
<tr>
<td>c</td>
<td>40 state SDTW</td>
<td>90.60%</td>
</tr>
<tr>
<td>d</td>
<td>~74 state SDTW</td>
<td>90.54%</td>
</tr>
</tbody>
</table>

as negative samples for training the single-class models of other classes.

In the target-class case, classifier performance can be evaluated by the Receiver Operating Characteristic (ROC) curve that shows all possible operating points. One point on the ROC curve denotes the false positive error rate with the corresponding true positive rate for a specific classification threshold. The area under the ROC curve is averaged over all cross-validations and positive sign classes to obtain a total score. The larger the area, the better. As we are only interested in the operating points with realistic (usable) results, only the ROC curve partial AUC between a false positive rate of 0 and 0.1 are considered (pAUC\_0.1). To get a more detailed view on performance, also the ROC curves themselves are combined by averaging true positive rates at fixed false positive rates.

Multi-class classification was also tested in 5-fold cross-validation, using a feature space of all 120 single-class models, trained on 60 and tested on 15 samples of each class. The performance is evaluated by the average of the classification error rate in each cross-validation.

### 7.5.1 SDTW Outperforms HMM

In the first experiment, we compared SDTW to HMM. Results can be seen as the pAUC’s in table 7.1. The HMM’s have 40 states and Bakis topology (left to right with single-state skips and self-transitions). The length of the HMM is a trade-off between modeling detail and the minimum length (maximum speed) that can be recognized. This HMM size is comparable to [26], where the average length was 41 at the same frame rate of 25 fps. The HMM’s are trained with Baum-Welch, but evaluated using the Viterbi algorithm. In SDTW, the transition probabilities were not used (assumed equal). To see the influence of different aspects of SDTW, the HMM is converted to full-scale SDTW in three steps. First ("b" in table 7.1), the trained HMM models are evaluated as SDTW by using their state means and covariances. This already gives a significant performance improvement over "a". Apparently, HMM really suffers from the rigid warping constraints. In "c", also the training is done using SDTW. This results in a comparable performance increase over "b" as "b" had over "a". This is not so surprising, as the same warping restrictions of HMM are expected to be a limitation during training as well. The third step "d" increases the length of the SDTW model to the average length of the positive training examples. This is not possible with HMM since a HMM can never have more states than the number of frames of the smallest sequence. However, no significant performance change can be observed due to increasing the number of states (p-value 0.52 in a paired t-test of the pAUC’s). This is because the DSL signs do not have as many different details as the number of frames.
Table 7.2: A comparison between different ways of handling transitions in SDTW by pAUC0.1.

<table>
<thead>
<tr>
<th></th>
<th>Method</th>
<th>pAUC0.1</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>SDTW + trans. prob. in warp only</td>
<td>90.54%</td>
</tr>
<tr>
<td>b</td>
<td>SDTW + trans. prob. in warp &amp; likelihood</td>
<td>90.27%</td>
</tr>
<tr>
<td>c</td>
<td>unbiased SDTW</td>
<td>88.14%</td>
</tr>
<tr>
<td>d</td>
<td>cityblock SDTW</td>
<td>87.81%</td>
</tr>
<tr>
<td>e</td>
<td>unbiased SDTW + trans. prob. in warp only</td>
<td>87.69%</td>
</tr>
</tbody>
</table>

recorded here (25 per second). Therefore, multiple frames can be modeled with the same state. Adding more states does not provide better modeling accuracy.

7.5.2 Transition Probabilities are Questionable

To test the influence of transition probabilities in SDTW, several possibilities of using transition probabilities have been compared. According to Lemma 2, we expect a negative effect of using transition probabilities in the likelihood computation. However, according to Lemma 1, we expect a positive effect of transition probabilities for the warping itself. The results in table 7.2 are only partially consistent with our predictions. As predicted by Lemma 2, using transition probabilities in the class-likelihood computation resulted in a decrease of performance ("b" versus "a"). Although small, the difference with using transition probabilities only in warping was significant with p-value $= 8 \times 10^{-22}$. Note that we cannot be certain if this performance decrease is due to Lemma 2 or because a poor modeling of transition probabilities (e.g. the memory-less assumption).

Adding transition probabilities only in the warping step ("a" in table 7.2) had no effect compared to SDTW without transition probabilities ("d" in table 7.1). The p-value was 0.74. This might be explained from the huge scale difference between transition probabilities and the observation likelihoods. Because of the high-dimensional Gaussian models, likelihoods can differ so much that they totally dwarf the influence of the transition probabilities. However, there will probably also be cases where the differences of the observation likelihoods are not so large, so that still does not explain why no difference can be seen at all. Another explanation may be that the bias towards short paths in the SDTW warping may overrule the influence of the transition probabilities. To test this hypothesis, the experiments are repeated using unbiased warping ("e" in table 7.2). Although now the transition probabilities in warping indeed have an influence (p-value $= 9 \times 10^{-6}$ compared to "c"), the effect is opposite from what was expected. The transition probabilities actually have a negative influence on warping instead of positive, suggesting poor modeling capability. Apparently, the standard use of transition probabilities in SDTW is questionable. On the contrary, the bias in SDTW does provide an important positive effect on performance (table 7.1"d" versus 7.2"e"). This is because it gives preference to diagonal transitions, and therefore shorter, more linear paths. Linear warping is mostly the best guess/prior when observation likelihoods are inconclusive, hence this result is in support of Lemma 1. City block STDW (table 7.2"d") is also unbiased, because the diagonal transition is omit-
7.5. Experiments

Table 7.3: Results for target-class classification using a single model or using 3 different methods in MCL space. Measured pAUC0,1.

<table>
<thead>
<tr>
<th>Method</th>
<th>single model</th>
<th>MCL Fisher</th>
<th>MCL QDC</th>
<th>MCL SVM</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>HMM</td>
<td>84.61%</td>
<td>96.97%</td>
<td>95.94%</td>
</tr>
<tr>
<td>b</td>
<td>SDTW</td>
<td>90.54%</td>
<td>97.22%</td>
<td>96.29%</td>
</tr>
<tr>
<td>c</td>
<td>SDTW+CDFL</td>
<td>95.35%</td>
<td>90.86%</td>
<td>74.17%</td>
</tr>
<tr>
<td>d</td>
<td>SDTW+CDFD</td>
<td>95.46%</td>
<td>94.84%</td>
<td>91.17%</td>
</tr>
<tr>
<td>e</td>
<td>SDTW+Q-DFFM</td>
<td>96.62%</td>
<td>97.42%</td>
<td>96.29%</td>
</tr>
<tr>
<td>f</td>
<td>SDTW+DF-Fisher</td>
<td>92.70%</td>
<td>92.70%</td>
<td>92.70%</td>
</tr>
<tr>
<td>g</td>
<td>SDTW+DF-SVM</td>
<td>95.29%</td>
<td>94.84%</td>
<td>91.17%</td>
</tr>
<tr>
<td>h</td>
<td>HMM+CDFD</td>
<td>91.03%</td>
<td>94.84%</td>
<td>91.17%</td>
</tr>
<tr>
<td>i</td>
<td>HMM+Q-DFFM</td>
<td>95.73%</td>
<td>97.22%</td>
<td>96.29%</td>
</tr>
<tr>
<td>j</td>
<td>SDTW&amp;DFFM5</td>
<td><strong>97.50%</strong></td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>

7.5.3 Hybrid Approach Best for Single-Model Target Classification

First, we have compared the hybrid methods to SDTW and HMM, when a single model is used. The results are shown in the "single model" column of table 7.3. Besides our proposed CDFD and Q-DFFM methods ("d" and "e", respectively), we have also tested a Fisher classifier ("f"), and SVM with radial basis function kernel ("g"). However, for these methods we have also applied our DF feature selection, since otherwise the dimensionality was too high. The Combined Discriminative Feature Likelihood (CDFL) method ("c") is an intermediate step between the likelihood computation in SDTW and CDFD. Instead of Gaussian modeling of all features per frame together, as in SDTW, all features are modeled independently by a 1D Gaussian, and only the selected features are used. The result of CDFL is significantly better than SDTW ("b"). This is in accordance with Observation 2, which implies that a lot of measurements can be discarded to improve performance.

The benefit of a hybrid approach is clearly visible, with the best result for Q-DFFM ("e" *). The result for Q-DFFM is even better than CDFD ("d"), with p-value = 5 × 10⁻⁹. However, the practical advantage of CDFD is that the operating point can be set more intuitively. The threshold is directly proportional to the allowed variation in the selected measurements. Although this may not be a theoretical advantage, in practice, setting the operating point of a classifier is a problem in itself. We have also tested combinations of HMM+CDFD/Q-DFFM ("h" and "i") to see if the proposed limitations of HMM are really a problem. Indeed, the results are worse when com-
pared to the same classification methods combined with SDTW warping ('h' versus 'd' and 'i' versus 'e'). These differences can be due to the rigid HMM warping and/or the lower number of states in HMM (40), which is a consequence of the warping rigidity.

One of the main advantages of splitting warping and classification (following Lemma 3) is that class-conditional likelihood estimation can be done using only the features that are meaningful for a particular class. Here we give three DSL sign examples that were accepted by the corresponding model of 'SDTW+Q-DFFM', but rejected by that of 'SDTW'. The original video files "ExCHOP.avi", "ExBANANA.avi" and "ExMETAL.avi" can be downloaded from [25]. Figures 7.1, 7.2 and 7.3 show the motion paths of both hands through space and time. The original videos are also provided in the supplemental material. For the sloppy example of the sign "to chop", in figure 7.1 (b) the time-signal of the height of both hands is compared with that of a very 'good' example. Especially the end of the stroke (before retraction) shows a large difference, although this is not a very important part of the sign. In the example of the sign "metal" (figure 7.2), the tracking does not start correctly. Both hands actually started at the lowest point (on the table), while, according to the measurements, the left hand started almost 15 cm higher. At the retraction in the end, the left hand is erroneously assigned to the same object as the right hand. In the example of "banana" (figure 7.3) the tracking even mistakenly swaps around left and right hand assignment before the middle of the sign. Q-DFFM can still distinguish a correct sign from other signs by the essential parts that are performed or tracked correctly, while SDTW already gets confused when the (irrelevant) beginning or ending of a sign are different then usual. Note that when signs are used within a sentence, co-articulation would also change the beginnings and endings considerably, depending on preceding/succeeding signs or resting position.
Figure 7.1: Example of the sign “to chop”, which is detected by Q-DFFM but not by SDTW. In (a) the trajectory in 3D space and in (b) the height of both hands against time.
Figure 7.2: Example of the sign "metal", which is detected by Q-DFFM but not by SDTW. In (a) the trajectory in 3D space and in (b) the height of both hands against time.
Figure 7.3: Example of the sign "banana", which is detected by Q-DFFM but not by SDTW. In (a) the trajectory in 3D space and in (b) the height of both hands against time.
7.5.4 Hybrid Model-Combining Requires Different Approach

So far, we have classified a target-class by its likelihood using the sign's own SDTW model. Combining the outputs of multiple models is common practice to increase performance. Therefore, in this experiment, we concatenate the likelihood output of the target-class classifier for the actual target-class and the outputs of the target-class classifiers trained for all 95 or 96 (depending on the cross-validation step) non-target classes (as they were trained for the experiments in paragraph 7.5.3). This forms a 96- or 97-D Multiple Class Likelihood (MCL) feature space in which a second-stage classifier can be trained. This is done with the likelihood estimations of HMM, SDTW, SDTW+CDFD and SDTW+Q-DFFM. We applied three different classifiers in MCL space: Fisher, Quadratic (Gaussian) Discriminant Classifier (QDC) and linear SVM.

The results for target-class classification in MCL space are shown in the last three columns of table 7.3. While HMM and SDTW have improved by combining multiple likelihood models ("a","b"), the hybrid target-class classification methods have decreased in performance ("d","e"). This is probably because the single-model hybrid classifiers are too specialized for discrimination of one class. CDFD likelihood is meaningful only for signs of which at least $T_{C}$ of DF is similar to the target sign. The Q-DFFM likelihood is a linear separation with the target sign on one end and all the other signs on the other. Therefore, the MCL dimensions corresponding to non-target-classes may contain less information about the real target-class for CDFD and Q-DFFM than HMM and SDTW models do. A change of HMM/SDTW likelihood for the target-class model, due to an allowable sign variation, may strongly correlate to changes of likelihoods for models of other classes. These relations can be exploited by the second-stage classifier. Furthermore, the single-model hybrid approaches are more vulnerable to errors in the warping of the single SDTW model.

Contrary to the Q-DFFM output, DFFM does contain information that can be useful for other classes, for its 96 (or 97) dimensions consist of linear separations for all sign classes in the training set, based on the alignment to the target-class SDTW model. The DFFM mappings for the 96 (or 97) SDTW models in the training set could be combined in a single 9216-(or 9409-)D feature space that would be highly correlated. However, this high dimensionality poses a computationally complex optimization problem which is beyond the scope of this article. Instead, as a proof of concept for this multi-model hybrid approach, we have expanded the SDTW MCL space with the 5 best separating dimensions of each DFFM mapping of SDTW-synchronized features. This results in a (96 or 97)$\times$(1+5)=576-(or 582-) D feature space. The results for a Fisher classifier in this MCL&F space is shown in table 7.3("j"). QDC and SVM were not able to run at this data size. Despite of the 500% increase of dimensionality, the performance with the added information has increased from 97.22% to 97.50% with a p-value of 0.009.
Table 7.4: Classification accuracy for multi-class classification using three different discriminants in MCL space. Measured pAUC₀.₁.

<table>
<thead>
<tr>
<th>Method</th>
<th>MCL Fisher</th>
<th>MCL QDC</th>
<th>MCL NN</th>
</tr>
</thead>
<tbody>
<tr>
<td>a HMM</td>
<td>90.8%(σ2.0)</td>
<td>70.3%(σ1.5)</td>
<td>76.2%(σ4.3)</td>
</tr>
<tr>
<td>b SDTW</td>
<td>90.8%(σ1.4)</td>
<td>79.2%(σ1.7)</td>
<td>80.0%(σ3.6)</td>
</tr>
<tr>
<td>c SDTW+CDFD</td>
<td>76.0%(σ2.89)</td>
<td>36.6%(σ6.6)</td>
<td>43.0%(σ3.6)</td>
</tr>
<tr>
<td>d SDTW+Q-DFFM</td>
<td>83.7%(σ1.1)</td>
<td>84.3%(σ2.2)</td>
<td>80.0%(σ2.7)</td>
</tr>
<tr>
<td>e SDTW&amp;DFFM5</td>
<td>*92.3%(σ1.2)</td>
<td>-</td>
<td>82.9%(σ2.6)</td>
</tr>
</tbody>
</table>

7.5.5 Multi-class Classification Not Suitable to Detect a Target-Class

Multi-class classification is performed in the exact same way as the model combining for target-class classification in paragraph 7.5.4. Only now the second stage classifier is trained as a multi-class classifier combining the outputs of single-model target-class classifiers for all 120 sign classes. This gives five recognition rates (corresponding to the cross-validations) for classifying a sign as one of 120 classes. The average rates and standard deviations are shown in table 7.4. Since this is a multi-class problem, Nearest Neighbor (NN) is used instead of SVM. Because there are three pairs and two triplets of sign classes that cannot be distinguished by motion alone, the maximum achievable recognition rate by using motion alone is 94.1%. If hand shape would be added, only one ambiguous pair would remain, raising the limit to 99.2%. It cannot be expected that this rate can be achieved using the 2D hand size change as the only hand shape feature.

Remarkably, HMM ("a") now performs better than SDTW ("b"). However, the difference is practically 0 (p-value = 0.94). The single-model hybrid methods underperform in the combined space, just like when combined for target-class classification. Again, the hybrid model-combining method SDTW&DFFM5 ("e") results in a significant improvement over SDTW, with an average accuracy of 92.3%. One of the five cross-validations even reached a recognition rate of 93.7%, which comes close to the maximum of 94.1% that can be achieved with hand motion alone. The improvement over HMM outputs combined with Fisher has a p-value of 0.019.

Since the result of multi-class classification can be used to detect a target-class, we can compare multi-class classification with the target-class classifiers described above. The SDTW&DFFM5 Fisher-combined multi-class classifier would erroneously recognize a random sign of an unseen class as the target class at a false positive rate of 1/120, while a target sign would be recognized correctly 92.3% of the time. For the best target-class classifier (SDTW&DFFM5, table 7.3 "j"), the true positive rate at a false positive rate of 1/120 is 93.0% on average. This is significantly higher, with p-value 0.054 over the five cross-validations.
7. Sign Language Recognition by combining Statistical DTW and Independent Classification

7.6 Conclusions

We have proposed and evaluated a hybrid approach to sign language recognition by using SDTW only for time-warping and a separate classifier on the warped features. One of the main advantages of this approach is that non-discriminative features can be discarded to reduce dimensionality and noise. This is especially important in sign language, as signs are often constrained only within a subset of all possible degrees of freedom. The two single-model classification methods we proposed (SDTW+CDFD and SDTW+Q-DFFM) both significantly outperform SDTW by itself in target-class classification.

We have also confirmed that SDTW provides a significant improvement over HMM because of the warping rigidity in HMM. However, we have observed that transition probabilities in SDTW provide a poor prior on DTW path shape, and can even decrease recognition performance. On the other hand, the DTW warping bias, introduced by not compensating for the increased length of non-diagonal transitions, actually improved performance, acting as a prior on path shape with preference for shorter, more linear paths.

Furthermore, we have found that when a second-stage classification on the likelihood outputs of multiple target-class classifiers is applied, results from multiple SDTW or HMM models improve, while the hybrid methods degrade. We have shown that a successful model-combining hybrid method can be obtained by including the DFFM mappings for separate SDTW models in the feature space for the second stage, in addition to SDTW likelihoods. This resulted in a significant improvement over HMM and SDTW both in target-class classification using combined models and in multi-class classification.

Although recognition relied mainly on 3D hand motion features, it can be expected that these results generalize to more detailed measurements such as hand/body pose and facial expressions.
Bibliography


7. Sign Language Recognition by combining Statistical DTW and Independent Classification


7. Sign Language Recognition by combining Statistical DTW and Independent Classification
Chapter 8

Learning to Recognize a Sign from a Single Example

©2008 IEEE. Personal use of this material is permitted. However, permission to reprint/republish this material for advertising or promotional purposes or for creating new collective works for resale or redistribution to servers or lists, or to reuse any copyrighted component of this work in other works, must be obtained from the IEEE.

Abstract We present a method to automatically construct a sign language classifier for a previously unseen sign. The only required input of a new sign is one example, performed by a sign language tutor. The method works by comparing the measurements of the new sign to signs that have been trained on a large number of persons. The parameters of the respective trained classifier models are used to construct a classification model for the new sign. We show that the performance of a classifier constructed from an instructed sign is significantly better than that of Dynamic Time Warping (DTW) with the same sign. Using only a single example, the proposed method has a performance comparable to a regular training with five examples, while being more stable because of the larger source of information.

8.1 Introduction

Obtaining a representative training set for a person-independent sign language recognition method can be a tedious task. The small intra-person variability, compared to the inter-person variability ([1]), requires that a training set contains examples of many different individuals. We have obtained a training set of 120 signs of Dutch Sign Language (DSL), performed by 75 different persons. The hand-motion of the signs is captured by a 3D visual hand tracking system [2]. If a sign language application requires to recognize a sign that is not within this set (which is a highly likely situation), it would save a lot of time and effort if the existing set can be exploited to simplify learning of an unseen sign. In this paper, we attempt to simplify this in the most extreme way: by learning from only one example. We assume, however, that the example is well-performed (e.g. by a sign language tutor).

The classification method we use is based on the Combined Discriminative Feature Detectors (CDFD), as presented by us in [2]. Instead of Statistical Dynamic Time Warping (SDTW), we will use a variant of Derivative DTW (DDTW) [3] to synchronize time-frames.

8.2 Related Work

At this moment, sign language recognition with only a single training example has a very limited history. In [4] Wang, Chen and Geo perform a Hidden Markov Model (HMM) training of 2435 sign language signs with three examples per sign, measured with data-gloves. From each three examples, 18 new examples are generated by resampling three different sign components. This expansion of the training set increased recognition performance from 84.19% to 85.35%. Kadir et al. [5] have proposed a visual method that can learn sign language recognition with as little as one example per sign class. Recognition accuracy on a set of 164 sign classes was 89.1% with five training examples and dropped to 76.2% with a single training example. In both of these papers, training and testing was done with examples from the same person. As already argued in [1], results with a single person cannot be generalized to person-independent recognition.

Contrary to generating artificial examples or using model-based generalization,
8.3 Classification Method

The timing and length of a sign depend on a signers’ speed. Therefore, a direct comparison of measurements is not possible. Classification must contain some form of
time-synchronization. In Hidden Markov Models (HMM) and Statistical Dynamic Time Warping (SDTW) [9], time-warping and likelihood estimation are combined in a single probabilistic framework. However, in [2] we have argued that warping and likelihood estimation should be handled as separate problems. Because many examples are needed to train a HMM or SDTW model, we will use Dynamic Time Warping (DTW) for time-synchronization. DTW works with only a single example. Classification of warped signals is performed with a variant of Combined Discriminative Feature Detectors (CDFD) [2], described in paragraph 8.3.2.

8.3.1 Dynamic Time Warping

Dynamic Time Warping (DTW) compares two vector sequences \( t = [t_1, \ldots, t_N_t] \) and \( r = [r_1, \ldots, r_N_r] \), with \( t_i, r_j \in \mathbb{R} \). Using the cost function \( d : \mathbb{R}^F \times \mathbb{R}^F \to \mathbb{R} \) (where \( F \) is the number of feature types) a \( N_t \times N_r \) cost matrix \( C^d \) is computed that contains the distances \( C^d(i, j) \) between all pairs of vectors \( t_i \) and \( r_j \). Usually, the Euclidean distance is used:

\[
d(t_i, r_j) = ||t_i - r_j||
\]

(8.1)

It is common practice to normalize both signals \( t \) and \( r \) to have zero mean and unit variance. In Derivative Dynamic Time Warping (DDTW), proposed by Keogh and Pazzani [3], the derivatives of the features are used instead of the features themselves. The advantage is that differences in signal height (due to an offset or amplitude difference) do not influence the difference of derivatives as much as the absolute difference between signals. This makes DDTW more robust against large amplitude and offset differences. However, some important signal properties are lost by taking the derivative. Since location is an important property in many sign language signs, omitting it would eliminate important cues. Therefore, we have chosen a compromise by concatenating features and their derivatives:

\[
d(t_i, r_j) = ||\{t_i, \Delta t_i\} - \{r_j, \Delta r_j\}||
\]

(8.2)

Where

\[
\Delta x_i = (x_{i+1} - x_{i-1})/2
\]

(8.3)

\( t_i, \Delta t_i, r_j \) and \( \Delta r_j \) have to be normalized before computing Euclidean distance, to get comparable contributions of all dimensions. This poses a problem in sign language recognition, where many signs are performed single-handed. Blowing up measurements of a still hand to have standard deviation 1 leads to an increased influence of noise in warping. We have overcome this by normalizing dimensions in groups of similar measurements. The highest standard deviation of all dimensions in a group is used to normalize all dimensions in the group. When similar dimensions of both hands are in the same group (e.g., horizontal hand position), the measurements of a still hand will be scaled by the same factor as similar measurements of the other (moving) hand. By grouping all spatial dimensions of hand positions into the same group, blowing up dimensions orthogonal to movement is also prevented.
Figure 8.1: Finding the optimal DTW path $\Phi^*$ through cost matrix $C(t, r)$ with dynamic programming. Starting from $\Phi(1) = (1, 1)$ (the bottom left), $\arg\min_{\Phi(n)} C(\Phi(t, r))$ iteratively optimizes the cost of the sub-path up to each location $(i, j)$ in $C(t, r)$ by choosing the last transition $P(n)$ from $\Phi(n)$ to $\Phi(n + 1) = (i, j)$. Note that the actual value of $n$ depends on this choice, as it is a choice between three alternative paths that end in different $\Phi(n)$ with possibly different lengths $n$.

In the rest of this document, we have omitted specifying the cost function $d$ in $C^d$, as it is clear from the context. DTW finds the alignment path $\Phi^*$ in matrix $C(t, r)$, that minimizes

$$
C_\Phi(t, r) = \frac{\sum_{n=1}^{N} d(\phi_t(n), \phi_r(n)) w(P(n))}{\sum_{n=1}^{N} w(P(n))}
$$

(8.4)

$$
C^*(t, r) = C_{\Phi^*}(t, r) = \min_{\Phi} \frac{\sum_{n=1}^{N} d(\phi_t(n), \phi_r(n)) w(P(n))}{\sum_{n=1}^{N} w(P(n))}
$$

(8.5)

Where $\Phi = \{\phi_t(1), ..., \phi_t(N_\phi), \phi_r(1), ..., \phi_r(N_\phi)\}$ are the steps of the path through the 2D correspondence matrix $C(t, r)$ of the time frames of $t$ and $r$, $P(n)$ is a transition step between two subsequent points of the path $\Phi$ through $C(t, r)$ and $w(P(n))$ is a function that assigns a weight to a transition type. $C^*(t, r)$ is also used as the final match cost. Some constraints of the path are implied to confine the minimization procedure to practical results. The most common constraints are that the path starts in $\Phi(1) = (1, 1)$ and ends in $\Phi(N_\phi) = (N_t, N_r)$, and the set of possible transitions $P$ is limited to

$$
P(n) = [(\phi_t(n + 1) - \phi_t(n)), (\phi_r(n + 1) - \phi_r(n))]
$$

$\in \{[0, 1], [1, 0], [1, 1]\}$,

(8.6)

corresponding to horizontal, vertical and diagonal steps, respectively. (8.5) is solved using dynamic programming. The path finding algorithm is illustrated in figure 8.1.
8.3.2 Combined Discriminative Feature Detectors

The Combined Discriminative Feature Detectors (CDFD) method \cite{2} consists of a robust feature selection procedure, explained in paragraph 8.3.2, and a method of constructing binary feature detectors that are used as a voting scheme, explained in paragraph 8.3.2.

**Discriminative Features**

The features used for classification are reduced to a set of discriminative features by a robust statistical test. A feature $\hat{t}_j(m)$ of type $m$ corresponding to the synchronized time frame $j$ is selected for classification only if the middle 50% of its distribution (between the 0.25 and 0.75 quantile) over the set $\chi_p$ of training examples of the correct sign (positive examples) has an overlap of less than 25% with the distribution of the set $\chi_n$ of training examples of incorrect signs (negative examples).

**Feature Detection**

The classification is based on assuming an independent Normal distribution $\mathcal{N}(\mu_j(m), \sigma_j(m))$ of each feature type $m$ in reference frame $j$. However, since the feature likelihoods are binarized anyway, the distance from $\mu_j(m)$ will do equally well:

\[
q(\hat{t}_j(m)) = \begin{cases} 
1, & \text{for } |\hat{t}_j(m) - \mu_j(m)| \leq T_j(m) T_g \\
0, & \text{for } |\hat{t}_j(m) - \mu_j(m)| > T_j(m) T_g 
\end{cases}
\]  

(8.7)

where $T_g$ is the gauge parameter that will determine the operating point of the final classifier and $T_j(m)$ is the calibrated threshold that accepts 90% of the positive training data for a particular feature at $T_g = 1$. The choice of 90% as the calibration point is a trade-off between generalizing (include the complete range of allowed variation of a feature) and the expected reliability of the training set (reject outliers). The training set contains tracking errors and signs that were not performed well enough to be correct. We assume that these errors are below 10% for all selected features. Excluding the outer 10% of the positive distribution should eliminate the influence of these outliers in determining the default boundary of allowed variation.

With $q(\hat{t}_j(m))$, all outliers outside of the allowable variation are penalized equally with a score of 0, no matter how great their distance to the mean feature value. Likewise, all variation inside the allowed interval gets the same score of 1. This makes it possible to accept sloppy but completely correct signs (e.g. signs that are made smaller than usual), while rejecting incorrect signs that are very similar to a subset of the feature models (e.g. incomplete signs).

The combined output is generated by:

\[
Q_r(t) = \sum_{j=1}^{N_c} \sum_{m=1}^{N_m} s_j(m) q(\hat{t}_j(, m)). \tag{8.8}
\]
Where \( s_j(m) \) is 1 for selected features (from section 8.3.2) and 0 otherwise, and \( N_m \) is the number of feature types. A sign is classified by:

\[
C_r(t) = \begin{cases} 
  \text{correct}, & Q_r(t) \geq T_C \\
  \text{incorrect}, & Q_r(t) < T_C 
\end{cases}
\]  

(8.9)

where \( T_C \) is the value that classifies 50% of the positive training set correctly at \( T_g = 1 \) (median of \( Q_r \)). \( T_C \) determines the fraction in time that a sign needs to be correct, hence allows for some tracking errors or small errors or hesitations in making the correct sign. Because in our application, a sign has to be made correct from beginning to end, \( T_C \) is kept fixed. Instead, \( T_g \) determines the allowed variation and is adapted to the allowed sloppy-ness (the operating point).

### 8.4 Cross-Generalization Method

The goal of cross-generalization is to learn about a new class from generalizing information of known classes. In the CDFD method, the information of the known classes consists of three components:

1) The feature values \( \mu_j(m) \) of the warping signal (the reference sign),
2) The feature selection \( s_j(m) \) of feature type \( m \) at frame number \( j \),
3) The feature detection threshold \( T_j(m) \) of feature type \( m \) at frame number \( j \).

The information of the new class is the single example, which will be used as the warping signal and will also be used as \( \mu_j(m) \). Since \( s_j(m) \) and \( T_j(m) \) can only be determined from statistics of multiple positive examples, they cannot be determined from a single example. Instead, they can be derived from generalizing the values of \( s_j(m) \) and \( T_j(m) \) from comparable parts of CDFD models of different classes, that are already trained on multiple positive examples. Generalization is performed primarily at frame level. Meaning that all features at one time frame together form a tuple that can be compared and chosen to use for generalization to a frame of the novel class. This level is chosen because important aspects of Dutch Sign Language (DSL) can be very short, e.g. a sudden stop of a motion. These short events will be overlooked when only longer segments would be considered. Because we expect that many of the features and their possible range of variation will depend on each other, generalizing at feature level seems a step too far.

It can be expected that using the reference sign as feature mean \( \mu \) influences feature selection \( s_j(m) \) and the optimal threshold level \( T_j(m) \). Therefore, the trained CDFD models from which is generalized are also trained using the reference sign as \( \mu \).

A measure is needed to compare a frame \( j \) of the new reference sign to a frame \( i \) of one of the reference signs of the trained classes. When matching at frame-level, all information about frame time is lost. However, the first and last parts of a sign contain the resting pose and the preparation or retraction. These parts are usually left out by the DF selection because they are very similar for all signs. Therefore, frame time can be very relevant to generalize feature selection \( s_j(m) \). To overcome this shortcoming in generalizing at frame-level, relative frame time is added as an additional dimension of inter frame distance. Frame difference is computed by the
squared Euclidean distance:

\[
D(r_j, t_i) = \left( \frac{j}{N_r} - \frac{i}{N_t} \right)^2 + \sum_{m=1}^{N_m} (r_j(m) - t_i(m))^2
\]  

To prevent unbalanced influence of features, depending on their variance, all features and frame times are first normalized to have variance 1 over the total set of concatenated frames from all trained signs.

Using only the frame \( t_i \) with minimal \( D(r_j, t_i) \) to determine \( s_j(m) \) and \( T_j(m) \) may produce unstable results. If matching is not perfect or there is no frame in the trained set very similar to the frame of the new sign, the statistics of the best matching frame may not represent the statistics of the frame \( r_j \) of the new sign very well. Robustness can be increased by combining the models of multiple matching frames. Therefore, a set of frames \( S \) is found, as the \( k \) frames with the lowest value of \( D(r_j, t_i) \). \( s_j(m) \) and \( T_j(m) \) are derived as the respective medians over \( S \).

### 8.5 Experiments

The data that have been used for recognition consist of estimates of 3D locations of both hands at 25 times per second, plus the size of the segmented hands in the image. They are measured using a passive stereo camera together with a 3D hand tracking algorithm. An image from the recorded video is shown in figure 8.2. From these measurements, a set of 25 higher level features are extracted. The data set consists of 120 different signs of the Dutch Sign Language (DSL), each performed once by 75
different persons. The amount of test data is increased by combining results of 5-fold cross-validations.

Sign classification is evaluated as a two-class (or target-class) problem, where one sign (the target sign) must be distinguished from all other possible gestures. This means there are actually 120 different classifiers that are evaluated at each cross-validation step. Because the purpose of target-class classification is to reject unseen classes, the 120 signs classes were split into 96 training classes and 24 test classes. Only the target-class has examples both in the training set (60 examples) and the test set (15 examples). Whenever a target class is one of the assigned training classes, only the remaining 95 non-target classes are used for training, and when the target-class is one of the assigned test classes, only the remaining 23 non-target classes are used for testing.

For each test of one target-class classifier, a Receiver Operating Characteristic (ROC) curve is determined. The partial Area Under the Curve (pAUC$_{0.1}$) between a False Positive (FP) rate of 0 and 0.1 is used as a performance measure. This lower part is more relevant than the rest, since higher FP rates are impractical operating points.

For the generalized CDFD (gCDFD) method, a $k$ of 10 frames was used. For each generalized classifier, the trained models corresponding to the non-target classes were used as a source for model construction, while only one reference example (not part of the data set) was known about the target class. The ROC curves and corresponding average pAUC$_{0.1}$ for different methods and settings are shown in figure 8.3. The best (highest) ROC curve is for DWT+CDFD. To have a reference of the best possible performance with the proposed cross-generalization method, also the result of DTW+CDFD is shown for when the reference sign is used for the feature means.
Figure 8.4: Percentage of Partial Area Under the Curve (pAUC\textsubscript{0.1}) of the ROC per sign class, for generalized CDFD (gCDFD) versus Dynamic Time Warping (DTW) with the same instruction example. For both methods, the sign classes are separately ordered on pAUC\textsubscript{0.1}. This leads to a small reduction in pAUC\textsubscript{0.1} from 96.01% to 95.44%. Using only the frame-time (and not the actual features) in the frame matching of equation 8.10 leads to a drop of pAUC\textsubscript{0.1} to 89.16%. When the 10 frames for generalization are chosen randomly, the performance deteriorates to a pAUC\textsubscript{0.1} of 80.09. This shows the importance of choosing the right source to generalize from.

Figure 8.5 shows the learning curve of DTW+CDFD for an increasing training set size. The learning curve already starts to level out at 10 examples. On average, gCDFD performs comparable to CDFD trained with 5 examples. However, the standard deviation \( \sigma_{cv} \) of the pAUC\textsubscript{0.1} over the five cross-validations, averaged over all sign classes, is higher for CDFD. This means that when five examples are used for training, gCDFD provides more stable results than CDFD. This is because cross-generalization provides the stability of a large source of information, while training with a small set results in a highly variable quality of the target-class training set.

The baseline method for single-example recognition is DTW itself. DTW uses only the reference example to get a warping distance that is used as a classifier. It is probably the most commonly used method of single-example learning of sign language. The result for generalized CDFD (gCDFD) in combination with DTW (pAUC\textsubscript{0.1} 91.52%) is significantly higher than DTW (pAUC\textsubscript{0.1} 88.38%). The p-value of the paired t-test is \( 3 \times 10^{-15} \). This shows that cross-generalization can really help gesture recognition when only one example is available. Figure 8.4 shows that the distribution of differences between DTW and DTW+gCDFD is also spread relatively evenly between ‘difficult’ and ‘easy’ classes.
Figure 8.5: DTW+CDFD Learning curve, compared to the performance level of generalized CDFD (gCDFD), which uses only a single example. The vertical axes show percentages of partial Area Under the Curve (pAUC\textsubscript{0.1}) of the ROC curves. DTW+CDFD shows a ceiling effect around 10 examples in both the pAUC\textsubscript{0.1} (a) as well as the its average standard deviation (b) over cross validations $\sigma_{cv}$. 
Note that, although cross-generalization has increased performance significantly over DTW, the performance is still significantly lower than CDFF with appropriate training. This large difference suggests that a database of 96 trained signs is not enough to generalize to any other sign. It can be expected that the required number of trained source-classes will only get larger when more modalities of sign language are used (higher dimensionality), such as hand shape, lip movement and facial expression.

8.6 Conclusions

We have applied cross-generalization to sign language recognition by proposing a method for classifier construction from classifiers trained for the recognition of other signs. Generalization is performed at time-frame level by comparing frames of a new example with frames of the trained models. Information about frame time, which is lost by this approach, is explicitly added as an additional dimension of frame difference. The proposed cross-generalization method leads to a significant improvement over Dynamic Time Warping. Using only a single example for learning, the proposed method performed comparable to a regular training with a set of five examples, while having more stable results.
Bibliography


Chapter 9

Discussion

The primary objective of the research in this PhD work was to achieve robust visual gesture recognition. This has been done by focusing on a specific application (electronic sign language learning) and targeting research on some crucial components, with this application in mind. The result was a working system that could actually be tried out in practice. As mentioned in the introduction, this primary goal of a robust, functional system was expected to achieve a secondary, more general goal: To gain tools, knowledge, insights and inspiration, for focusing future work in computer vision research. The discussion below elaborates on the insights that have been gained in the process towards the primary goal, and the practical experience with the prototype recognition system.

9.1 Tracking

In tracking of dynamic objects, the knowledge about an object’s recent history, and about its physical behavior, can greatly reduce the search space for estimating the current state. However, when tracking autonomous, conscious entities, such as human beings, care must be taken not to extend the motion prediction model further than what can be reasonably assumed. In many cases, a Taylor expansion further than zeroth order may not be valid, since a spontaneous decision to switch behavior may happen at any given moment. When a certain time-lag is acceptable, such switches of behavior may be detected and incorporated in the state predictions. However, this may not be possible when immediate (real-time) tracking is required, or if motion cannot be modeled as ‘piecewise-predictable’.

As demonstrated in chapter 2, particle filtering does not solve the problem of incorrect modeling. A large uncertainty translates to a large random innovation variable in the state transition, causing a wide spread of particles over the state-space. The first priority of a tracking methodology is to prevent ‘loss of target’. Since this is already difficult enough, estimating a full probability density function is, in many cases, simply a bridge too far. Accuracy is valuable, however impossible without first achieving robustness.
One of the strengths of particle filtering lies in the random sampling of the state space. Where a failed naive deterministic search for the object state is bound to fail again in the next time-step, a random search can cover the state space with much greater resolution, by searching at different locations each time. However, this ‘side-effect-benefit’ of particle filtering can be achieved in a much more structured, efficient way. By randomly sampling all particles independently, many small areas are sampled multiple times, while leaving other areas of the state space unexplored. In the hand tracking of the prototype system, this was simply solved by searching in a regular grid pattern around the previous hand location. The whole pattern was randomly shifted in each frame, according to a uniform distribution within one period of the grid. Many variations of this principle are possible, such as a structured instead of random shifting, to maximize the coverage of the search over time, or a decrease of grid density further away from the predicted location, to cover a larger area over a larger period of time, with the same amount of computation. If there is any other need to use particle filtering, such an ‘intelligent’ deterministic sampling process can be easily implemented within an importance sampling framework.

9.2 Color

Under certain conditions, color can be a robust property for object recognition. It can be invariant to scale, resolution, orientation and shape. This makes color especially suitable for detection and segmentation of highly articulate objects, such as hands.

In general, however, the conditions do not allow for using color as a robust property. Even if reflection would be 100% Lambertian (which is in general not the case) reflected color is not only a function of the object surface itself, but also of the spectra of all light sources that illuminate that surface. A typical environment consists of a myriad of light sources with different spectra. Even if there is only a single lamp in a completely shielded room, the spectrum of illumination will not necessarily be homogeneous. There will be a range of orientations under which an object’s surface is not (primarily) illuminated by the lamp, but by the indirect, colored reflections of light coming from other surfaces, such as a colored carpet or wallpaper.

There are mainly two ways to obtain robustness from color information: ‘control’ or ‘knowledge’. In the prototype system, the illumination was controlled by a shielded tent with identical lamps and neutral-colored walls, ceiling, table and computer-screen contents. In many cases, however, full control is not possible. In such cases, color information cannot be used simply as a preprocessing step. Knowledge about object type, shape and environment are necessary to interpret color. When the invariance to object shape is regarded as the only benefit of color, this looks like a chicken-and-egg problem. Therefore, the use of color will never be generalizable in a strictly bottom-up approach to computer vision. In the future, however, more and more active (analysis by synthesis) vision approaches can be expected, where multiple object aspects are regarded together, combined in integral iterative processes of hypothesis testing, fitting and tracking, utilizing an ever increasing awareness of environment and context. Color may finally obtain a rightful place in unconstrained environments, albeit in a completely different form from how it is currently used.
9.3 Features

Similar to the use of color, the popular bottom-up approach to computer vision has caused a strong trend towards maximizing invariance of features. Complex invariant features may allow the application of simple pattern recognition methods to complex images. Incorporating invariance already in the first steps of processing comes at a price, though. Variations in details may contain important information about object state, which is lost when all information from low-level processing is invariant to object state. In more sophisticated, multi-layer computer vision systems, invariance may shift towards higher levels of processing, making use of simple, variant features, such as the isophote properties described in chapter 3, which apply a meaningful transformation of low-level image content, with a minimal loss of relevant information. Ideally, state-invariance will become unnecessary altogether, when object localization and state estimation are combined in an integral, active vision process.

9.4 Recognition of Time-Variable Signals

In chapter 7, some shortcomings of Dynamic Time Warping (DTW) and Hidden Markov Models (HMM) were shown to be harmful for recognition performance with sign language. Although the proposed improvements did make a significant difference, they are not likely to be the best solution possible. For instance, for some signals (perhaps even for some signs), the warping does make a difference for the meaning. Furthermore, repetitive signals with less repetitions than the model, will still be more difficult to recognize, because features mapped to the extra model repetition will not correspond properly.

Another aspect that has not been addressed is the relation between the level of detail and the length of a dynamic model. More complex signals may better be split up into separate dynamic models, combined into higher-level dynamic models. Again, in the future, an active approach will probably be most effective, where hypotheses can be fitted to the data and iteratively tested and adjusted.

An interesting direction for future research may also be a mutually exclusive warping and class likelihood estimation. Using the same measurements for class likelihood as for warping will result in a biased result. This is because a time warping method attempts to make different signals as similar as possible. Even if the signals are from different classes. By computing the likelihood of a warped signal based on features that have not been used to obtain the warping, this bias could be reduced. To use all available features for classification, multiple warping/likelihood computation results may be combined, where, each time, the features are divided differently between warping and likelihood computation.

9.5 Human Computer Interaction

When it comes to Human-Computer Interaction (HCI), robustness is doubtless many times more important than accuracy or richness of extracted information. In an in-
dustrial process, the consequences of errors due to unprecedented variations of item instances can simply be derived from the error rate. However, this perspective is meaningless from the viewpoint of the objects/subjects themselves.

When someone buys a game computer that works under 99% of circumstances, and either his living room or his physical appearance falls in the 1% of circumstances in which the machine makes errors, the other 99% of circumstances, under which the game computer does function correctly, are totally irrelevant to him. The device is absolutely 100% non-functional to this consumer, and he will accept nothing else than a full refund of his money. This additional subject-centered perspective is the crucial difference between industrial and HCI applications, that should be reflected in the role of computer vision research in the context of HCI. Focussing on the refinement of methods that depend on unrealistic assumptions, may be a complete waste of time and money.

At the same time, shying away from controlling the environment may be an unnecessary limitation. If a factor of the environment can be controlled without compromising usability, there is no reason not to increase robustness in this way. Some controlled elements may even increase usability. Such as the tent in the prototype system, which also created a more pleasant environment for the user: softly lit and free from distractions. Computer vision does not have to solve everything. An integral approach should extend beyond the data that comes out of the back end of the camera.
Appendix A

Supplemental Material for Chapter 7

©2008 IEEE. Personal use of this material is permitted. However, permission to reprint/republish this material for advertising or promotional purposes or for creating new collective works for resale or redistribution to servers or lists, or to reuse any copyrighted component of this work in other works, must be obtained from the IEEE.

This chapter is based on the supplemental material for the article published as “Sign Language Recognition by combining Statistical DTW and Independent Classification”, by J. F. Lichtenauer, E. A. Hendriks and M. J. T. Reinders in the IEEE Trans. Pattern Analysis and Machine Intelligence, ?? 2008.
A.1 Introduction

This appendix belongs to the supplemental material of chapter 7. It is accompanied by video files that can be found in [1]:

1. Supplemental.pdf (the original document from which this appendix is derived)
2. 3DTracking.avi
3. ExCHOP.avi
4. ExBANANA.avi
5. ExMETAL.avi

File 2 belongs to section A.5, to explain the image processing steps for tracking the hands, used for generating the experimental data set. Files 3-5 are example video’s of signs in the data set, which belong to paragraph 7.5.3 of chapter 7.

This appendix contains additional information that did not contribute to the readability of chapter 7. Section A.2 contains an experiment to show the effect of changing a parameter in CDFD, section A.3 is an introduction on Dynamic Time Warping (DTW), section A.4 describes the practical setup that was used for recording the data, section A.5 explains the visual tracking procedure, section A.6 describes the processing for extracting the features and section A.7 explains the new curvature method that we used for extracting the curvature features.

A.2 Robustness CDFD

The threshold value $T_j(m)$ in the CDFD method explained in section V-B is set on the value that accepts 90% of the training set. This choice is a trade-off between strictness and the influence of outlier data. The sensitivity of the performance to this choice is shown in figure A.1. The average pAUC of one-class classification with CDFD (using SDTW) is plotted for multiple acceptance rates. The performance for acceptance rates lower than 90% drops gradually, while there is a steep drop for higher rates. This suggests that the safe side is a conservative choice.
Figure A.1: Sensitivity for the feature acceptance rate on which $T_j(m)$ is set
A.3 Dynamic Time Warping (DTW)

DTW compares two vector sequences $t = [t_1, \ldots, t_N]$ and $r = [r_1, \ldots, r_N]$, with $t_i, r_j \in \mathbb{R}$. Using the cost function $d : \mathbb{R}^F \times \mathbb{R}^F \rightarrow \mathbb{R}$ (where $F$ is the number of feature types) a $N_t \times N_r$ cost matrix $C^d$ is computed that contains the distances $C^d(i, j)$ between all pairs of vectors $t_i$ and $r_j$. Usually, the Euclidean distance is used:

$$d(t_i, r_j) = ||t_i - r_j||$$  \hspace{1cm} (A.1)

In the rest of this document, we have omitted specifying the cost function $d$ in $C^d$, as it is clear from the context. DTW finds the alignment path $\Phi^*$ in matrix $C(t, r)$, that minimizes

$$C^* (t, r) = C_{\Phi^*} (t, r) = \min_{\Phi} C_{\Phi} (t, r).$$  \hspace{1cm} (A.3)

Where $\Phi = \{\phi_t(1), \ldots, \phi_t(N_{\phi})\}, \phi_r(1), \ldots, \phi_r(N_{\phi})\}$ are the steps of the path through the 2D correspondence matrix $C(t, r)$ of the time frames of $t$ and $r$, $P(n)$ is a transition

---

Figure A.2: Finding the optimal DTW path $\Phi^*$ through cost matrix $C(t, r)$ with dynamic programming. Starting from $\Phi(1) = (1, 1)$ (the bottom left), $\min_{P(n)} C_{\Phi} (t_{\{1, \ldots, i\}}, r_{\{1, \ldots, j\}})$ iteratively optimizes the cost of the sub-path up to each location $(i, j)$ in $C(t, r)$ by choosing the last transition $P(n)$ from $\Phi(n)$ to $\Phi(n+1) = (i, j)$. Note that the actual value of $n$ depends on this choice, as it is a choice between three alternative paths that end in different $\Phi(n)$ with possibly different lengths $n$. The minimizing path corresponds to the optimal alignment.

$$C_{\Phi} (t, r) = \frac{\sum_{n=1}^N d(t_{\phi_t(n)}, r_{\phi_r(n)})w(P(n))}{\sum_{n=1}^N w(P(n))}$$  \hspace{1cm} (A.2)

Where $\Phi = \{\phi_t(1), \ldots, \phi_t(N_{\phi})\}, \phi_r(1), \ldots, \phi_r(N_{\phi})\}$ are the steps of the path through the 2D correspondence matrix $C(t, r)$ of the time frames of $t$ and $r$, $P(n)$ is a transition
step between two subsequent points of the path $\Phi$ through $C(t, r)$ and $w(P(n))$ is a function that assigns a weight to a transition type. $C^\ast(t, r)$ is also used as the final match cost. Some constraints of the path are implied to confine the minimization procedure to practical results. The most common constraints are that the path starts in $\Phi(1) = (1, 1)$ and ends in $\Phi(N_\phi) = (N_t, N_r)$, and the set of possible transitions $P$ is limited to

$$P(n) = \{[\phi_t(n+1) - \phi_t(n), \phi_r(n+1) - \phi_r(n)]\}$$

$$\in \{[0, 1], [1, 0], [1, 1]\},$$

(A.4)
corresponding to horizontal, vertical and diagonal steps, respectively. (A.3) is solved using dynamic programming. The path finding algorithm is illustrated in figure A.2.

**A.4 System Setup**

Figure A.3 gives an overview of the physical setup of the camera system. The person is seated in front of a desk. At frontal view, stereo cameras are placed to record the person’s signs.

![Figure A.3: Practical setup of ELo.](image)

Signs are recorded with two calibrated digital cameras, Allied Vision Technologies ‘Guppies’, at 25 fps, resolution 640 x 480. Currently, start and end of a sign must be indicated by putting the hands in a fixed position on the table top.

To record the data set for training and testing, the stereo baseline and camera position were up-scaled to adult size. Furthermore, the recording was done in a studio...
with 2000W of indirect lighting. Most test persons did not know sign language them-
selves but learned it from an example video of the sign made by one of three different
instructors.

In the experiments we present, extreme timing variations due to hesitations (which
we would expect in a practical application) do not occur. The only cause of the timing
variations in the used data set is due to inter-personal variation, which can be expected
in most applications of sign language recognition. The differences with respect to
timing variation compared to most other research on sign language recognition are that
on the one hand we used more different persons, but on the other hand most persons
were no (native) signers and were mimicking an instruction video. To prevent the
mimicking of the same instruction video from reducing the inter-personal variation,
we used instruction videos from three different instructors (randomly chosen).

A.5 Image Analysis

The image processing operations used to measure 3D hand locations can be divided
into two layers: single-camera tracking, followed by disparity refinement. In this
order, computing a complete disparity map can be omitted. This saves a signifi-
cant amount of redundant computation, as we only need disparity measurements of
hands and face. A demonstration is and short explanation is provided in the video file
"3DTracking.avi".

A.5.1 Single-Camera Tracking

The hands and face are found around their previous location by a combination of blob
tracking and template searching. This is done separately for both cameras (2D), but
uses the depth from the previous time frame as prior information on hand size. When
possible and necessary, tracking is automatically (re-)initialized by assigning the skin
blobs to hands and head according to the blob positions.

Blob Tracking

finds the blob with a center of gravity closest to the previous hand location. The blobs
are connected components of a skin color segmentation. As long as no occlusion
occurs and segmentation is reliable, blob tracking is both fast and robust.

To get a reliable skin segmentation, we use the adaptive model described in [2].
The method fits a 3-part piecewise linear model to the positive samples in RGB space.
The model is robust against intensity offset and ambient lighting color. Furthermore,
initialization on one person applies to a large range of other skin colors. However,
this depends on lighting conditions and skin color difference. In a semi-automatic
initialization procedure, skin and other (non-skin) samples are collected from a camera
image, in which the user annotates skin/non-skin regions. These samples are used to
model skin color and to set a threshold on the distributions.

The initialized model provides a skin likelihood for any RGB color tuple. How-
ever, simply thresholding this likelihood results in a lot of false positive skin detec-
Figure A.4: Image processing example. In (a) the left camera image with the back-projected 3D hand positions as squares of which the size is determined from the estimated depth. (b) motion segmentation. (c) the skin segmentation where a buffered face image is used to remove skin pixels of the face.

Categories. Instead, two different segmentation thresholds are applied: A high $H$ and a low $L$ threshold. The $H$ segmentation contains little false positives, but many misses. $L$ covers almost all skin area, but also contains a lot of false positives in background and clothes. False positives are reduced to a minimum by using the positive detections of $L$ only around areas with positive detections of $H$. This is usually done using hysteresis thresholding. To limit computation time spend on dilations, we reduced this to only one large dilation after the first threshold. To filter out sporadic false positives in $H$, they are removed by a density filtering $F_d$ that sets a lower threshold on the number of positive pixels in a local neighborhood around each positive pixel, using the integral image \[3\] from which the image sum over a rectangle can be computed from the cumulative values at the 4 corners. The final skin segmentation is obtained by combining the low threshold and high threshold results, followed by a morphological closing $C$ to connect falsely detached segments:

$$S_s = C \left\{ D(F_d(H)) \right\} \cap L \right\} \quad (A.5)$$

Where $\cap$ denotes a logical AND, D a dilation.

For computational efficiency, the $H$ and $L$ thresholds are applied off-line to all possible RGB tuples $C = [C_R, C_G, C_B]^T \in \{0, .., 255\}^3$ and stored in lookup tables $T_H(C_R, C_G, C_B)$ and $T_L(C_R, C_G, C_B)$, respectively. This also makes it possible to combine the likelihood model (a simplified generalization of reality) with a histogram of the positive and negative initialization samples. Non-skin values in $T_L$ that coincide with a high number of positive samples are set positive to reduce false negatives. Positive values in $T_H$ that coincide with a high number of negative samples are removed from $T_H$ to reduce false positives. To reduce data size for effective caching, RGB space is quantized into 64x64x64 color bins and the boolean table values are packed into 32 bit words, resulting in 64KB of data all together. To further reduce false negatives, $T_H$ is applied to a larger image size (320x240) and the result saved to a 160x120 segmentation in which one pixel is positive if at least one of four corresponding pixels in the larger image is positive.
Template Tracking

finds the local maximum correlation with a template copied from the hand location in the previous frame. In the template search, the template value differences are weighted by a Gaussian function after limiting them to a maximum of 20, to reduce the effect of outlier and background pixels inside the template and search area. The template search is automatically adapted to the situation. The search grid scale is linearly dependent on the distance of each hand in the previous frame, and the grid size (number of points) is reduced significantly if no motion is detected at the previous hand location. Furthermore, only grid points within the skin segmentation are considered. When motion is detected at the previous hand location, this is further reduced to grid points at areas with both skin and motion. Motion areas are segmented by a threshold on the local sum of absolute frame differences, using the integral image method. Figure A.4 (b) shows a motion segmentation example. The noise threshold for motion segmentation is determined in the same initialization procedure as in paragraph A.5.1. Because it is very cumbersome and unpractical to get an image containing no motion at all (especially with a wide angle camera) the 50% most still regions were used for setting the threshold. This assumes that at least 50% of the image contains no motion (only noise). This is usually the case for a normal situation where only one person sits in front of the camera. It is very difficult to track a hand in front of the face. Especially when the hand changes shape. Therefore, the search area is further reduced by face segmentation. In each video frame where no hand is near the head area, the area of the head is copied from the gray image. The pixels in each new frame that are similar enough to the corresponding pixels in the buffered face image are removed from the skin segmentation, resulting in a face-less skin segmentation image, used to reduce tracking search space. Figure A.4 (c) shows a face-less skin segmentation example.

Combined Blob/Template Tracking

The results of blob and template tracking are combined depending on the situation. When a hand blob is not connected to the other hand and outside of the head/hair area, the blob center is considered most reliable. It is averaged with the template search result to get a more precise estimation, but only if they are close enough. Otherwise, only the blob center is used. When a hand blob is merged with the other hand blob, or close to the head/hair, only the template search result is used. When two hands blobs are merged, and their difference in depth is large, only the closest hand is tracked, while the other is assumed to stand still.

A.5.2 Disparity Refinement

For the result of single-camera tracking in one camera, the stereo disparity is measured by a coarse to fine block search of the found hand patch along the epi-polar curve (distorted line) in the other camera image, with a range slightly wider than the maximal expected displacement from the previous 3D location. If the single-camera tracking results are good and the estimated disparities from left to right and right to left are correct, they should be very close to each other. In that case, they are averaged to
A.6 Features

The data that are used for recognition consist of estimates of 3D locations of both hands over time, plus the size of the segmented hands in the image. They are measured from the images of a calibrated stereo camera, explained in figure A.3. The image analysis procedures to extract the measurements from the camera output is described in section A.5. From these measurements, a set of 25 higher level features are extracted. As the average sign length is around 3 seconds, or 75 frames, the total amount of features can be more than 1500 per sign. The feature types obtained for each video frame are shown in table A.1. X, Y, Z are horizontal, vertical and depth coordinates, respectively. The median face location was taken as the origin for the hand coordinates. Furthermore, t is the time frame number, s is arc length of the hand motion path, B is hand blob size in pixels. No other hand shape features than size change get a more precise and stable estimate of the hand location. If the two results do not correspond, the result that is closest to the previous 3D location of the hand is used. If the result is physically impossible (too far or too fast), it is ignored and the previous 3D location and templates are retained. The refined 3D hand locations are projected back to camera coordinates to facilitate tracking in the next frame.

Table A.1: 25 feature types extracted for classification.

<table>
<thead>
<tr>
<th>Feature Type</th>
<th>Formula</th>
</tr>
</thead>
<tbody>
<tr>
<td>left/right hand coordinates (6)</td>
<td>( \mathbf{h}<em>{l/r}(i) = [\bar{X}</em>{l/r}(i), \bar{Y}<em>{l/r}(i), \bar{Z}</em>{l/r}(i)]^T )</td>
</tr>
<tr>
<td>left/right hand motion (2)</td>
<td>( \dot{h}_{l/r}(i) = S \left(</td>
</tr>
<tr>
<td>left/right hand acceleration (2)</td>
<td>( \ddot{h}_{l/r}(i) = S \left(</td>
</tr>
<tr>
<td>left/right side-way orientation (2)</td>
<td>( \theta_{\text{SL/r}}(i) = \arcsin(d\bar{X}_{l/r}(s)/ds) )</td>
</tr>
<tr>
<td>left/right upward orientation (2)</td>
<td>( \theta_{\text{UL/r}}(i) = \arcsin(d\bar{Y}_{l/r}(s)/ds) )</td>
</tr>
<tr>
<td>left/right forward orientation (2)</td>
<td>( \theta_{\text{FL/r}}(i) = \arcsin(d\bar{Z}_{l/r}(s)/ds) )</td>
</tr>
<tr>
<td>left/right hand motion curvature (2)</td>
<td>( \kappa_{l/r}(i) = S \left( -\kappa_{l/r}(i), c_\kappa \right) )</td>
</tr>
<tr>
<td>left/right hand motion curvature change (2)</td>
<td>( \dot{\kappa}<em>{l/r}(i) = S \left( d\kappa</em>{l/r}(i)/dt, c_\kappa \right) )</td>
</tr>
<tr>
<td>left/right hand size change (2)</td>
<td>( \dot{B}<em>{l/r}(i) = S \left( dB</em>{l/r}(i)/dt, c_B \right) )</td>
</tr>
<tr>
<td>left-right difference 3D (1)</td>
<td>( \Delta h_{l/r} =</td>
</tr>
<tr>
<td>left-right difference vertical (1)</td>
<td>( \Delta Y_{l/r} = Y_l(i) - Y_r(i) )</td>
</tr>
<tr>
<td>left-right difference depth (1)</td>
<td>( \Delta Z_{l/r} = Z_l(i) - Z_r(i) )</td>
</tr>
</tbody>
</table>
could be robustly extracted from the skin blobs. Other gesture recognition systems in literature have also included details like blob main axis orientation and/or eccentricity [4, 5]. However, this was mostly done with colored gloves or other circumstances that result in perfectly segmented hands without wrist, instead of a hand together with the variably exposed part of the wrist. We did not use colored gloves because it would disturb the children too much and limit the allowable color of clothes even further than only skin-like colors. Several measurements are mapped with a sigmoid function $S(f, c)$:

$$S(f, c) = \frac{1}{1 + \exp(-f/c)}, S(f, c) \in [0, 1]$$  \hspace{1cm} (A.6)

Where $c$ is a scaling parameter that determines where the sigmoid flattens out. In the time derivative features, sigmoid mapping acts as a soft threshold to obtain invariance to signer speed. For curvature $\tilde{\kappa}$, it reduces a logarithmic infinite-range scale measurement into a limited feature range. A bar $\bar{\cdot}$ above a variable or function in table A.1 denotes a Gaussian smoothing over time, to reduce the influence of measurement noise.

The extraction of meaningful higher-level motion information is one of the most crucial steps of the proposed gesture recognition method. This step adds valuable knowledge and invariance to the classifier input, instead of only supplying it the raw measurements.

### A.6.1 Pre-Processing

Because the measurements contain noise from inaccuracies and tracking errors, they are filtered in two steps:

1) Median filtering with filter size $S_m$ to discard extreme outliers.
2) Gaussian filtering with standard deviation of $\sigma$.

The minimal choice of $S_m$ and $\sigma$ depends on the accuracy and robustness of the image processing steps, but on the other hand should be small to retain as much small-scale pose change as possible. In our case, a median filter with $S_m$ of 3 time frames was sufficient. For smoothing curved motion paths, $\sigma = \sigma_h$ of 1 time frame was a reasonable trade-off.

The 3D coordinate system used for tracking was aligned with the stereo camera.

### A.6.2 Hand Coordinate System

To be robust against variations of signer location, the face location was taken as the origin for the hand coordinates. To prevent tracking errors in the face location to superimpose on the locations of the hands, a single head location is estimated for each hand gesture, by taking the median 3D coordinates of the center of the face during the whole gesture. We assume that the head movement is not significant during a gesture.

For a hand location $h = [X, Y, Z]^T$ in millimeter, $X$ is the horizontal coordinate (directed positively in left direction for the left hand and in the right direction for the right hand), $Y$ the vertical coordinate directed upwards and $Z$ the horizontal coordinate directed in the forward orientation of the person, parallel to the virtual camera axis.
(rectified stereo camera). Because the $X$ coordinates of the hands are taken positive to their respective hand side, during a symmetric gesture both hands share identical coordinates, although they are at the opposite side of the head. Mirroring a gesture now comes down to simply swapping the properties of the two hands. This makes it easy to recognize signs with mirror-invariance using a classifier trained on examples with only one of the two possibilities.

A.6.3 Time-Derivative

At each time frame $i$, a time derivative of a hand property $x$ is estimated by:

$$\frac{\delta x(i)}{\delta i} = \frac{x(i + 1) - x(i - 1)}{2}$$  \hspace{1cm} (A.7)

For the first and last time frames, the neighboring results are replicated.

A.6.4 Hand Speed

The amount of motion is described by speed $\dot{h}$ and acceleration $\ddot{h}$:

$$\dot{h}(i) = \| \frac{\delta h(i)}{\delta i} \|$$  \hspace{1cm} (A.8)

$$\ddot{h}(i) = \frac{\delta \dot{h}(i)}{\delta i}$$  \hspace{1cm} (A.9)

Where $\ddot{h}(i)$ is $\dot{h}(i)$ filtered again with the Gaussian filter described above, using the same $\sigma = \sigma_h$. Speed and acceleration can have any real value between $[0, \infty)$ or $(-\infty, \infty)$, respectively. The exact values of $\dot{h}$ and $\ddot{h}$ are subject to (personal) variation. However, holding a hand still does have a different meaning than moving a hand constantly or accelerating / decelerating it. Therefore the features are warped into $\hat{\dot{h}}$ and $\hat{\ddot{h}}$ respectively, with a sigmoid function that behaves linearly around 0 but compresses and limits higher and lower values, effectively applying a ‘soft-clipping’. The sigmoid function is shown in figure A.5.

$$\hat{\dot{h}} = \frac{2}{1 + \exp(-\dot{h}/c_{\dot{h}})} - 1, \hat{\dot{h}} \in [0, \infty], \hat{\dot{h}} \in [0, 1]$$  \hspace{1cm} (A.10)

$$\hat{\ddot{h}} = \frac{2}{1 + \exp(-\ddot{h}/c_{\ddot{h}})} - 1, \hat{\ddot{h}} \in [-\infty, \infty], \hat{\ddot{h}} \in [-1, 1]$$  \hspace{1cm} (A.11)

The constants $c_{\dot{h}}$ and $c_{\ddot{h}}$ determine how fast the sigmoid function converges to 1 or -1. This has to be significantly above measurement noise level and also above the human capability and perception of ‘not’ or ‘constantly moving a hand’, but short enough to map all significant movement and acceleration to approximately the same value. This is especially useful for generalization to less-controlled situations where persons might sign significantly faster than the signs in the training set. In our case, $c_{\dot{h}} = 5$ and $c_{\ddot{h}} = 0.5$ were appropriate values. These values soft-clip speed above approximately $15\text{mm/frame}$ ($375\text{mm/s}$) and acceleration above $1.5\text{mm/frame}^2$ ($37.5\text{mm/s}^2$) and below $-1.5\text{mm/frame}^2$. 

Figure A.5: Sigmoid function $\frac{2}{1 + \exp \left( -\frac{f}{c_f} \right)} - 1$, used for soft-clipping of feature type $f$.

### A.6.5 Motion Orientation

The orientation of 3D motion is described by three angles relative to three perpendicular planes:

1) The vertical symmetry plane of the body, parallel to the virtual camera axis. Orientation with respect to this plane $\theta_S$ is defined as the ‘sidewayness’:

$$\theta_S = \arcsin \left\{ \left[ 1, 0, 0 \right] \frac{\delta h(i)}{\delta t} \right\} \| \frac{\delta h}{\delta t} \|$$

$$\approx \arcsin \left( \frac{dX(s)/ds}{\| \frac{\delta h}{\delta t} \|} \right)$$

(A.12)

2) The horizontal plane. Orientation with respect to this plane $\theta_U$ is defined as the ‘upwardness’:

$$\theta_U = \arcsin \left\{ \left[ 0, 1, 0 \right] \frac{\delta h(i)}{\delta i} \right\} \| \frac{\delta h}{\delta t} \|$$

$$\approx \arcsin \left( \frac{dY(s)/ds}{\| \frac{\delta h}{\delta t} \|} \right)$$

(A.13)

3) The camera plane: The vertical plane parallel to the image plane of the stereo camera. Orientation with respect to this plane $\theta_F$ is defined as the ‘forwardness’:

$$\theta_F = \arcsin \left\{ \left[ 0, 0, 1 \right] \frac{\delta h(i)}{\delta i} \right\} \| \frac{\delta h}{\delta t} \|$$

$$\approx \arcsin \left( \frac{dZ(s)/ds}{\| \frac{\delta h}{\delta t} \|} \right)$$

(A.14)

Note that to describe a 3D orientation only two of these properties are sufficient. However, some signs of sign languages contain motions that are not defined in absolute orientation, but are rather defined in one or two of the above terms and often even confined to orientation in one of the three planes. No minimal description with only two parameters can contain invariance for motions within or relative to all three planes.
A feature selection procedure should take care of choosing the invariant orientation properties.

A.6.6 Curvature

The change of motion orientation is important to distinguish between straight motion and various sizes of curves, circles or a complete change of direction. This property is captured in a single feature of ‘curvature’ $\kappa$ that is invariant to absolute orientation and speed. However, curvature estimation in a discrete set of sampled points is not a trivial matter. Variations of sample distance (speed) and momentary changes of orientation pose significant problems for methods based on curve fitting or angle differences. A simple change of direction (180 degrees turn) is easily overlooked or highly underestimated. Furthermore, not too many subsequent samples should be taken into account, as the sampling rate of 25Hz is low compared to the fast orientation changes in some signs. To meet all requirements as close as possible, we have developed a novel, computationally efficient method that estimates the osculating circle from the Largest Isosceles Triangle (LIT) between three subsequent samples. It is explained in section A.7.

For the curvature feature $\tilde{\kappa}$, used for recognition, $\kappa$ is warped with the sigmoid function of (A.6) and smoothed with a Gaussian filter with $\sigma = \sigma_\kappa$, indicated by the bar above the fraction:

$$\tilde{\kappa} = \frac{1}{1 + \exp (\kappa/c_\kappa)}, \ k \in [0, \infty], \ \tilde{\kappa} \in [0, 0.5] \quad (A.15)$$

Where the constant $c_\kappa$ is chosen to obtain reasonable variation of $\tilde{\kappa}$ within the working range of $\kappa$. This is necessary because $\kappa$ has logarithmic behavior (since it is a scale property), but logarithmic features are difficult to use for classification purposes. The choice of $c_\kappa$ depends on the maximal curvature of circular motion that can be measured. This depends on the spatial sampling density, which is dependent on both hand speed and frame rate. A significantly higher curvature indicates either a change of direction (180$^\circ$ turn) or an angular frequency that is too high to follow with the current sampling rate. We have chosen $c_\kappa = .05$ as a trade-off that maps all extreme curvatures to approximately the same value, significantly higher than the regular curvature values, without dwarfing them. The smoothing is to suppress noise due to measurement errors and smooth out large curvature deviations due to under-sampling.

In our case, $\sigma_\kappa = 2$ was a good trade-off between noise suppression and motion detail. The derivative of $\tilde{\kappa}(i)$, $\tilde{\kappa}(i)$, is meaningful to distinguish between a transition from a straight motion to a curved/corner motion or the other way around:

$$\dot{\kappa}(i) = \frac{\delta \tilde{\kappa}(i)}{\delta i} \quad (A.16)$$

Similar to acceleration, the absolute change of curvature is not as informative as the sign of change. Therefore, $\dot{\kappa}$ is also soft-clipped with a sigmoid function:

$$\hat{\kappa} = \frac{1}{1 + \exp (-\kappa/c_\kappa)}, \ \hat{\kappa} \in [-0.5, 0.5], \ \hat{\kappa} \in [0, 1] \quad (A.17)$$
Where \( c_\kappa = .02 \) is a trade-off that maps most corners and changes of direction to approximately the same value, but significantly above noise level.

### A.6.7 Hand Size

The apparent hand size is measured from the segmented skin blobs in both cameras. The exact size is not very reliable due to personal variation of hand size and length of sleeve, but the size change is coded in a property \( \dot{B}(i) \):

\[
\dot{B}(i) = \frac{\delta \left( B_L(i) + B_R(i) \right)}{\delta i}
\]  
(A.18)

where \( B_L \) and \( B_R \) are the pixel size in the left and right camera, respectively, and the bar above \( B_L(i) + B_R(i) \) indicates filtering with a Gaussian filter with standard deviation \( \sigma_B = 2 \) is a trade-off between noise suppression and detail. Finally, \( \dot{B} \) is soft-clipped to obtain signer-invariance:

\[
\tilde{\dot{B}} = \frac{2}{1 + \exp \left( -\dot{B}/c_\dot{B} \right)} - 1, \quad \tilde{\dot{B}} \in [-\infty, \infty], \quad \dot{B} \in [-1, 1] 
\]  
(A.19)

Where \( c_\dot{B} = 10 \) clips \( \dot{B} \) significantly above noise level.

### A.7 LIT Curvature Estimation

Curvature \( \kappa \) of a motion path is calculated by

\[
\kappa = \frac{1}{R}
\]  
(A.20)

with \( R \) the radius of the osculating circle of the hand trajectory. \( R \) is estimated from the Largest Isosceles Triangle (LIT) between a sample and its two neighbors. The estimated osculating circles at two subsequent samples around a 180 degrees turn are shown in figure A.6.

Having three subsequent 3D positions of a hand \( h_{i-1}, h_i \) and \( h_{i+1} \), and the two vectors from \( h_i \) to the neighboring positions

\[
a_i = h_{i-1} - h_i \]  
(A.21)

\[
b_i = h_{i+1} - h_i, \]  
(A.22)

first, the symmetrical sides of the largest isosceles triangle between the three points are found by normalizing \( a_i \) and \( b_i \) to the shortest length:

\[
a_i' = \frac{a_i}{||a_i||} \min(||a_i||, ||b_i||),
\]  
(A.23)

\[
b_i' = \frac{b_i}{||b_i||} \min(||a_i||, ||b_i||).
\]  
(A.24)
Vectors $c_i$ and $d_i$ are found by

$$c_i = \frac{b_i - a_i'}{2},$$  \hspace{1cm} (A.25)  

$$d_i = a_i' + c_i.$$  \hspace{1cm} (A.26)  

Curvature $\kappa(i)$ is determined by:

$$\kappa(i) = \begin{cases} 
\frac{1}{||c_i||}, & \text{for } ||c_i|| \leq ||d_i|| \\
\frac{2||d_i||}{||a_i'||^2}, & \text{for } ||c_i|| > ||d_i|| 
\end{cases}$$ \hspace{1cm} (A.27)  

The exception in (A.27) for an angle between the three points of less than 90° (like in figure A.6a) is necessary to estimate the osculating circle of a sharp turn that is much shorter than the distance between three samples. When the motion path in figure A.6 would be completely flattened in vertical direction, then $\kappa(i) \to \infty$ and $\kappa(i+1) \to 0$. An infinitely sharp turn of 180 degrees will always give at least one curvature measure of $\kappa = \infty$. Although mathematically this is regarded as an exceptional situation, the results get very close in signs with repetitive linear motions. On the contrary, with a method based on curve fitting (e.g. B-spline), the infinitely short turn will be overlooked completely, and an angle derivative (normalized by the distance between samples) will highly underestimate the curvature depending on the sampling rate or motion speed. An alternative solution in a curve fitting approach would be to find the maximal curvature between samples. LIT curvature makes such extra calculations unnecessary. Furthermore, note that in the ideal noise-free case of a perfect circular motion and uniform sampling at a rate of at least $2/\pi$ times the angular frequency, LIT curvature is exact, unlike B-splines and angle derivatives.
Bibliography


Figure A.6: Curvature estimation using the Largest Isosceles Triangle (LIT) between three points. A 3D hand motion path (curved gray arrow) is sampled at four locations, indicated with $h$. The osculating circle is estimated from the LIT in (a) for time $i$ and in (b) for $i+1$. Although in these examples the estimations of the osculating circle radii do not match the curve exactly at the two respective time points, they are a better representation of the turn in the curve in between three subsequent points. The turn is an important feature that would otherwise be missed.
Samenvatting

Het fundamentele doel van dit proefschrift is om meer inzicht te krijgen in wat er komt kijken bij het toepassen van een beeldherkenningssysteem in de praktijk, wanneer de gebruiksomstandigheden niet volledig vastgelegd kunnen worden. Het uitgangspunt hierbij is dat onderzoek op geïsoleerde onderdelen van beeldherkenning dikwijls leidt tot 'te' generieke oplossingen. Dat deze oplossingen de robuustheid en nauwkeurigheid missen, die enkel gehaald kunnen worden met een integrale benadering van een specifieke applicatie. Bovendien kan een integrale benadering, en het daadwerkelijk uitproberen van een beeldherkenningssysteem in de praktijk, weer leiden tot nieuwe inzichten die bepalend kunnen zijn voor de richting van toekomstig onderzoek in beeldherkenning.

De toepassing voor het onderzoek in dit proefschrift is automatische gebarenherkenning voor terugkoppeling bij actief leren in een elektronische leeromgeving voor gebarentaal. Het doel van deze leeromgeving is om de gebarenschat te vergroten van dove en slechthorende kinderen, tussen 3 en 5 jaar oud, en zo te helpen om hun taal achterstand te verkleinen. Het onderzoek is gericht op een aantal aspecten waarvan werd aangenomen dat deze de belangrijkste invloed hadden op de robuustheid van gebarenherkenning. Dit waren: het volgen en voorspellen van bewegingen, de extrac- tie van relevante structuurinformatie uit het beeld, huidskleur detectie, het toevoegen van de derde dimensie van handlocaties, het omgaan met variatie in tijd als de vorm van een gebar en het beperken van het benodigde werk om het systeem een nieuw gebar te leren.

‘Particle filtering’ is een veelgebruikte methode voor het volgen en voorspellen van handbewegingen. Tests met het CONDENSATION algoritme laten echter tegenstrijdigheden zien in het omgaan met verschillende situaties. Wanneer de bewegingen onvoorspelbaar zijn (zoals het geval is bij het volgen van menselijke handen) heeft een particle filter moeilijkheden om het object niet te verliezen. Onder verschillende omstandigheden blijken verschillende strategieën nodig te zijn om hier het beste mee om te gaan.

Isofoot eigenschappen kunnen gebruikt worden als abstracties van een gedeelte van een beeld. Een voordeel van isofoot eigenschappen is, o.a., dat ze onafhankelijk zijn van het contrast van een beeld. In experimenten met gezichtsdetectie op basis van isofoot eigenschappen blijken de resultaten hiermee beter te zijn dan met pixels, gradiënten of de veelgebruikte Haar kenmerken.

Omdat het detecteren van gezichten veel rekenenkracht kost, en de methoden hi-
ervoor minder goed toepasbaar zijn op het detecteren van handen, is het aantrekkelijk om deze lichaamsdelen te kunnen herkennen op basis van hun kleur alleen. In de praktijk gedraagt kleur zich helaas minder voorspelbaar dan wat met een enkelvoudig licht-reflectie model kan worden beschreven. Afwijkingen van fysische modellen voor reflectie worden veroorzaakt door eigenschappen en instellingen van de gebruikte camera, maar ook door de combinatie van verschillende lichtbronnen en reflecties. Door deze onzekerheden te combineren in een algemener model kan meer robuustheid verkregen worden in onbekende omstandigheden. Deze generalisatie gaat helaas ten koste van nauwkeurigheid in gunstige omstandigheden. Om robuustheid te combineren met nauwkeurigheid hebben we een adaptief chromatisch model voorgesteld, dat met een kleine set van metingen de kleurvariatie van een gezicht modelleert met een bimodal, stuksgewijs lineair model in de rood/groen/blauw ruimte.

Gebarentaal vindt plaats in een driedimensionale ruimte, terwijl in beelden slechts tweedimensionale metingen kunnen worden gedaan. Daarom hebben we, met behulp van stereometrie, de metingen van handlocaties in de beelden van twee cameras omgerekend naar de driedimensionale posities van handen in de ruimte. De experimenten laten zien dat deze rijkere informatie inderdaad leidt tot verbetering van herkenning van gebaren. Daarnaast blijkt het perspectief van een enkele grootboekcamera op korte afstand ook een vergelijkbare verbetering te geven. Het nadeel van deze laatste oplossing is echter een verminderde robuustheid, omdat het perspectief erg afhankelijk is van de locatie van een persoon ten opzichte van de camera.

Met behulp van dynamische herkenningsmethoden zoals “Hidden Markov Model” (HMM) of Statistische “Dynamic Time Warping” (SDTW) kan een sequentie van gemeten eigenschappen van een persoon worden herkend als een specifiek gebaar. Deze modellen kunnen omgaan met verschillen in tempo, in tegenstelling tot reguliere methoden van patroonherkenning, die alleen om kunnen gaan met een vaste set van gemeten kenmerken. Een van de nadelen van HMM en SDTW is echter dat ze aan- nemen dat wat belangrijk is voor het vinden van de tijdsvervoering, even belangrijk is voor het herkennen van een klasse. Bovendien zijn ze gebaseerd op factorisatie van kansen voor verschillende tijdstippen, waardoor afhankelijkheden van metingen tussen tijdstappen niet gemodelleerd kunnen worden. Om deze redenen stelden we voor om tijdsregistratie en herkenning van elkaar te scheiden in opeenvolgende stappen. Experimenten laten een significante verbetering zien ten opzichte van HMM of SDTW alleen.

In de praktijk is het moeilijk realiseerbaar om veel voorbeelden van gebaren te krijgen van verschillende personen, om een herkenningsysteem mee te trainen. Om het systeem robuust te maken voor kleine trainingsets hebben we het systeem gebruik laten maken van al geleerde gebaren, waarvoor wel veel voorbeelden beschikbaar waren. Hierbij hebben we aangenomen dat, wanneer een gedeelte van het nieuwe gebaar erg lijkt op een gedeelte van een geleerd gebaar, de variatie ervan op dezelfde manier gedeeldeerd kan worden. Met een enkel voorbeeld als trainingsmateriaal, presteert dit generaliserende systeem even goed als wanneer er vijf voorbeelden gebruikt worden voor de reguliere trainingsmethode.

Uit dit proefschrift kan geconcludeerd worden dat robuustheid niet alleen van toepassing is op praktische applicaties van beeldverwerking, maar ook een plaats
verdient in fundamenteel onderzoek. Het combineren van invalshoeken uit verschillende disciplines, zoals fysica, data analyse, neuropsychologie en mens-machine interactie, zorgt ervoor dat alle aspecten van een beeldverwerkingsproces integraal in beschouwing kunnen worden genomen. Hiermee kunnen robuustere oplossingen worden verkregen dan met elk van de disciplines afzonderlijk.

Samenvatting van het proefschrift: “Gebarenherkenning door middel van Computer Visie: Een Integrale Aanpak”.

J.F. Lichtenauer, Londen, Februari 2009
Acknowledgements

Before and during my PhD I have met many people that contributed to my work and wellbeing, making it possible for me to realize everything that I have achieved. Either by offering their knowledge or advice, by facilitation or support, by sharing interesting and inspiring thoughts, by celebrating successes or mourning hardship together, by sharing leisure times, or, simply, by just being around. I would like to thank some of you specifically.

First of all, my gratitude goes out to my supervisor Emile Hendriks and promotor Marcel Reinders. There has not been even a single moment in which I had wished for better support than you were able to provide. I also really enjoyed working together, whether or not a meeting was efficiently fused with nourishment by cake, lunch, or dinner. Our discussions were a welcome intellectual as well as communicative challenge for me. To my relief, however, I often found myself observing my point of view being explained or defended by Emile to Marcel or vice versa. Not only did this make life much easier for me, these quarrels were often very amusing as well! I’m also grateful to Emile for the motivating image processing lectures and the excursion to LKEB that started my interest for this area of research, and I’m grateful to Marcel for introducing me to the ICT group to do my master of science project.

Secondly, I was very lucky to work together on this project with Jeroen Arendsen and Gineke ten Holt. It has never been more inspiring and enjoyable for me to share and discuss about opposing views, as with Jeroen. Gineke, it has been an invaluable source of support and inspiration to me, to be able to reflect and discuss with you about the technical details of sign language. Furthermore, you have changed my life forever by introducing me to the simple but complicated board game of “Go”. But please excuse me, because I’m actually in the middle of a 19x19 game against my colleague right now, and it is my turn to make a move...

Ioannis Patras, thanks for the help and inspiring discussions about particle filtering. I have also much enjoyed your and Maja’s company in South Korea and San Diego. I still treasure the pleasant memories of both conference visits.

My gratitude goes out to Bart de Ruijscher, who, through his graduation work, gave me more insight into parallel computation of image processing functions on programmable hardware, and to Michiel van Vlaardingen, who laid bare the many difficulties of hand pose recognition, and boldly took them on by graduating on this subject. I have enjoyed the collaboration with Georgi Gaydadjiev, and I am grateful for his support and dedication in the graduation project of Bart. I also thank Onno van
Nierop for his generosity and willingness in sharing his 3D hand model with Michiel, and Aadjan van der Helm, for backing this up with technical support.

Apart from technical collaboration, even more crucial was the teamwork together with the faculty of Industrial Design Engineering, the Dutch Foundation for the Deaf and Hard of Hearing Child and the Royal Auris group. Without it, this thesis would not have been possible at all. The specific aspects of a sign language, the iterative creative process of fusing requirements with technical possibilities, and the problems that are encountered during field-testing with the actual user group, could not have been replaced by any theoretical research or laboratory simulation. Marjan Bruins, Ans van Doorn, Marille Elzenaar, Michel Esveld, Connie Fortgens, Irene Kamp, Hanneke de Ridder, Huib de Ridder, Myrthe van Rossum, Gerard Spaai, Abbie Vanhoutte, Bernadette Vermeij and Elmar Wenners, I feel grateful and warmed, thinking about the time and effort you have put into the ELo project and the material crucial to my research. I want to take this opportunity to also thank all the people who have participated as subjects in the sign language data set, as well as Gineke ten Holt, Jeroen Arendsen and Robbert Eggermont for conducting the recordings.

The ICT group has been a very pleasant and inspiring environment for me to work in during my PhD. Ivo Shterev, thank you for sharing your work, thoughts and your subtle sense of humor with me. Richard Hendriks, I really appreciated the conversations I’ve had with you, I enjoyed sharing the office with you and I was sad to move to a new office without you. Thank you also, Jan Biemond, Bob Duin, Alan Hanjalic, Richard Heusdens, Jesper Jensen, Inald Lagendijk (also for supervising and challenging me with intriguing questions about trivial matters, during my master’s, and for your trust, to which I owe my current career to a great deal), Jan van der Lubbe, Lodewyk Wessels, Andrei, Artsiom, Bart Kroon, Bangjun, Bartek, Carmen, David Tax (I felt reassured to always find you open for discussion), Dick, Feifei, Jan Østergaard, Jenneke, Jeroen de Ridder (I enjoyed the great times we had during and around assisting the student lab works together), Jun Wang, Kosmas, Domenico, Ela, Eugene, Gjenna, Hasan, Ioana, Jan Erkelens, Leo, Maarten Clements, Mark van Staaldruinen, Martin, Miranda, Omar, Pavel, Pim, Peter-Jan, Rogier, Ronald, Sergey, Theo Knijnenburg, Umut, Wan-Jui, Wouter, and Zeki for defining the space in which I could work while enjoying companionship and support.

For administrative and technical support, I thank Hans, Anja, Saskia Peters, Ben van den Boom, and Robbert Eggermont. It was very reassuring to be able to count on your willingness to help and your flexibility in solving problems. I’m also grateful to Andre Redert, for sharing his experience and the video equipment he used for his own project.

During my PhD research and the writing up, I have crossed paths with Michel Valstar, Antonis Oikonomopoulos and Maja Pantic. I have enjoyed your discussions, support and company and I am looking forward to continuing this in our time ahead together in London!

The success of a PhD often depends highly on the support of friends and family. This has been especially the case for me. It created a space in which I could find, develop and be myself, receive support and enjoy distraction.

My warm gratitude goes out to Yunlei. Having you as a companion has added a lot
of smiles to my life. You have also given me a taste of Chinese culture (I have now learned that Chinese people do like cheese! ;-) and I have enjoyed your unconditional support and your amazing way of fusing humor with self-expression. Xin and Jason, I wish you lot’s of happiness together with Nini, born close to our birthdays. I’m looking forward to celebrating more shared birthdays in the future!

Frank, your dedication and self-assurance in not aiming for, but simply shooting at targets that seem over-ambitious in time as well as in height, has been a subject of my astonishment, amazement, frustration, as well as admiration. Observing that such an approach can actually reach goals, now and then, has been an important lesson for me, as someone who has mainly been occupied by aiming instead of shooting. One of the great achievements resulting from this learning experience has been the successful 1000 meter climb onto the rocky summit of Skye. This was also the place and time where we learned that aiming for the sky carries the risk that the clouds hanging up there may obscure the view... Sometimes, you just have to shoot again. Looking forward to the next attempt! I also wish you and Femke a happy life together.

Matthijs, je advies om zo ver mogelijk door te gaan in mijn opleidingstraject is blijkbaar niet voor niets geweest. Al denk ik dat leren pas echt begint wanneer je niet meer afhankelijk bent van wat anderen je vertellen. Bedankt voor je steun, je duurzame vriendschap, en de prachtige eenvoud van je adviezen en suggesties. Laura, heel erg bedankt voor je warme hartelijkheid. Ik kijk er naar uit om je beter te leren kennen in de toekomst.

Alexandra, it has been a great experience and adventure to get to know you. Seeing things from your perspective helped me to look with less judgement. The way you inspire people and act as a mirror, by just being yourself, is touching and unique. Thank you for sharing this with me, and for seeing me.

The consciousness meetings have been among the highlights during my PhD. Discovery of the human being (including myself) is something that has long been an interest to me. To be able to explore this topic together with others has been a very welcome source of inspiration, learning and connection. Carmen, Pavel, Bob, Dick, David, Yunlei, Ela, Marina, Wan-Jui, Sergey, Marco, Karthick, Theo and, Henriëtta, thank you for sharing, discussing and connecting on this topic.

In the same line, the last year has involved particularly interesting new discoveries. I will be ever grateful to Taco, who has introduced me to the beautiful form of interacting and being, called ‘Nonviolent Communication’ (NVC). And, equally, to Beauchamp, for bringing together, facilitating and guiding me and others, to practice with and connect through NVC. This has been an important inspiration for a few of my propositions.

Saskia, I am very grateful for the trips and adventures you took me on in a far distant past. For what I remember, I enjoyed them a lot and I was lucky to have such an adventurous sister. Currently, I feel humbled when I think of everything you must have had to overcome. I hope you can find a new place of stability and predictability, but also experience challenge, adventure and creativity, without feeling overwhelmed. Your strength to let problems disappear when enjoying your interests is admirable and your humor, intelligence and unique way of thinking are a pleasure and an inspiration to me.
Mirjam, thanks for being a guide, when we were following the same educational path. Your advice and support have helped me a lot during that period. I am proud of your bold choice recently, to take the initiative in a career step. I am happy to see you have finally found a satisfying job that challenges you in a constructive way. As sad I was to see Hanneke de Ridder leave the NSDSK and the ELo project, I am happy now to share her with you as your new employer! Funny how often coincidence seems to be involved in the most significant parts of life.

Fred, thank you for your support, encouragement and appreciation of any challenge, progress or achievement that I share with you as your son and friend, while leaving me free in my choices. I feel I can always count on you to be willing to talk about anything.

Thank you Bettie, for your unconditional love, care and support as my mom, and also for the unrivaled depth of our regular discussions on philosophy and psychology. I wish that anyone you meet could see the warmth, intelligence and humor that waits inside for those who take the effort to look for it. I am lucky to experience it, learn from it and get inspired and motivated by it.
Curriculum Vitae


Between 2003 and 2008 he conducted his PhD studies, at same research group, under the supervision of Emile Hendriks and Marcel Reinders. The topic of this PhD was robustness in gesture recognition by computer vision. During his PhD, he assisted in student laboratories and industrial courses and co-supervised several students in their graduation project.

Between January and March 2003, Jeroen has been employed by Operator Group Delft (OGD), to improve the accuracy and robustness of image processing methods used in the Automatic Curve Extraction (ACE) software that was under development at the Royal Netherlands Meteorological Institute (KNMI). The goal of the ACE project is to automatically extract large amounts of historical meteorological data from the charts from meteorological instruments such as tipping bucket rain gauge recorders and barometers.

Since April 2008, Jeroen has been working as a postdoctoral researcher in the Visual Information Processing group, within the department of Computing of Imperial College London. His work focusses on obtaining robust, real-time tracking of poses and dynamics of the human face. This research is part of the MAHNOB project, funded by the European Research Council (ERC). The aim in MAHNOB is to create a system for automatic analysis of human naturalistic behavior.

List of Publications

The following publications have resulted from Jeroen’s master’s and PhD studies:
Journals


Conferences


Master’s Theses

