Eye Tracking

Denis Leimberg
Martin Vester-Christensen

LYNGBY 2005
Master Thesis IMM-Thesis-2005-8

IMM

Printed by IMM, Technical University of Denmark

Preface

This thesis has been prepared over six months at the Section of Intelligent Signal Processing, Department of Informatics and Mathematical Modelling, at The Technical University of Denmark, DTU, in partial fulfillment of the requirements for the degree Master of Science in Engineering, M.Sc.Eng.

The reader is assumed to possess a basic knowledge in the areas of statistics and image analysis.

Kgs. Lyngby, February 2005.

Martin Vester-Christensen and Denis Leimberg
[martin@kultvizion.dk and denis@kultvizion.dk]

Acknowledgements

This thesis would never have amounted to what it is without the invaluable help and support from the following people.

First and foremost, we thank out project supervisor Lars Kai Hansen for his ability to motivate, explain complicated matters and enormous patience with two not-so-bright students. His dedication towards his students is rare and greatly appreciated.

Bjarne k. Ersbøll, co-supervisor for his enthusiasm, and for rubbing of professionalism to two novices in the great world of scientific publications.

Hans bruun Nielsen, for support during our entire education and providing a top-tuned, ultra-fast optimizer.

Mikkel B. Stegmann, for always sparing a moment, and for helping out without reservations.

Henrik Aanæs, yet another from the Image Analysis section, providing help and assistance.

Dan Witzner Hansen, for inspiring and insightful discussions, and for introducing the world of eye tracking.

Kaare Brandt Petersen, for sparing a day, during his busy ph.d.-finalizing period, to provide for hairy mathematical derivations.

Martin E. Nielsen, for proof reading in the eleventh hour.

Martin Rune Andersen, an exclusive membership of the lunch club, providing us with excellent poker tricks.

Peter Ahrendt, for loosing in Hattrick - Thank you for volunteering.

Sumo Volleyball, by Shizmoo games, for letting the temperaments run hot over the internet and not in the office.

Tue Lehn-Schiøler, for stepping up in time of need and provide an emergency coffee maker.

Maïa Weddin, a project room companion, for tolerance and injecting guilty conscience, with the sound of fast keyboard typing. Thank you for your high spirits.

The COGAIN Network of Excellence - The reason for the project. Thank you for inspiration - and a fabulous dinner.

We would like to thank the whole Intelligent Signal Processing Section for providing a pleasant and inspiring atmosphere, with interesting discussion during lunch.

Most important of all, our families for love and support. Bettina for putting up when the times got rough. Mathilde, for practically being without a father for two months, and Malene for understanding.

Abstract

This thesis presents a complete system for eye tracking avoiding restrictions on head movements. A learning-based deformable model - Active Appearance Model(AAM) - is utilized for detection and tracking of the face. Several methods are proposed, described and tested for eye tracking, leading to determination of gaze.

The AAM is used for a segmentation of the eye region, as well as providing an estimate of the pose of the head.

Among several, we propose a deformable template based eye tracker, combining high speed and accuracy, independently of the resolution. We compare with a state of the art active contour approach, showing that our method is more accurate.

We conclude, that eye tracking using standard consumer cameras is feasible providing an accuracy within the measurable range.

Keywords:

Face Detection, Face Tracking, Eye Tracking, Gaze Estimation, Active Appearance Models, Deformable Template, Active Contours, Particle Filtering.

Resumé

Denne afhandling prænsenterer et komplet eye tracking system, som undgår restriktioner med hensyn til hovedbevægelser. En datadrevet statistisk model - Active Appearance Model(AAM) - benyttes til detektion og tracking af ansigtet. En række forskellige eye tracking metoder foreslås, beskrives og testes. Dette fører til bestemmelse af blikretning.

Regionen omkring øjet udtrækkes vha. af AAM'en. Ligeledes fås et estimat af hovedets retning.

Blandt flere metoder foreslås en eye tracker baseret på deformable templates, som kombinerer høj hastighed og præcision uafhængigt af opløsning. Vi sammenligner med en *state of the art* aktiv kontur metode. Vores metode er mest præcis.

Vi konkluderer at standard kameraer er fuldt tilstrækkelige til formålet eye tracking. Præcisionen er indenfor usikkerheden af det målbare område.

Nøgleord:

Ansigt Detektion, Ansigt Tracking, Eye Tracking, Blikretning, Active Appearance Modeller, Deformable Template, Aktive konturer, Partikel Filtrering.

Contents

1 Introduction to Eye Tracking **17**

2 Eye Tracking Systems **23**
2.1 IR Eye Trackers 24
2.2 IR Free Eye Trackers 25
2.3 Commercial systems 26

3 Motivation and Objectives **27**
3.1 Thesis Overview 27
3.2 Nomenclature 28

I Face Detection and Tracking **31**

4 Introduction **33**
4.1 Recent Work 33
4.2 Overview 34

5 Modeling Shape **35**
5.1 Aligning the Training Set 35
5.2 Modeling Shape Variation 37
5.2.1 Principal Component Analysis 37
5.2.2 Choosing the Number of Modes 38
5.2.3 Low Memory PCA 38
5.3 Creating Synthetic Shapes 39
5.4 Summary 40

6 Modeling Texture **43**
6.1 Building the Model 43
6.2 Image Warping 44
6.3 Modeling Texture Variation 45
6.4 Summary 47

7 The Independent Model **49**
7.1 Defining the Independent Model 49
7.2 Summary 50

8 The Inverse Compositional Algorithm **51**
8.1 Introduction 51
8.2 The Gauss-Newton Algorithm 52
8.3 The Lucas-Kanade Algorithm 53
8.4 The Inverse Compositional Algorithm 55
8.5 Including Appearance Variation 56
8.6 Summary 58

9 Fitting the Active Appearance Model **59**
9.1 Computing the Warp Jacobian 59
9.1.1 The Shape Jacobians 60
9.1.2 The Parameter Jacobians 61
9.1.3 Assembling the Warp Jacobian 61
9.1.4 Steepest Descent Images 61
9.1.5 The Parameter Update 63
9.2 Warp Inversion 63
9.3 Warp Composing 65
9.4 Including a Global Shape Transform 65
9.4.1 Warping 66
9.4.2 Computing the Jacobian 67
9.4.3 Warp Inversion 67
9.4.4 Warp Composition 68
9.4.5 Appearance Variation 68
9.5 The AAM Search 68
9.6 Summary 70

10 Extracting Head Pose **71**
10.1 Computing 3D Shape from an AAM 71
10.2 Summary 73

11 Discussion **75**
11.1 Forces 75
11.2 Drawbacks 75

II Eye Tracking 77

12 Introduction 79
12.1 Recent Work 79
12.2 Overview 81

13 Segmentation-Based Tracking 83
13.1 Thresholding 83
13.1.1 Double Threshold 84
13.2 Template Matching 85
13.2.1 Iris Tracking 87
13.2.2 Outlier Detection 88
13.3 Color-Based Template Matching 90
13.4 Deformable Template Matching 92
13.4.1 Optimization 94
13.4.2 Constraining the Deformation 95
13.5 Summary 97

14 Bayesian Eye Tracking 101
14.1 Active Contours 101
14.2 Assumptions 101
14.3 Dynamic Model 102
14.4 Observation Model 103
14.4.1 Statistics of Gray-Level Differences 103
14.4.2 Distributions on Measurement Lines 104
14.4.3 Marginalizing over Deformations 105
14.5 Probabilistic Contour Tracking 107
14.6 Constraining the Hypotheses 108
14.7 Maximum a Posteriori Formulation 110
14.8 Optimization by EM 110
14.8.1 Motivation 111
14.8.2 Applying EM 111
14.9 Optimization by Deformable Template Matching 114
14.10Parameters of the Method 114
14.11Summary 115

15 Gaze Determination 117
15.1 The Geometric Model 117

16 Discussion **121**
16.1 Segmentation-Based Tracking 121
16.2 Bayesian Eye Tracking 122
16.3 Comparison 122

III Experimental Results **125**

17 Experimental Design **127**
17.1 System Overview 127
17.1.1 Camera 128
17.1.2 Computer Interface 128
17.1.3 Face Detection and Tracking 128
17.1.4 Eye Tracking 129
17.1.5 Gaze Determination 129
17.2 System 129
17.3 Data 130
17.4 Algorithm Evaluation 130
17.5 Overview 130

18 Face Detection and Tracking **131**
18.1 Constructing the AAMs 131
18.2 Convergence 133
18.2.1 The Average Frequency of Convergence 133
18.3 Tracking 138
18.4 Discussion 140
18.4.1 Convergence 140
18.4.2 Tracking 142
18.5 Improvements 142
18.5.1 Summary 143

19 Eye Tracking **145**
19.1 Performance of Segmentation-Based Eye Trackers 147
19.2 Performance of Bayesian Eye Trackers 149
19.2.1 Ability to Handle Eye Movements 152
19.3 Comparison of Segmentation-Based and Bayesian Eye Trackers 153
19.3.1 Influence of Gaze Direction 154
19.3.2 Human Computer Interaction 157
19.4 Gaze Estimation 158

19.5 Discussion 159
19.5.1 Interpretation of Performance 160

IV Discussion and Future Work 163

20 Summary of Main Contributions 165
20.1 Face Detection and Tracking 165
20.2 Eye Tracking 166

21 Propositions for Further Work 169

22 Conclusion 171

A Face Detection and Tracking 173
A.1 Piecewise Affine Warps 173

B Bayesian Eye Tracking 177
B.1 Bayesian State Estimation 177
B.1.1 Particle Filtering 177
B.2 Derivation of the Point Evaluation Function 178
B.3 The EM Algorithm 179
B.4 EM Active Contour algorithm 181

C Heuristics for Speeding Up Gaze Estimation 183

D Towards Emotion Modeling 189

Chapter 1

Introduction to Eye Tracking

Figure 1.1: "The authors", Photo: Bo Jarner.

Every day of life, most people use their eyes intensively for perceiving, learning, reading, watching, navigating etc. Despite the seeming ease with which we perceive the world around us, visual perception is actually a complex process that occurs at a level below conscious awareness. The light structure seen by the eye is continuously sampled, causing the eyes to move in order to make the next important light structure sample. The brain attempts to make sense of the information obtained. In this way, we perceive the scene.

The task at hand is creating a technique used to determine where a person is looking - Gaze direction. The dictionary[14] defines *gaze* as;

"To view with attention."

The concepts underlying eye tracking are to track the movements of the eyes and determine the gaze direction. This is, however, difficult to achieve and sophisticated data analysis and interpretation are required.

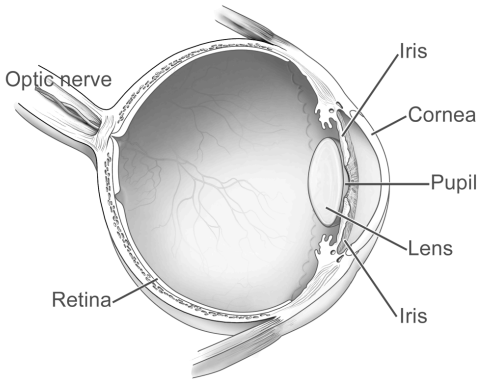


Figure 1.2: The structure of the eye. An excellent website containing abundance of eye related information can be found at National Eye Institute[41].

Eye movements during reading and image identification provide useful information about the processes by which people understand visual input and integrate it with knowledge and memory. Eye tracking is exploited for adult or child psychology studies, human-machine interfaces, driver awareness monitoring to improve traffic safety etc.

Eye trackers enables to determine the direction of gaze, but unfortunately not whether users actually "see" something - e.g. if daydreaming.

"You can't depend on your eyes when your imagination is out of focus."

- Mark Twain

A vast amount of research has been put into eye tracking leading to a variety of methods and different applications. In the following, examples of applications are given - and even more can be imagined. Subsequently, a more technical description is given in chapter 2.

Figure 1.3: Which shampoo do you look at first?[5]

Marketing

Which objects attract the attention of customers, is of great interest for market researchers; what shelves and which products are catching the shoppers' attention in supermarkets, and what images or written words are viewed while flipping through a magazine.

Web page designers are interested in what a viewer read, how long they stay on a particular page, and which page they view next. An experiment is shown in figure 1.4.

Figure 1.4: During an experiment a number of persons were asked to view the image and then report what information they could expect to find on this website. Analysis of eye-tracking data suggests users first fixate on graphics and large text even when looking for specific information[51].

A great deal of research is in the field of TV advertisers - which images grab the viewers' attention, and which are ignored. Maybe, even more focus should be put into this field?

Disabled people

The quality of life of a disabled person can be enhanced by broadening his communication, entertainment, learning and productive capacities. By looking at control keys displayed on a computer monitor screen, e.g. as seen in figure 1.5, the user can perform a broad variety of functions including typing, playing games, and running most Windows-compatible software.

Figure 1.5: Example of human computer interaction[89].

Simulator

The attention of e.g. airplane pilots can be investigated utilizing eye tracking. Experienced pilots develop efficient scan patterns, where they routinely look at critical instruments and out of the cockpit. An eye tracker can assist instructors to determine whether the student pilots are developing good scan patterns, and whether their attention is at the right places during landing or in emergency situations.

Similar systems are useful for determining driver awareness as illustrated in figure 1.7.

Figure 1.6: (*Left*) The attention of airplane pilots can be investigated utilizing eye tracking[42]. (*Right*) Eye tracking can be utilized to aid pilots in their weapons control while flying[33].

Defence

Eye tracking can be exploited in various applications in the defence industry. One of the main purposes is to aid pilots in their weapons control. Thus allow the pilots to observe and select targets with their eyes while flying the plane and firing the weapons with their hands.

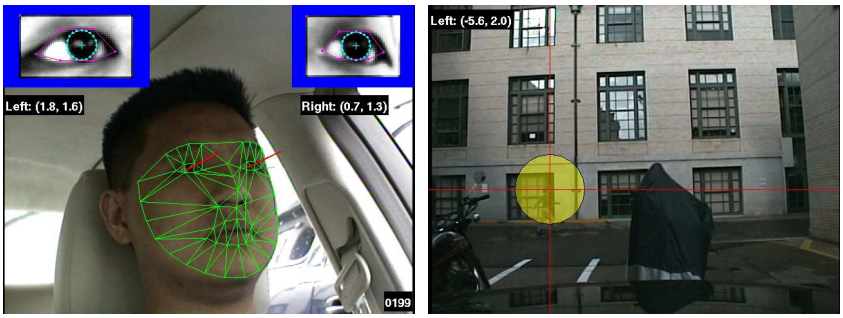


Figure 1.7: Driver awareness[43]. (*Left*) The driver's gaze is mapped into (*right*) an external scene

Robot-Human Interaction

The gaps in communication between robot and human can be bridged. Does the human actually communicate with the robot or someone else? What is holding their attention? What does the human want the robot to interact with?

Video Games

Eye tracking will add a new dimension onto video games in the future. Identify the threat, acquire the target, move the scene right or left, etc.

Chapter 2

Eye Tracking Systems

A vast amount of research has been put into eye tracking, leading to a variety of methods. The problem is definitely not a trivial task, and the methods used depend highly on the individual purpose.

Recording from skin electrodes[44] is among the simplest eye tracking technologies. This method is useful for diagnosing neurological problems. A very accurate, but uncomfortable, method utilizing a physical attachment to the front of the eye - a contact lens.

Figure 2.1: Head mounted eye tracker[53].

One of the main difficulties is to compensate for head movements. As a consequence, a headrest or a head mounted eye tracker[76], as seen in figure 2.1, can be exploited. The disadvantages are a restriction in movement and the bulky equipment. For laboratory experiments, the method may be feasible, but for long term use by, for instance disabled people, a less intrusive method is preferable.

To reduce the level of intrusion on the user, a remote camera setup can be used. However, this reduce the resolution of the eyes. Camera-based eye tracking can be classified on whether infrared (IR) light is used or not. IR and Non-IR eye tracking systems from the literature, are are described in section

2.1 and 2.2, respectively. Finally, we present an overview of commercial systems in section 2.3.

2.1 IR Eye Trackers

Infrared illumination along the optical axis, at a certain wavelength, results in an easily detectable bright iris. The pupil reflects almost all received IR light back to the camera, producing the bright pupil effect as seen in figure 2.2(left). This is analogous to the red eye effect in photography[45].

Ohno et al. presents a remote gaze tracking system using a single camera and on-axis IR light emitters[62]. The gaze position is computed given the two estimated pupil centers utilizing an eyeball model.

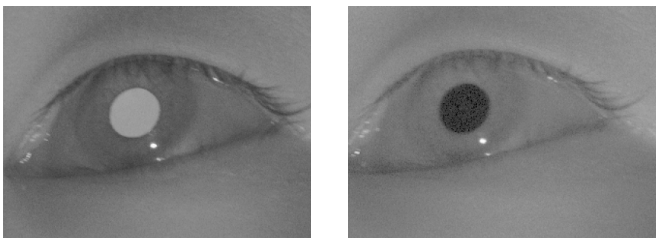


Figure 2.2: IR illuminated eyes [58]. *(1)* Bright pupil image generated by IR illumination along the optical axis. *(2)* Dark pupil image generated by IR illumination off the axis.

Illumination from an off-axis source generates a dark pupil image as seen in figure 2.2(right). The combination of on-axis and off-axis illumination is utilized by Ji and Yang[45], Morimoto et al.[58], Zhu et al.[101]. In the detection step, the images are subtracted to obtain a difference image, which is thresholded and connected components are applied to filter out noise. Zhu et al.[101] uses a combination of Kalman filtering and mean shift tracking.

The gaze precision is dependent on the eye resolution, which can be improved by a close up image of the eye. Perez et al. presents a remote gaze tracking system combining a wide field of view face camera and a narrow field of view eye camera illuminated by four infrared light sources[67]. In this way, the resolution of the eye is kept high, while ensuring robustness regarding head movements.

Multiple cameras are applied frequently in the literature to estimate the 3D pose of the head, improving the precision of gaze. Ohno et al. propose a remote gaze tracker, combining a stereo camera set for eye detection and an IR camera for gaze tracking[61]. The two systems run independently, controlled by two connected PC's. Eye position data is sent to the gaze

tracking unit on request. Talmi and Liu use three cameras[86] - two static face cameras for stereo matching of the eyes, and one camera focusing on one of the viewer's eyes. In order to find both eyes of the two head cameras, the principal component analysis technique is applied - analogous to eigenfaces in the litterature[23]. Head movements are compensated by utilizing the 3D pose obtained from stereo matching. Ruddarraju et al. [68] propose a vision-based eye tracking system from multiple IR cameras. The eye tracking is utilized by a Kalman filter, while Fisher's Linear discriminant is used to construct appearance models for the eyes. The 3D pose is estimated by a combination of stereo triangulation, interpolation and a camera switching method to use the best representations.

2.2 IR Free Eye Trackers

A remote eye tracker using a neural network to estimate the gaze is presented by Stiefelhagen et al.[81]. Smith et al. presents a system for analyzing driver visual attention[75]. In [43] Ishikawa et al. describes a system for driver gaze tracking using a single camera setup. The entire face region is modeled with an Active Appearance Model, which is used to track the face from frame to frame. Gaze is determined by a geometric model.

Detection of the human eye is a difficult task due to a weak contrast between the eye and the surrounding skin. As a consequence, many existing approaches uses close-up cameras to obtain high-resolution images. Hansen and Pece[36] presents an active contour model combining local edges along the contour of the iris. However, this imposes restrictions on head movements. Analogous to IR based trackers, multiple cameras are applied in many existing approaches improving the precision of gaze estimate. Wang and Sung uses two cameras[92]. One camera is a global camera covering of the entire head used to determine the pose of the subjects head. The head pose controls a second camera, which focuses on one eye of the person. They claim higher accuracy as a result of this setup. Xie et al. presents a method utilizing two Kalman filters[97]; one with purpose to track the eyes and one which compensates for head movements. Matsumoto and Zelinsky propose a tracker based on template and stereo matching[56]. Facial features are detected by using templates, and are subsequently used for 3D stereo matching. The performance of the gaze direction measurement are reported to be excellent. However, each user initially has to register face and feature points.

2.3 Commercial systems

A mouse replacement device allowing the user to move the mouse pointer anywhere on the screen, by looking at some location, is developed by Eyetech Digital Systems[84]. "Clicking" can be done with an eye blink, a hardware switch, or by staring (dwell). The eyes are illuminated from two off-axis IR light sources resulting in an easily detectable dark pupil.

Tobii Technology[89] exploits IR and a wide-field-of-view high resolution camera. This is integrated into a TFT monitor as shown in figure 1.5. Similar methods are developed by Eye Response Technologies[87] and LC Technologies [88]. A performance evaluation comparison of Tobii and LC technologies eye trackers can be found in [17].

Smart Eye AB[74] has designed a system capable of utilizing IR with multiple cameras - up to four. The method is able to continue tracking even though one camera is fully occluded. While the face is being tracked, gaze direction and eyelid positions are determined by combining image edge information with 3D models of the eye and eyelids.

SensoMotoric Instruments specializes in the development of ergonomic chin rest, head mounted and remote systems[42]. Applied Science Laboratories has also a wide range of products[50].

Seeing Machines is engaged in the research, development and production of advanced computer vision systems for research in human performance measurement, advanced driver assistance systems, transportation, biometric acquisition, situational awareness, robotics and medical applications[71].

Chapter 3

Motivation and Objectives

> *"What if eye trackers could be downloaded and used immediately with standard cameras connected to a computer, without the need for an expert to setup the system?"*
>
> - D.W. Hansen et al.[37].

If the above would ever become true, then everyone could be in possession of eye tracking systems. However, more work need to be done. Many methods has been developed, as mentioned above, nevertheless suffering from subjects as restrictions on freedom of movement, poor image resolution, discomfort using multiple cameras, expensive IR equipment etc.

Thus, the main objectives set forth was to:

> *Develop a fast and accurate eye tracking system enabling the user to move the head naturally in a simple and cheap setup.*

3.1 Thesis Overview

The interpretation of the main objective, naturally divides the problem of eye tracking into three components - Face detection and tracking, eye tracking, and gaze determination. Additionally, to achieve a simple and cheap setup, we restrict ourselves to use a standard digital video camcorder.

The thesis is structured into four parts, where each part requires knowledge from the preceding parts.

Part I: Face Detection and Tracking Presents a statistical method to overcome the problem of tracking a naturally moving head.

Part II: Eye Tracking Presents several tracking methods - segmentation-based and bayesian - applied on the eye image obtained from part I. Combining information from the statistical method and pupil location, enables for gaze determination.

Part III: Experimental Results Evaluation of performance and problems of the system.

Part IV: Discussion and Future Work Finally, possible extensions are discussed and the thesis work is concluded.

Some of the techniques and preliminary results are found in abbreviated form in papers prepared during the thesis period[52]. The two papers are attached as appendix C and D.

3.2 Nomenclature

To ease understanding the mathematics, variables without an explicit denotation conform to the nomenclature below.

I	An image.
T	An image template.
λ	Length of the axes defining an ellipse.
c_x	Center of ellipse, x-coordinate.
c_y	Center of ellipse, y-coordinate.
ϕ	Orientation of ellipse or gaze direction.
θ	Orientation of head pose.
$\mathcal{E}$	Cost function regarding deformable template model.
$\mathcal{M}$	Measurement line along the contour.
ν	Coordinates on the measurement line.
μ	Position of the boundary regarding a specific contour.
ϵ	Deformation of the contour.
$\mathbf{g}$	A vector of image intensities.
$\mathbf{g}_0$	Intensity vector of the mean texture.
$\mathbf{s}$	A vector of vertex coordinates.
$\mathbf{s}_0$	The coordinate vector for the mean shape.
$\mathbf{\Phi}_s$	Matrix of shape eigenvectors.
φ_{s_i}	The i'th shape eigenvector.
$\mathbf{\Phi}_g$	Matrix of texture eigenvectors.
φ_{s_i}	The i'th texture eigenvector.
$\mathbf{b}_s$	A vector of shape parameters.
b_{s_i}	The i'th shape parameter.
$\mathbf{b}_g$	A vector of texture parameters.
b_{g_i}	The i'th shape parameter.
$\mathbf{x}$	A state vector or the coordinate x_i, y_i of the i'th pixel inside a convex hull.
$\mathbf{W}(\mathbf{x}; \mathbf{b}_s)$	A warp of the pixel at $\mathbf{x}$, defined by the relationship between a shape $\mathbf{s}$ and the mean shape $\mathbf{s}$, given by $\mathbf{b}_s$.

Part I

Face Detection and Tracking

Chapter 4

Introduction

A number of eye trackers available today, assumes very limited movement of the head. This may be tolerable for short periods of time, but for extended use, not being allowed to move the head is very uncomfortable. If the eye tracking system is to be a part of a driver awareness system, head movements should not only be allowed, they should be encouraged.

Allowing the user to move his/her head, requires that the system is able to track its movement and pose. This is the topic of this part of the thesis.

4.1 Recent Work

In recent years, several techniques have been proposed for head tracking and 3D pose recovery.

An approach is to use distinct image features. In [18] Choi et al. estimate the facial pose by fitting a template to 2D feature locations. The parameters of the fit are estimated using the EM algorithm. Shih et al.[73] presents a face extraction method based on double threshold and edge detection using Gabor filters. They work well when the features are reliably tracked over the image sequence.

When good feature correspondence are not available, utilizing the texture of the entire head is more reliable. A remote eye tracker using a neural network to estimate the gaze is presented by Stiefelhagen et al.[81]. The face is tracked by use of a statistical color-model consisting of a two-dimensional Gaussian distribution of normalized skin colors. Zhu et al.[100] combines appearance using principal component analysis with 3D head motion estimation using optical flow. In [49] Cascia et al. proposes a fast 3D head tracker based on, models of the head as a texture mapped cylinder. Tracking is formulated as an image registration problem. Ba et al.[7] views the head tracking and

pose estimation as a coupled problem. They claim reduce sensitivity of the pose estimation on the tracking accuracy, which leads to more accurate pose estimates.

Face detection has received quite a bit of attention in recent years. Especially in the field of face recognition. A very successful class of methods for face detection are the Active Appearance Models. An active appearance model is a non-linear, generative, and parametric model of an object[57]. Several head tracking approaches uses an active appearance model. Noticeably is Dornaika et al.[24][26][25] uses a parameterized 3D active appearance model for tracking the head and facial features. They combine it with a Kalman filter for prediction and report excellent results.

In [43] Ishikawa et al. presents an eye tracking system for driver awareness detection. They utilize an Active Appearance Model, recently proposed by Matthews and Baker[57], which is very fast and reliable. It has the added feature of providing the head pose while tracking.

4.2 Overview

In this part the head tracking and pose estimator is presented. It is responsible for finding and extracting the region of the eyes, and provides the head pose part of the gaze direction.

It utilizes an algorithm called and Active Appearance Model. It is used to create a statistical model of faces, and can be used to find and track the head.

Recently Matthews and Baker introduced a new more effective Active Appearance Model, and the bulk of this part is used to introduce and describe this model. First statistical models of shape and texture are introduced. Then a way to fit these models to images using general non-linear optimization is described. Finally, extraction of pose parameters from the fitted model is covered.

Chapter 5

Modeling Shape

A *shape* is defined as;

> *"... that quality of a configuration of points which is invariant under some transformation."*
>
> - Tim Cootes[21]

In this framework of face detection and tracking, shape is defined as n 2D points, *landmarks*, spanning a 2D mesh over the object in question. The landmarks are either placed in the images automatically[12] or by hand. Figure 5.1 shows an image of a face[80] with the annotated shape shown as red dots. Mathematically the shape $\mathbf{s}$ is defined as the $2n$-dimensional vector of coordinates of the n landmarks making up the mesh,

$$\mathbf{s} = [x_1, x_2, \ldots, x_n, y_1, y_2, \ldots, y_n]^T . \tag{5.1}$$

Given N annotated training examples, we have N such shape vectors $\mathbf{s}_i$, all subject to some transformation. In 2D the usual transformation considered is the Similarity Transformation(rotation, scaling and transformation). We wish to obtain a model describing the inter-shape relations between the examples, and thus we must remove the variation given by this transformation. This is done by aligning the shapes in a common coordinate frame as described in the next section.

5.1 Aligning the Training Set

To remove the transformation, ie. the rotation, scaling and translation of the annotated shapes, they are aligned using iterative Procrustes analysis[21]. Figure 5.2 show the steps of the iterative Procrustes analysis. The top figure

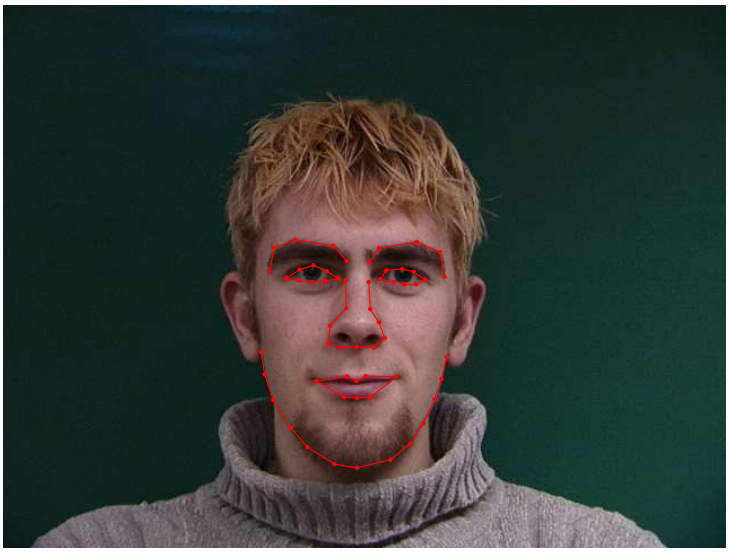


Figure 5.1: Image of a face annotated with 58 landmarks[60].

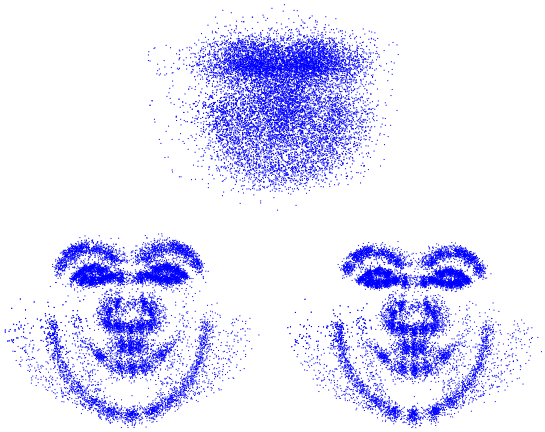


Figure 5.2: Procrustes analysis. The top figure shows all landmark points plotted on top of each other. The lower left figure shows the shapes after translation of their centers of mass, and normalization of the vector norm. The lower right figure is the result of the iterative Procrustes alignment algorithm.

show all the landmarks of all the shapes plotted on top of each other. The lower left figure show the initialization of the shape by the translation of their centers of mass and normalization of the norm of the shape vectors. The lower right figure is the result of the iterative Procrustes algorithm.

The normalization of the shapes and the following Procrustes alignment results in the shapes lying on a unit hypersphere[21]. Thus the shape statistics will have to be calculated on the surface of this sphere. To overcome this problem, an approximation, that the shapes lie on the tangent plane to the hypersphere, is made. Thus ordinary statistics can be used. The shape $\mathbf{s}$ can be projected onto the tangent plane at the mean using,

$$\mathbf{s}' = \frac{\mathbf{s}}{\mathbf{s}^T \mathbf{s}_0}, \tag{5.2}$$

where $\mathbf{s}_0$ is the estimated mean shape given from the Procrustes alignment.

With the shapes aligned in a common coordinate frame it is now possible to build a statistical model of the shape variation in the training set.

5.2 Modeling Shape Variation

The result of the Procrustes alignment is a set of $2n$ dimensional shape vectors $\mathbf{s}_i$ forming a distribution in the space in which they live. In order to generate shapes, a parameterized model of this distribution is needed. Such a model is of the form $\mathbf{s} = M(\mathbf{b})$, where $\mathbf{b}$ is a vector of parameters of the model. If the distribution of parameters $p(\mathbf{b})$ can be modeled, constraints can be put on them such that the generated shapes $\mathbf{s}$ are similar to that of the training set. With a model it is also possible to calculate the probability $p(\mathbf{s})$ of a new shape.

5.2.1 Principal Component Analysis

To constitute a shape, neighboring landmark points must move together in some fashion. Thus some of the landmark points are correlated and the true dimensionality may be much less than $2n$. Principal Component Analysis(PCA) rotates the $2n$ dimensional data cloud that constitutes the training shapes. It maximizes the variance and gives the main axis of the data cloud.

The PCA is performed as an eigenanalysis of the covariance matrix, $\mathbf{\Sigma}_s$, of the training data.

$$\mathbf{\Sigma}_s = \frac{1}{N-1}\mathbf{S}\mathbf{S}^T, \tag{5.3}$$

where N is the number of training shapes, and $\mathbf{S}$ is the $n \times N$ matrix $\mathbf{S} = [\mathbf{s}_1 - \mathbf{s}_0, \mathbf{s}_2 - \mathbf{s}_0 \ldots \mathbf{s}_N - \mathbf{s}_0]$. $\mathbf{\Sigma}_s$ is a $n \times n$ matrix. Eigenanalysis of the $\mathbf{\Sigma}_s$

matrix gives a diagonal matrix $\mathbf{\Lambda}_l$ of eigenvalues λ_i and a matrix $\mathbf{\Phi}_l$ with eigenvectors ϕ_i as columns. The eigenvalues are equal to the variance in the eigenvector direction.

PCA can be used as a dimensionality reduction tool by projecting the data onto a subspace which fulfills certain requirements, for instance retaining 0.95% of the total variance or similar. Then only the eigenvectors corresponding to the t largest eigenvalues fulfilling the requirements are retained. This enables us to approximate a training shape instance $\mathbf{s}$ as a deformation of the mean shape by a linear combination of t shape eigenvectors,

$$\mathbf{s} \approx \mathbf{s}_0 + \mathbf{\Phi}_s \mathbf{b}_s \tag{5.4}$$

where $\mathbf{b}_s$ is a vector of t *shape parameters* given by

$$\mathbf{b}_s = \mathbf{\Phi}_s^T (\mathbf{s} - \mathbf{s}_0), \tag{5.5}$$

and $\mathbf{\Phi}_s$ is the matrix with the t largest eigenvectors as columns.

5.2.2 Choosing the Number of Modes

The simplest way to find the number of modes, t, is to chose the number of eigenvectors explaining a percentage of the total variance of the training set. Since total variance is the sum of all eigenvalues λ_i, the largest t eigenvalues can be chosen such that[21],

$$\sum_{i=1}^{t} \lambda_i \geq \alpha \sum_{i=1}^{2n} \lambda_i \tag{5.6}$$

A second way is to choose t from the study of how well the model approximates the training examples. Models are built with an increasing number of modes. This can be further refined by using a Leave-One-Out test scheme, where one of the examples are retained and the model is trained on the rest. The best approximation by the current model to the test shape, is then calculated using (5.4) and (5.5). The quality of the approximation is calculated as the mean Euclidean distance between the test shape and the approximation. This is repeated, retaining each shape as a test shape. The level for which the total error is below a threshold, is the number of eigenvectors, t, to be used.

5.2.3 Low Memory PCA

Consider the $N \times N$ matrix $\mathbf{\Sigma}_s$

$$\mathbf{\Sigma}_s = \frac{1}{N-1} \mathbf{S}^T \mathbf{S}. \tag{5.7}$$

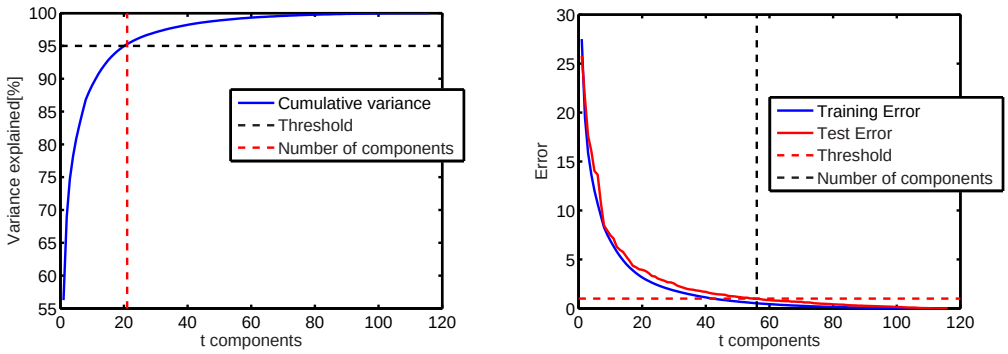


Figure 5.3: Choosing the number of modes. Two ways of choosing the optimal number of eigenvectors to be retained is depicted. In the left figure, the choice is made by choosing the lowest number of vectors explaining 95% of the total variance. The blue curve is the accumulated sum of the variance explained by each vector. In this case, the level is reached by using the 21 first eigenvectors. In the right figure, the choice is made by a requirement on the quality of the fit. It is done in a leave-one-out fashion. One shape is retained as a test shape, while the model is built on the rest of the shapes. Equations (5.4) and (5.5) are then used to calculate the best approximation to the test shape. The mean Euclidean distance between the test shape and the approximation is the recorded. This is repeated, retaining each shape as a test shape. The level for which the total error is below a threshold is the number of eigenvectors to be used.

It can be shown[19] that the non-zero eigenvalues of the matrix are the same as the eigenvalues of the covariance matrix (5.3),

$$\mathbf{\Lambda}_l = \mathbf{\Lambda}_s \tag{5.8}$$

and the eigenvectors $\mathbf{\Phi}_s$ corresponds as,

$$\mathbf{\Phi}_l = \mathbf{S}\mathbf{\Phi}_s. \tag{5.9}$$

If, as often is the case, the number of training samples N is smaller than the number of landmarks n, a substantial reduction in the amount of memory and time required to apply PCA is gained. This trick is absolutely crucial when calculating PCA on the texture data as will be seen later.

5.3 Creating Synthetic Shapes

With the help from PCA we have obtained a model of the object, given by the training shapes. With this model it is possible to create new instances of the object similar to the training shapes.

A synthetic shape $\mathbf{s}$ is created as deformation of the mean shape $\mathbf{s}_0$ by a linear combination of the shape eigenvectors $\mathbf{\Phi}_s$,

$$\mathbf{s} = \mathbf{s}_0 + \mathbf{\Phi}_s \mathbf{b}_s, \tag{5.10}$$

where $\mathbf{b}_s$ is the set of shape parameters. However, in order for the new instance to be a 'legal' representation of the object, we must chose the parameters $\mathbf{b}_s$ so that the instance is similar to those of the training set. If we assume for a moment, that the parameters describing the training shapes are independent and gaussian distributed, then a way to generate a new legal instance would be to constrain the value b_i to $\pm 3\lambda_i$.

Figure 5.4 shows three rows of shapes. The middle row is the mean shape. The left and right rows are synthesized shapes generated by deformation of the mean shape by two standard deviations given by $\pm 2\sqrt{\lambda_i}$.

However, using a gaussian distribution as an approximation of the shape distribution might be an over-simplification. It is assumed, that the shapes generated by parameters within the limits on $\mathbf{b}_s$, is plausible shapes. This is not necessary the case. For instance if an object can assume two different shapes, but not any in between, then the distribution has two separate peaks[21]. In such cases non-linear models of the distribution might be the answer. Cootes et al.[21] suggests using a mixture of gaussians to approximate the distribution. Nevertheless, using gaussian mixtures is outside the scope of this thesis, and approximations using a single gaussian is used.

5.4 Summary

In this chapter, a mathematical framework for statistical models of shapes, has been presented. The model is based on applying PCA to the training shapes. Thus compact model describing the variability of the training shapes is obtained.

The shape model is only one part of the complete active appearance model, and in the next chapter the theory will be extended to include a model of the object texture.

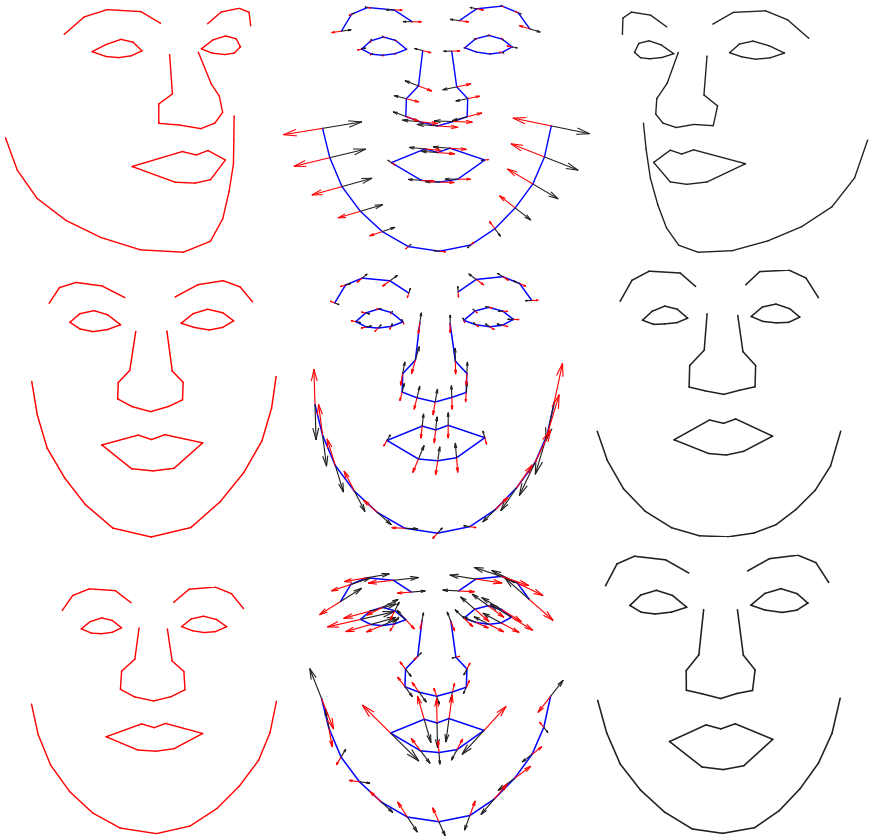


Figure 5.4: Mean shape deformation using first, second and third principal mode. The middle shape is the mean shape, the left column is minus two standard deviations corresponding to $b_{s_i} = -2\lambda_i$, the right is plus two standard deviations given by $b_{s_i} = 2\lambda_i$. The arrows overlain the mean shape indicates the direction and magnitude of the deformation corresponding to the parameter values. The color of the arrows correspond to the instances shown in the first and third column. Especially clear is the effect if the first eigenvector. It describes the left-right rotation of the head.

Chapter 6

Modeling Texture

This chapter describes the statistical model of texture. Together with the shape model, this formulates the face appearance model. The texture model tries to capture the variability of the human face in terms of its color, facial hair etc.

6.1 Building the Model

The texture model is built from a set of annotated images of faces. An annotated face is depicted in figure 5.1. The mesh spanned by the annotated landmarks is triangulated using Delaunay triangulation as seen in figure 6.1. Contrary to the normal computer vision definition of texture as a surface property of an object, it is defined here as the intensities of the pixels inside the mesh spanned by the landmarks[79].

The texture data of each training image is collected as the pixel intensities of the pixels inside the mesh and stored as vectors,

$$\mathbf{g} = [g1, g2, \ldots, g_m]^T . \tag{6.1}$$

Thus if $\mathbf{I}_{train}$ denotes a training image, and $\mathbf{x}$ denotes the coordinates of the set of pixels inside the mesh defined by the landmarks, $\mathbf{g}$ is formed by the following equation

$$\mathbf{g} = \mathbf{I}_{train}(\mathbf{x}). \tag{6.2}$$

The texture model describes the changes in texture across the training set. To ensure, from image to image, that the pixel statistics stems from the same place in the face, the training data must have the same shape. This is done by warping all training images back into the mean shape $\mathbf{s}_0$, using affine warps.

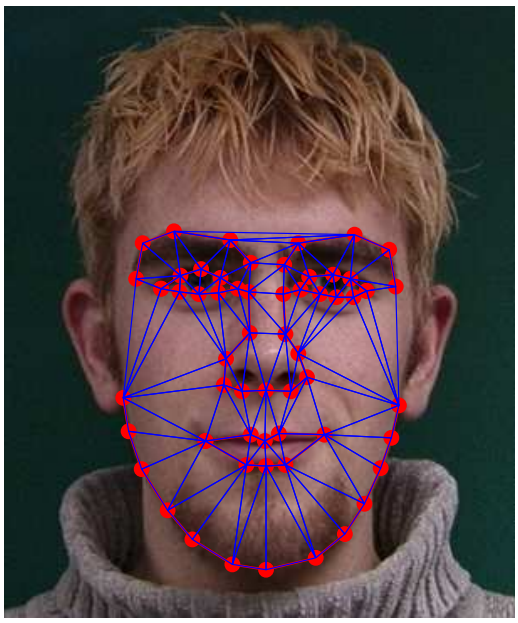


Figure 6.1: An annotated face overlain the Delaunay triangulation of the mesh formed by the landmarks.

6.2 Image Warping

Transforming the training images into a common coordinate frame involves image warping. Basically, image warping is transforming an image with one spatial configuration into another. An image can be warped using a number of different transformations, but, as for the shapes, only similarity transformations are considered. Since an AAM can model a deformable object, a single similarity warp is not enough to describe the often non-linear deformation of the object. To overcome this, a collection of similarity warps is used, in the form of a piecewise affine warp.

Warping is done by considering the shape as mesh of triangles, and then using piecewise affine warping to warp each of the triangles. The triangulation is done using *Delaunay* triangulation. The Delaunay triangulation connects an irregular set of points by a mesh of triangles. All triangles satisfy the Delaunay property which requires that no triangle has any vertices inside its circumcircle[72]. Figure 6.2(left) depicts the Delaunay triangulation of the mean shape. This triangulation is used on all other shapes in the training set. The right side of figure 6.2 shows the corresponding triangulation of one of the training shapes. Thus each triangle in the triangulated mean shape has a corresponding triangle in every training shape. Such a pair of triangles define an unique affine transformation. The collection of warps of

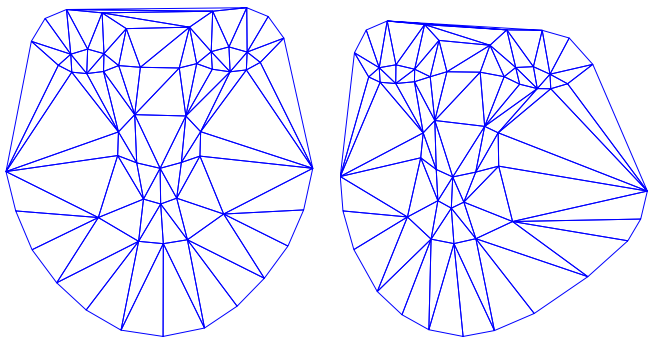


Figure 6.2: Left: The mean shape triangulated using the Delaunay algorithm. Right: A training shape with the triangulation applied.

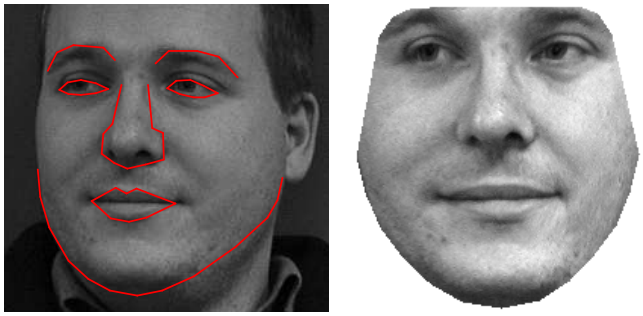


Figure 6.3: Left: One of the training samples with shape overlain. Right: The training sample warped into the mean shape reference frame.

all triangles in a shape denotes a piecewise affine warp from the mean shape to the training shape.

Warping the texture from an annotated training example into the reference frame, is done as follows; for each pixel $\mathbf{x}$ inside the annotated mesh, 1) find the triangle in which the pixel lies, 2) apply the warp given for this triangle, and finally 3) sample the training image at the resulting location. Figure 6.3 shows the image corresponding to the triangulation shown in figure 6.2 and the face warped into the mean shape reference frame. See appendix A.1 for a more thorough explanation of piecewise affine warping.

6.3 Modeling Texture Variation

As for the shape variability, the texture variability is modeled using PCA. The texture vectors (6.1) are stored as columns in a texture matrix $\mathbf{G}$. PCA

Figure 6.4: Three eigenvectors corresponding to the three largest eigenvalues of the texture covariance matrix. The first eigenvector is left.

Figure 6.5: Two synthesized textures, left and right with the mean texture in the middle.

is applied using the low memory covariance matrix as seen in (5.7),

$$\mathbf{\Sigma}_g = \frac{1}{N-1}\mathbf{G}^T\mathbf{G}. \tag{6.3}$$

As with the shapes only a fraction of the eigenvectors is retained. The eigenvectors of the covariance matrix are also known as *eigenfaces*[23], see figure 6.4 which show the eigenvector corresponding to the three largest eigenvalues. A new texture is synthesized, as with the shapes, by deforming the mean texture $\mathbf{g}_0$ with a linear combination of the texture eigenvectors.

$$\mathbf{g} = \mathbf{g}_0 + \mathbf{\Phi}_g\mathbf{b}_g, \tag{6.4}$$

where $\mathbf{b}_g$ is a vector of *texture parameters*. Figure 6.5 shows three textures. The middle texture is the mean texture. The left and right textures are made by deformation of the mean texture by $\pm 2\sqrt{\lambda_1}$.

6.4 Summary

In this chapter a statistical model of the texture of an object has been presented. As for the shape model, the model is based on applying PCA to texture data.

Together with the shape model, the texture model creates a statistical model of the human face. This is the topic of the upcoming chapter.

Chapter 7

The Independent Model

This chapter presents the unification of the statistical model of shape and the statistical model of appearance described in the chapter above.

The 'usual' way to unify the two models, is to use the term literally. In the original formulation by Cootes et al.[22] the models are combined using a third PCA. Thus, a model instance consist of both shape and texture created from one set of parameters. The advantage of the combined model formulation is that it is more compact, requiring less parameters to represent a given object, that with the independent formulation. However, restrictions are made on the choice of fitting algorithm.

Recently, Matthews and Baker[57] proposed to unify the models, by not unifying them so to speak. A model instance is made by creating a shape instance and a texture instance independently, using two separate sets of parameters. The unification is done by warping the instantiated texture into the created shape instance. Quite fittingly, they have named the model *The Independent Model*. With the independent formulation, the choice of fitting algorithm is free.

7.1 Defining the Independent Model

The independent model, models shape and texture independently as,

$$\mathbf{s} = \mathbf{s}_0 + \mathbf{\Phi}_s \mathbf{b}_s, \tag{7.1}$$

and

$$\mathbf{g} = \mathbf{g}_0 + \mathbf{\Phi}_g \mathbf{b}_g, \tag{7.2}$$

respectively. An instance of the AAM is thus created by first creating an instance of the shape $\mathbf{s}$ by setting the shape parameters $\mathbf{b}_s$. Thus $\mathbf{b}_s$ defines the relationship between the shapes $\mathbf{s}$ and $\mathbf{s}_0$ which defines a piecewise affine

Figure 7.1: Two synthesized faces. left and right with the mean texture in the middle.

warp $\mathbf{W}(\mathbf{x}, \mathbf{b}_s)$ of the set of pixels with coordinates $\mathbf{x}$ inside the mesh spanned by the mean shape $\mathbf{s}_0$. Thus the coordinates $\mathbf{x}'$ of the set of pixels inside the mesh spanned by $\mathbf{s}$ is given by,

$$\mathbf{x}' = \mathbf{W}(\mathbf{x}, \mathbf{b}_s). \tag{7.3}$$

Secondly, an instance of the texture model is created by setting the texture parameters $\mathbf{b}_g$. This results in a vector of intensities $\mathbf{g}'$ which can be formed into an image by

$$T_{s_0}(\mathbf{x}) = \mathbf{g}. \tag{7.4}$$

This results in an image T which is defined by the following equation,

$$T(\mathbf{x}') = \mathbf{T}_{s_0}(\mathbf{x}). \tag{7.5}$$

7.2 Summary

This chapter contains a description of the Independent Model, introduced by Matthews and Baker. With this, the statistical model of shape and texture have been concluded. To make the model really useful, and method for enabling it to do actual image segmentation by moving around an image, is needed. This is the topic of the next chapters.

Chapter 8

The Inverse Compositional Algorithm

The previous two chapters have described a statistical model of faces. In order to track moving faces, a method for deforming a model instance according to the image evidence, must be formalized.

In previous work on the AAM's[22], it is assumed that there exists a constant linear relationship between the error and the parameter updates. This, however can lead to false representations of the shape[57].

In [57] Matthews and Baker introduces an analytical method for finding the optimal set of parameters.

8.1 Introduction

Suppose an image I depicts an object, e.g. a face, of which we have built a statistical model as the one described in the previous chapters. The objective is then to find the optimal set of parameters, $\mathbf{b}_s$ and $\mathbf{b}_g$, such that the model instance $T(\mathbf{W}(\mathbf{x}, \mathbf{b}_s))$ is as similar as possible, to the object in the image. An obvious way to measure the success of the fit is to calculate the error between the image and the model instance. An efficient way to calculate this error is to use the coordinate frame defined by the mean shape $\mathbf{s}_0$. Thus, a pixel with coordinate $\mathbf{x}$ in $\mathbf{s}_0$ has a corresponding pixel in the image $\mathbf{I}$ with coordinate $\mathbf{W}(\mathbf{x}, \mathbf{b}_s)$. The error of the fit can then be calculated as the difference in pixel values of the model instance and the image,

$$\mathbf{f}(\mathbf{b}_s, \mathbf{b}_g) = (\mathbf{g}_0 + \Phi_g \mathbf{b}_g) - \mathbf{I}(\mathbf{W}(\mathbf{x}, \mathbf{b}_s)). \tag{8.1}$$

This is a function in the texture parameters $\mathbf{b}_g$ and the shape parameters $\mathbf{b}_s$. A cost function F can be defined as,

$$F(\mathbf{b}_s, \mathbf{b}_g) = \|\mathbf{g}_0 + \Phi_g \mathbf{b}_g - \mathbf{I}(\mathbf{W}(\mathbf{x}, \mathbf{b}_s))\|^2 . \tag{8.2}$$

The optimal solution to (8.2) can be found as,

$$(\mathbf{b}_s^*, \mathbf{b}_g^*) = \arg\min_{\mathbf{b}_s, \mathbf{b}_g} F. \tag{8.3}$$

Solving this, is in general a non-linear least squares problem, but luckily there exists well-proven algorithms[46] for doing so.

The next sections describes a new very fast method, introduced by Baker and Matthews[10], for fitting a deformable template to an image. To see the difference, a well proven method of template alignment is first described. Then the new algorithm is introduced. Both algorithms utilizes the Gauss-Newton non-linear optimization method.

8.2 The Gauss-Newton Algorithm

A method for solving non-linear least squares problems is the Gauss-Newton[46] method. It is used to find a (local) minimizer $\mathbf{p}^*$ of a cost-function,

$$F(\mathbf{p}) = \frac{1}{2}\mathbf{f}^\top \mathbf{f} \tag{8.4}$$

The algorithm is based on using a linear model of a function $\mathbf{f}(\mathbf{p})$ in the neighborhood of $\mathbf{p}$,

$$\mathbf{f}(\mathbf{p} + \Delta\mathbf{p}) \simeq \ell(\Delta\mathbf{p}) \equiv f(\mathbf{p}) + \mathbf{J}(\mathbf{p})\Delta\mathbf{p}, \tag{8.5}$$

where $\mathbf{J}$ is the Jacobian of $\mathbf{f}$. It assumes a known current estimate of $\mathbf{p}$ and then iteratively solves for an additive update $\Delta\mathbf{p}$ of the parameters..

Inserting (8.5) into (8.4),

$$F(\mathbf{x}; \mathbf{p} + \Delta\mathbf{p}) \simeq L(\Delta\mathbf{p}) \equiv F(\mathbf{x}; \mathbf{p}) + \Delta\mathbf{p}^\top \mathbf{J}^\top \mathbf{f} + \frac{1}{2}\Delta\mathbf{p}^\top \mathbf{J}^\top \mathbf{J}\Delta\mathbf{p}, \tag{8.6}$$

where $\mathbf{f} = \mathbf{f}(\mathbf{p})$ and $\mathbf{J} = \mathbf{J}(\mathbf{p})$. Finding the increment $\Delta\mathbf{p}$ is done by minimizing $L(\Delta\mathbf{p})$. Sufficient conditions for a local minimizer of $L(\Delta\mathbf{p})$ is that the gradient of L

$$\mathbf{L}'(\Delta\mathbf{p}) = \mathbf{J}^\top \mathbf{f} + \mathbf{J}^\top \mathbf{J}\boldsymbol{\Delta}\mathbf{p}, \tag{8.7}$$

is equal to zero and the Hessian,

$$\mathbf{L}'' = \mathbf{J}^\top \mathbf{J}, \tag{8.8}$$

is positive definite[46]. Such a minimizer $\Delta\mathbf{p}^*$ can be found by,

$$\begin{aligned}\left(\mathbf{J}^\top\mathbf{J}\right)\Delta\mathbf{p}^* &= -\mathbf{J}^\top\mathbf{f} \\ \Delta\mathbf{p}^* &= -\left(\mathbf{J}^\top\mathbf{J}\right)^{-1}\mathbf{J}^\top\mathbf{f}.\end{aligned} \tag{8.9}$$

The parameters are then updated,

$$\mathbf{p} = \mathbf{p} + \Delta\mathbf{p}^*. \tag{8.10}$$

8.3 The Lucas-Kanade Algorithm

Assume for a moment that the model instance is rigid template with constant texture. Then the fit boils down to a simple image alignment. One of the most important and widely used algorithms is the *Lucas-Kanade*-algorithm[54]. The best alignment is found by minimizing the difference between the pixel values of the image and of the template,

$$\mathbf{f}(\mathbf{p}) = I(\mathbf{W}(\mathbf{x};\mathbf{p})) - T(\mathbf{x}), \tag{8.11}$$

for all pixels $\mathbf{x}$ in the template T. $I(\mathbf{W}(\mathbf{x};\mathbf{p}))$ denotes that the image I has been warped into the templates coordinate system, see appendix A.1. The locally best minimizer $\mathbf{p}^*$ of the error function can be found be solving the following least squares problem,

$$F(\mathbf{p}) = \frac{1}{2}\sum_{\mathbf{x}}\left[I(\mathbf{W}(\mathbf{x};\mathbf{p})) - T(\mathbf{x})\right]^2, \tag{8.12}$$

where the sum is performed on all pixels in T.

The Lucas-Kanade algorithm utilizes the Gauss-Newton method for minimization. The following expression must be minimized,

$$F(\mathbf{p}) = \frac{1}{2}\sum_{\mathbf{x}}\left[I(\mathbf{W}(\mathbf{x};\mathbf{p}+\Delta\mathbf{p})) - T(\mathbf{x})\right]^2 \tag{8.13}$$

For the Lucas-Kanade algorithm the linear model from (8.5) becomes,

$$\ell(\Delta\mathbf{p}) = I(\mathbf{W}(\mathbf{x};\mathbf{p})) - T(\mathbf{x}) + \nabla I(\mathbf{W}(\mathbf{x};\mathbf{p}))\frac{\partial\mathbf{W}(\mathbf{x};\mathbf{p})}{\partial\mathbf{p}}\Delta\mathbf{p}, \tag{8.14}$$

where the Jacobian of $\mathbf{f}$ is,

$$\mathbf{J}(\mathbf{p}) = \nabla I(\mathbf{W}(\mathbf{x};\mathbf{p}))\frac{\partial\mathbf{W}(\mathbf{x};\mathbf{p})}{\partial\mathbf{p}}. \tag{8.15}$$

Here $\nabla I = \{\frac{\partial I}{\partial x}, \frac{\partial I}{\partial y}\}$ is the gradient of the image at coordinate $\mathbf{W}(\mathbf{x};\mathbf{p})$. It is computed in the coordinate frame of I and then warped into the coordinate frame of T using $\mathbf{W}(\mathbf{x};\mathbf{p})$. $\frac{\partial \mathbf{W}}{\partial \mathbf{p}}$ is the Jacobian of the warp $\mathbf{W}(\mathbf{x};\mathbf{p}) = (W_x(\mathbf{x};\mathbf{p}), W_y(\mathbf{x};\mathbf{p}))^\top$,

$$\frac{\partial \mathbf{W}}{\partial \mathbf{p}} = \begin{pmatrix} \frac{\partial W_x}{\partial p_1} & \frac{\partial W_x}{\partial p_2} & \cdots & \frac{\partial W_x}{\partial p_n} \\ \frac{\partial W_y}{\partial p_1} & \frac{\partial W_y}{\partial p_2} & \cdots & \frac{\partial W_y}{\partial p_n} \end{pmatrix} \tag{8.16}$$

Using (8.9) the minimizer for the Lucas-Kanade alignment algorithm becomes,

$$\Delta \mathbf{p}^* = -\mathbf{H}^{-1} \sum_{\mathbf{x}} \left[\nabla I(\mathbf{W}(\mathbf{x};\mathbf{p})) \frac{\partial \mathbf{W}(\mathbf{x};\mathbf{p})}{\partial \mathbf{p}} \right]^\top \left[I(\mathbf{W}(\mathbf{x};\mathbf{p})) - T(\mathbf{x}) \right], \tag{8.17}$$

where $\mathbf{H}$ is the Gauss-Newton approximation to the Hessian,

$$\mathbf{H} = \sum_{\mathbf{x}} \left[\nabla I(\mathbf{W}(\mathbf{x};\mathbf{p})) \frac{\partial \mathbf{W}(\mathbf{x};\mathbf{p})}{\partial \mathbf{p}} \right]^\top \left[\nabla I(\mathbf{W}(\mathbf{x};\mathbf{p})) \frac{\partial \mathbf{W}(\mathbf{x};\mathbf{p})}{\partial \mathbf{p}} \right]. \tag{8.18}$$

One iteration of the Lucas-Kanade algorithm proceeds as follows[11],

1. Warp I with $\mathbf{W}(\mathbf{x};\mathbf{p})$ to compute $I(\mathbf{W}(\mathbf{x};\mathbf{p}))$
2. Calculate $\mathbf{f}(\mathbf{p})$ using (8.11)
3. Calculate ∇I and warp with $\mathbf{W}(\mathbf{x};\mathbf{p})$
4. Calculate Jacobian $\frac{\partial \mathbf{W}}{\partial \mathbf{p}}$ of the warp at $\mathbf{p}$
5. Compute the Jacobian $\nabla I \frac{\partial \mathbf{w}}{\partial \mathbf{p}}$
6. Compute the Hessian matrix using (8.18)
7. Compute $\Delta \mathbf{p}^*$ using (8.17)
8. Update the parameters $\mathbf{p} = \mathbf{p} + \Delta \mathbf{p}^*$

Because the gradient ∇I is calculated at $\mathbf{W}(\mathbf{x};\mathbf{p})$ and the Jacobian of the warp $\frac{\partial \mathbf{W}}{\partial \mathbf{p}}$ at $\mathbf{p}$, they both depend on $\mathbf{p}$. Thus the Jacobian from (8.15), and thus the Hessian H aswell, has to be recalculated at every iteration of the algorithm. This makes the Lucas-Kanade a very computationally demanding algorithm, and not plausible in a real time setting.

8.4 The Inverse Compositional Algorithm

Recently Baker and Matthews[11] have introduced a new much faster fitting algorithm, in which the Jacobian and the Hessian can be precomputed. As the name implies the algorithm consists of two innovations. The compositional part refers to the updating of the parameters and the inverse part indicates that the image and the template switches roles. The function is changed to,

$$\mathbf{f}(\Delta\mathbf{p}) = T(\mathbf{W}(\mathbf{x};\Delta\mathbf{p})) - I(\mathbf{W}(\mathbf{x};\mathbf{p})) \tag{8.19}$$

While the Lucas-Kanade algorithm solves for an additive update $\Delta\mathbf{p}$ of the parameters $\mathbf{p} = \mathbf{p} + \Delta\mathbf{p}$, a compositional approach solves for an incremental warp $\mathbf{W}(\mathbf{x};\Delta\mathbf{p})$ which is composed with the current warp. For simple warps composing means a multiplication of two matrices, however for more complex warps, such as the piecewise affine warp, the meaning becomes more complex.

The goal in a compositional algorithm is to solve for $\Delta\mathbf{p}$ in,

$$F(\mathbf{p}) = \frac{1}{2}\sum_{\mathbf{x}} \left[I(\mathbf{W}(\mathbf{W}(\mathbf{x};\Delta\mathbf{p});\mathbf{p})) - T(\mathbf{x})\right]^2, \tag{8.20}$$

which is the compositional version of (8.13). The update to the warp is,

$$\mathbf{W}(\mathbf{x};\mathbf{p}) = \mathbf{W}(\mathbf{x};\mathbf{p}) \circ \mathbf{W}(\mathbf{x};\Delta\mathbf{p}), \tag{8.21}$$

where $\circ$ denotes that the two warps are composed.

The inverse part of the name denotes that the template T and the image I are changing roles, and (8.20) becomes

$$F(\mathbf{p}) = \frac{1}{2}\sum_{\mathbf{x}} \left[T(\mathbf{W}(\mathbf{x};\Delta\mathbf{p})) - I(\mathbf{W}(\mathbf{x};\mathbf{p}))\right]^2. \tag{8.22}$$

Thus, instead of composing the update onto the warping of the image, the update is used to warp the template.

The inverse compositional algorithm also utilizes the Gauss-Newton method to solve for $\Delta\mathbf{p}^*$. From (8.22) it can be seen that the incremental warp $\mathbf{W}(\mathbf{x};\Delta\mathbf{p})$ applies only to the template T. Thus the linear model from (8.5) is built around $\mathbf{0}$ becoming $\ell(\Delta\mathbf{p}) = f(\mathbf{0}) + \mathbf{J}(\mathbf{0})\Delta\mathbf{p}$, which gives

$$\ell(\Delta\mathbf{p}) = T(\mathbf{W}(\mathbf{x};\mathbf{0})) - I(\mathbf{W}(\mathbf{x};\mathbf{p})) + \nabla T(\mathbf{x})\frac{\partial\mathbf{W}(\mathbf{x};\mathbf{0})}{\partial\mathbf{p}}\Delta\mathbf{p}, \tag{8.23}$$

and the Jacobian is,

$$\mathbf{J}(\mathbf{x};\mathbf{0}) = \nabla T(\mathbf{x})\frac{\partial\mathbf{W}(\mathbf{x};\mathbf{0})}{\partial\mathbf{p}}. \tag{8.24}$$

Using (8.9), the local minimizer of (8.22) becomes

$$\Delta \mathbf{p}^* = -H^{-1} \sum_{\mathbf{x}} \left[\nabla T(\mathbf{x}) \frac{\partial \mathbf{W}(\mathbf{x};\mathbf{0})}{\partial \mathbf{p}} \right]^\top \left[T(\mathbf{W}(\mathbf{x};\mathbf{0})) - I(\mathbf{W}(\mathbf{x};\mathbf{p})) \right], \quad (8.25)$$

where $\mathbf{H}$ is the Gauss-Newton approximation to the Hessian,

$$\mathbf{H} = \sum_{\mathbf{x}} \left[\nabla T(\mathbf{x}) \frac{\partial \mathbf{W}(\mathbf{x};\mathbf{0})}{\partial \mathbf{p}} \right]^\top \left[\nabla T(\mathbf{x}) \frac{\partial \mathbf{W}(\mathbf{x};\mathbf{0})}{\partial \mathbf{p}} \right]. \quad (8.26)$$

As can be seen from (8.23) both the image gradient $\nabla T(\mathbf{x})$ and the warp Jacobian $\frac{\partial \mathbf{W}(\mathbf{x};\mathbf{0})}{\partial \mathbf{p}}$ is independent of $\mathbf{p}$. Thus, the Jacobian of $\mathbf{f}$ is independent of $\mathbf{p}$ and constant from iteration to iteration. This means the Jacobian, and the Hessian, can be precomputed making the algorithm very effective.

In [11] Baker and Matthews proves that the update $\Delta \mathbf{p}$ calculated using the inverse compositional algorithm is equivalent, to a first order approximation, to the update calculated using the Lucas-Kanade algorithm.

8.5 Including Appearance Variation

The Inverse Compositional algorithm introduced in the last section, assumes that the template has constant texture. So in order to make the algorithm work with AAM's, something has to be done. Now there is two parameters which controls the shape and appearance of the template, and thus the warp is now denoted $\mathbf{W}(\mathbf{x};\mathbf{b}_s)$ to indicate the connection with the AAM. The appearance of the template is governed by the parameter $\mathbf{b}_g$.

A template with appearance variation can be formulated as,

$$g(\mathbf{x}) = g_0(\mathbf{x}) + \sum_{i=1}^{m} b_{gi} g_i(\mathbf{x})), \quad (8.27)$$

where m is the number of texture components.

Inserting the new template (8.27) into (8.12) and rewriting it becomes,

$$F(\mathbf{p}) = \frac{1}{2} \left\| I(\mathbf{W}(\mathbf{x};\mathbf{b}_s)) - \left(\mathbf{g}_0(\mathbf{x}) + \sum_{i=1}^{m} b_{gi} \mathbf{g}_i(\mathbf{x}) \right) \right\|^2. \quad (8.28)$$

This expression must be minimized with respect to both the shape parameters $\mathbf{b}_s$ and the texture parameters $\mathbf{b}_g$ simultaneously. Denote the linear subspace spanned by a collection of vectors $\mathbf{g}_i$ by $\text{span}(\mathbf{g}_i)$ and by $\text{span}(\mathbf{g}_i)^\perp$

its orthogonal complement. The Euclidean norm can then be split into components[11],

$$\left\| I(\mathbf{W}(\mathbf{x};\mathbf{b}_s)) - \mathbf{g}_0(\mathbf{x}) + \sum_{i=1}^{m} b_{gi}\mathbf{g}_i(\mathbf{x}) \right\|^2_{\text{span}(\mathbf{g}_i)} + \left\| I(\mathbf{W}(\mathbf{x};\mathbf{b}_s)) - \mathbf{g}_0(\mathbf{x}) + \sum_{i=1}^{m} b_{gi}\mathbf{g}_i(\mathbf{x}) \right\|^2_{\text{span}(\mathbf{g}_i)^\perp} \tag{8.29}$$

where the first expression is the norm of the vector projected into the subspace spanned by the $\mathbf{g}_i$ vector. The second expression is the norm the vectors projected into the orthogonal complement, and thus it can be simplified. Since the norm is calculated in a subspace orthogonal to span($\mathbf{g}_i$) the term $\sum_{i=1}^{m} b_{gi}\mathbf{g}_i(\mathbf{x})$ has no influence. Thus (8.29) can be simplified to

$$\left\| I(\mathbf{W}(\mathbf{x};\mathbf{b}_s)) - \mathbf{g}_0(\mathbf{x}) + \sum_{i=1}^{m} b_{gi}\mathbf{g}_i(\mathbf{x}) \right\|^2_{\text{span}(\mathbf{g}_i)} + \left\| I(\mathbf{W}(\mathbf{x};\mathbf{b}_s)) - g_0(\mathbf{x}) \right\|^2_{\text{span}(\mathbf{g}_i)^\perp} \tag{8.30}$$

As seen the second term does not depend on $\mathbf{b}_g$ and for any $\mathbf{b}_s$ the value of the first term is always zero[11]. Thus the optimal set of parameters can be found in two steps: 1) Minimize the second term,

$$F_{po}(\mathbf{b}_s) = \left\| I(\mathbf{W}(\mathbf{x};\mathbf{b}_s)) - g_0(\mathbf{x}) \right\|^2_{\text{span}(\mathbf{g}_i)^\perp} \tag{8.31}$$

using the inverse compositional algorithm from section 8.4, to obtain the optimal $\mathbf{b}_s$. 2) Minimize the first term with respect to $\mathbf{b}_g$ which can be found as the closed form solution[9] to,

$$b_{gi} = \sum_{\mathbf{x}} \mathbf{g}_i(\mathbf{x}) \left[I(\mathbf{W}(\mathbf{x};\mathbf{b}_s)) - \mathbf{g}_0(\mathbf{x}) \right], \tag{8.32}$$

using the obtained $\mathbf{b}_s$.

Minimizing (8.31) has to be done in the linear subspace span$(\mathbf{g}_i)^\perp$. This can be achieved by altering the Jacobian in the inverse algorithm (8.24),

$$\mathbf{J}_\perp(\mathbf{x};\mathbf{0}) = \nabla\mathbf{g}_0(\mathbf{x})\frac{\partial\mathbf{W}(\mathbf{x};\mathbf{0})}{\partial\mathbf{b}_s} - \sum_{i=1}^{m}\left[\sum_{\mathbf{x}} \mathbf{g}_i(\mathbf{x}) \cdot \nabla\mathbf{g}_0(\mathbf{x})\frac{\partial\mathbf{W}(\mathbf{x};\mathbf{0})}{\partial\mathbf{b}_s}\right]\mathbf{g}_i(\mathbf{x}), \tag{8.33}$$

corresponding to subtracting the components in the direction of the vector $\mathbf{g}_i$ for $i = 1, \ldots, m$ one at a time, see [9] for details. The Hessian corresponding to (8.26) is then computed with the modified Jacobian

$$\mathbf{H}_\perp = \sum_x \left[\mathbf{J}_\perp(\mathbf{x}; \mathbf{0})^\top \mathbf{J}_\perp(\mathbf{x}; \mathbf{0}) \right]. \tag{8.34}$$

Finally the parameter update is obtained by,

$$\Delta \mathbf{b}_s^* = -H_\perp^{-1} \sum_{\mathbf{x}} \left[\mathbf{J}_\perp(\mathbf{x}; \mathbf{0}) \right]^\top \left[\mathbf{g}_0 - I(\mathbf{W}(\mathbf{x}; \mathbf{b}_s)) \right], \tag{8.35}$$

The inverse compositional algorithm the proceeds as follows[11]. First precompute,

- Calculate Jacobian $\frac{\partial \mathbf{W}}{\partial \mathbf{p}}$ of the warp $\mathbf{W}(\mathbf{x}; \mathbf{0})$.
- Calculate the gradients of the template ∇T_0.
- Calculate the warp Jacobian using (8.33)
- Compute the modified Hessian matrix using the modified Jacobian by (8.34).

then iterate until convergence

1. Warp I with $\mathbf{W}(\mathbf{x}; \mathbf{b}_s)$ to compute $I(\mathbf{W}(\mathbf{x}; \mathbf{b}_s))$
2. Calculate $\mathbf{f}(\mathbf{b}_s)$ (8.19)
3. Compute $\Delta \mathbf{b}_s^*$ using (8.35) with the modified Jacobian and Hessian.
4. Update the warp $\mathbf{W}(\mathbf{x}; \mathbf{b}_s) = \mathbf{W}(\mathbf{x}; \mathbf{b}_s) \circ \mathbf{W}(\mathbf{x}; \Delta \mathbf{b}_s)$

Comparing with the Lucas-Kanade algorithm from section 8.3 is seen that the computationally demanding operations of computing the Jacobian and the Hessian have been moved to a precomputation step. The algorithm is very fast and suitable for a realtime application.

8.6 Summary

In this chapter a very fast image general alignment algorithm has been described. It uses a compositional approach for updating the parameters, and avoid recalculation of the Jacobian and Hessian by reversing the roles of the template and the image.

Now the inverse compositional algorithm can be used to align deformable templates with appearance variation, such as an Active Appearance Model instance.

Chapter 9

Fitting the Active Appearance Model

In order to utilize the inverse compositional algorithm to optimize an AAM search, four things must be defined: 1) The warp $\mathbf{W}(\mathbf{x};\mathbf{b}_s)$ must be specified. 2) The warp Jacobian $\frac{\partial \mathbf{W}(\mathbf{x};\mathbf{0})}{\partial \mathbf{b}_s}$ must be derived and combined with the gradient $\nabla \mathbf{g}_0(\mathbf{x})$. 3) The inverse of the warp, $\mathbf{W}(\mathbf{x};\mathbf{b}_s)^{-1}$, must be defined. 4) Finally the composition of the two warps $\mathbf{W}(\mathbf{x};\mathbf{b}_s) \circ \mathbf{W}(\mathbf{x};\Delta\mathbf{b}_s)$ must be derived.

9.1 Computing the Warp Jacobian

As described in section 5.3 a shape s is defined as a deformation of the mean shape $\mathbf{s}_0$ by shape eigenvectors, as in (5.10) which is rewritten here,

$$\mathbf{s} = \mathbf{s}_0 + \mathbf{\Phi}_s \mathbf{b}_s = \mathbf{s}_0 + \sum_{i=1}^{m} b_{s_i} \varphi_{s_i}, \tag{9.1}$$

where φ_{s_i} is the i'th shape eigenvector from $\mathbf{\Phi}_s$, and b_{s_i} is the i'th shape parameter.

The spatial relationship between $\mathbf{s}$ and $\mathbf{s}_0$ obtained by $\mathbf{b}_s$ defines a piecewise affine warp $\mathbf{W}(\mathbf{x};\mathbf{b}_s)$ through the parameters $\mathbf{b}_s$ as described in 6.2 and A.1. Thus the chain rule can be applied to $\mathbf{W}(\mathbf{x};\mathbf{b}_s)$ yielding[57],

$$\frac{\partial \mathbf{W}}{\partial \mathbf{b}_s} = \sum_{i=1}^{n} \left[\frac{\partial \mathbf{W}}{\partial x_i} \frac{\partial x_i}{\partial \mathbf{b}_s} + \frac{\partial \mathbf{W}}{\partial y_i} \frac{\partial y_i}{\partial \mathbf{b}_s} \right] \tag{9.2}$$

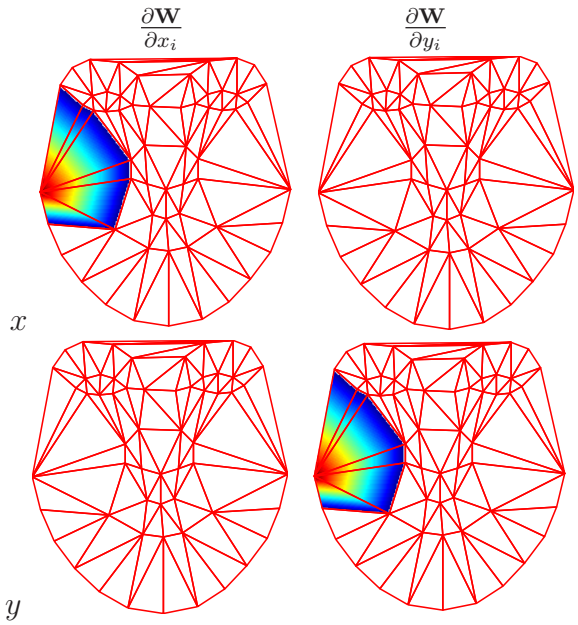


Figure 9.1: The Jacobians $\frac{\partial \mathbf{W}}{\partial x_i}$ and $\frac{\partial \mathbf{W}}{\partial x_i}$ with respect to the vertices of $\mathbf{s}$ for a vertex i. The Jacobians denote the rate of change of the warp with respect to the vertex i. The top row depicts the x component of the Jacobian and the bottom row the y component, colored with warm colors as high values. The Jacobians are only nonzero in the triangles which have i as a vertex. It has the maximum value of one at the vertex and decays away according to equation (9.4).

9.1.1 The Shape Jacobians

$\frac{\partial \mathbf{W}}{\partial x_i}$ and $\frac{\partial \mathbf{W}}{\partial y_i}$ is seen to be the Jacobian of the warp with respect to the vertices of a shape $\mathbf{s}$. Each vertex in the shape is part of one ore more triangles, and thus defines one or more warps. Rewriting (A.2), a warp is defined as

$$\mathbf{W}(\mathbf{x}; \mathbf{b}_s) = \alpha(x_i, y_i)^\top + \beta(x_j, y_j)^\top + \gamma(x_k, y_k)^\top, \tag{9.3}$$

and the Jacobians becomes,

$$\begin{aligned} \frac{\partial \mathbf{W}}{\partial x_i} &= (\alpha, 0)^\top = (1 - \beta - \gamma, 0)^\top \\ \frac{\partial \mathbf{W}}{\partial y_i} &= (0, \alpha)^\top = (0, 1 - \beta - \gamma)^\top, \end{aligned} \tag{9.4}$$

for the triangles which have (x_i, y_i) as a vertex. In the inverse compositional algorithm the Jacobians can be precomputed as $\frac{\partial \mathbf{W}(\mathbf{x};\mathbf{0})}{\partial x_i}$ and $\frac{\partial \mathbf{W}(\mathbf{x};\mathbf{0})}{\partial y_i}$, which are derivatives with respect to the vertices of $\mathbf{s}_0$. The result is shown in figure 9.1, which depicts the x and y component of $\frac{\partial \mathbf{W}(\mathbf{x};\mathbf{0})}{\partial x_i}$ and $\frac{\partial \mathbf{W}(\mathbf{x};\mathbf{0})}{\partial y_i}$ for a single

vertex i reordered as an image and with the shape $\mathbf{s}_0$ superimposed. Notice that the Jacobians is only nonzero in the triangles which have i as a vertex. It has the maximum value of one at the vertex and decays away according to equation (9.4).

9.1.2 The Parameter Jacobians

Expressing (5.10) for a single vertex i yields,

$$(x_i, y_i)^\top = (x_i^0, y_i^0)^\top + \sum_{j=1}^{m} b_j x_i^{s_j}, \tag{9.5}$$

where b_j is the jth parameter in $\mathbf{b}_s$, (x_i^0, y_i^0) is the ith vertex of the mean shape $\mathbf{s}_0$ and $x_i^{s_j}$ is the ith vertex of the jth eigenshape. Then differentiating (9.5) with respect to the parameters gives the second components of the warp Jacobian, $\frac{\partial x_i}{\partial \mathbf{b}_s}$ and $\frac{\partial y_i}{\partial \mathbf{b}_s}$ gives,

$$\frac{\partial x_i}{\partial \mathbf{b}_s} = (x_i^{s_1}, x_i^{s_2}, \ldots, x_i^{s_m}) \, \frac{\partial y_i}{\partial \mathbf{b}_s} = (y_i^{s_1}, y_i^{s_2}, \ldots, y_i^{s_m}) \,,$$

which again are computed for the vertices of the mean shape $\mathbf{s}_0$.

9.1.3 Assembling the Warp Jacobian

Calculating (9.2) by using (9.6) and (9.4) results in the warp Jacobian. This can also be reordered as images, as depicted in figure 9.2. The figure shows the Jacobian corresponding to the three first shape eigenvectors. The first row depicts the effects of the three eigenvectors, where most noticeable is the left-right rotation of the head induced by φ_{s_i}. The second and third row are the x and y components of the warp Jacobian, denoting the rate of change in the warp with respect to the shape eigenvectors. It can be seen that the rate of change corresponds to the movement depicted in the first row. Again most noticeably is the left-right head turn of the first column.

9.1.4 Steepest Descent Images

To calculate the modified Jacobian of the error function from (8.33) it is seen that the warp Jacobian must be multiplied with the gradient of the template $\nabla \mathbf{g}_0$. figure 9.3 depicts the x and y component of the gradient.

Baker and Matthews[11] denotes the Jacobian of (8.33) as *steepest descent images* which follows from the fact that they provide the steepest descent direction, as per equation (8.9). Figure 9.4 depicts two steepest descent images, corresponding to the first parameters of $\mathbf{b}_s$.

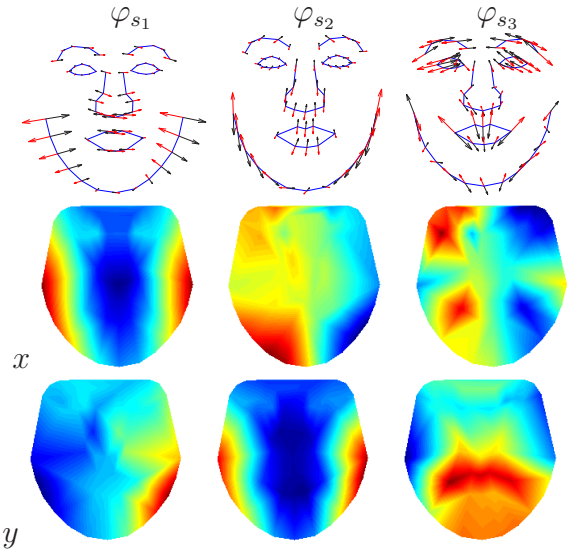


Figure 9.2: Top row: The mean shape $\mathbf{s}_0$ with the shape eigenvectors φ_{s_1}, φ_{s_1}, $\mathbf{s}_3$ and φ_{s_1} overlain. Middle row: The x component of the warp Jacobian $\frac{\partial \mathbf{W}}{\partial \mathbf{b}_s}$. Bottom row: The y component of the warp Jacobian $\frac{\partial \mathbf{W}}{\partial \mathbf{b}_s}$. The Jacobians denote the rate of change in the warp with respect to the parameters $\mathbf{b}_s$. The first parameter corresponds approximately to a left-right rotation of the head, which is also visible in the magnitude of the x component of the Jacobian. b_2 corresponds to an up-down motion,, and b_3 controls the magnitude of the smile. See also figure 5.4.

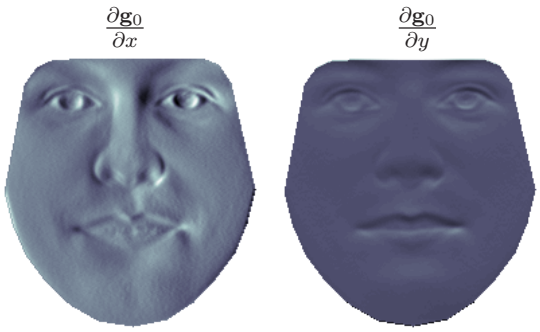


Figure 9.3: Gradient of the mean texture $\mathbf{g}_0$.

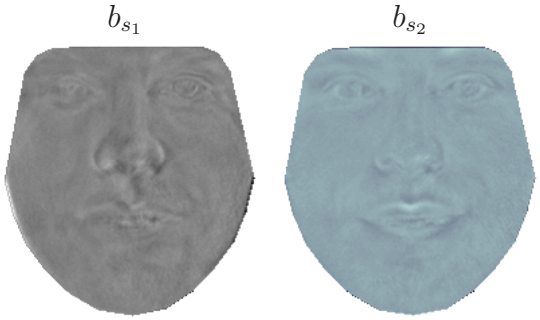


Figure 9.4: Steepest descent images.

9.1.5 The Parameter Update

Now all the parts of equation (8.35) are available and the parameter update $\Delta\mathbf{b}_s^*$ can be calculated. The bracketed part of the equation,

$$[\mathbf{J}_\perp(\mathbf{x};\mathbf{0})]^\top [\mathbf{g}_0 - I(\mathbf{W}(\mathbf{x};\mathbf{b}_s))] \tag{9.6}$$

is a dot product of a steepest descent image and the error between the template and the the area of the image covered by the warped shape. Figure 9.5 depicts and image with the shape mesh superimposed, and it the corresponding error image. The result of the all dot products yield the direction of steepest ascent. Together with the negative inverse Hessian the result is a m dimensional displacement vector $\Delta\mathbf{b}_s^*$in the direction of the steepest descent.

9.2 Warp Inversion

Since this is the *inverse* compositional algorithm the warp $\mathbf{W}(\mathbf{x};\Delta\mathbf{b}_s)$ must inverted to compute $\mathbf{W}(\mathbf{x};\Delta\mathbf{b}_s)^{-1}$. Inverting the warp is not as straightforward as it might seem. The set of piecewise affine warps does not form a group[57] and thus the inverse is not defined. The approach is to use a first order approximation to the inverse. Taylor expansion yields,

$$\begin{aligned}\mathbf{W}(\mathbf{x};\Delta\mathbf{b}_s) &= \mathbf{W}(\mathbf{x};\mathbf{0}) + \frac{\partial\mathbf{W}(\mathbf{x};\mathbf{0})}{\partial\mathbf{b}_s}\Delta\mathbf{b}_s \\ &= \mathbf{x} + \frac{\partial\mathbf{W}(\mathbf{x};\mathbf{0})}{\partial\mathbf{b}_s}\Delta\mathbf{b}_s,\end{aligned} \tag{9.7}$$

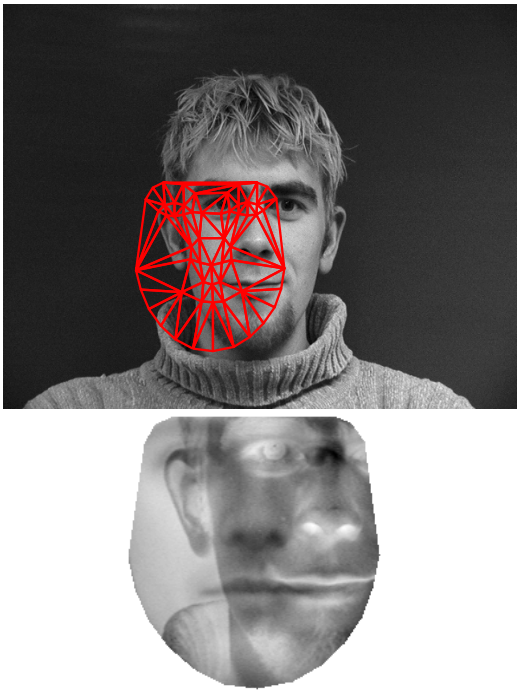


Figure 9.5: Top: An AAM instance overlain the target image. Bottom: The error image, corresponding to the pixels under the instance warped into the mean shape and subtracted from the mean texture.

since $\mathbf{x} = \mathbf{W}(\mathbf{x}; \mathbf{0})$. From this follows

$$\begin{aligned} \mathbf{W}(\mathbf{x}; \Delta\mathbf{b}_s) \circ \mathbf{W}(\mathbf{x}; -\Delta\mathbf{b}_s) = & \quad \mathbf{x} \quad - \frac{\partial \mathbf{W}(\mathbf{x}; \mathbf{0})}{\partial \mathbf{b}_s} \Delta\mathbf{b}_s \\ & + \quad \mathbf{x} + \frac{\partial \mathbf{W}(\mathbf{x}; \mathbf{0})}{\partial \mathbf{b}_s} \Delta\mathbf{b}_s \\ = & \quad \mathbf{x}. \end{aligned} \tag{9.8}$$

Thus to a first order in $\Delta\mathbf{b}_s$ the inverse warp is,

$$\mathbf{W}(\mathbf{x}; \Delta\mathbf{b}_s)^{-1} = \mathbf{W}(\mathbf{x}; -\Delta\mathbf{b}_s) \tag{9.9}$$

9.3 Warp Composing

The remaining step is to compose the warp $\mathbf{W}(\mathbf{x}; -\Delta\mathbf{b}_s)$ with the current warp $\mathbf{W}(\mathbf{x}; \mathbf{b}_s)$. Since the piecewise warp does not form a group a first order approximation must be made here also[57].

Denote the destination of the mean shape $\mathbf{s}_0$ under the warp $\mathbf{W}(\mathbf{x}; \Delta\mathbf{b}_s)^{-1}$ by $\mathbf{s}_0 + \Delta\mathbf{s}_0$. Since $\mathbf{W}(\mathbf{x}; \Delta\mathbf{b}_s)^{-1} = \mathbf{W}(\mathbf{x}; -\Delta\mathbf{b}_s)$ and from (9.1) we get,

$$\Delta\mathbf{s}_0 = -\sum_{i=1}^{m} \Delta b_{s_i} \varphi_{s_i}. \tag{9.10}$$

Computing $\Delta\mathbf{s}$ from $\Delta\mathbf{s}_0$ requires calculating the changes to the current shape $\mathbf{s}$. This can be done by warping $\mathbf{s}_0 + \Delta\mathbf{s}_0$ with the current warp. However since a vertex of $\mathbf{s}_0$ can belong to multiple triangle, it is not clear which warp to choose. Two different triangles produces two different results. Baker and Matthews[57] solves this using a heuristic approach. For each vertex i they warp $(x_i, y_i) + (\Delta x_i, \Delta y_i)$ with every warp corresponding to the triangles connected to the vertex. The final destination is then the mean of all destinations, and $\Delta\mathbf{s}$ can be found.

This enables us to find the parameters b'_{s_i} of $\mathbf{W}(\mathbf{x}; \mathbf{b}_s) \circ \mathbf{W}(\mathbf{x}; -\Delta\mathbf{b}_s)$,

$$b'_{s_i} = \mathbf{s}_i^\top (\mathbf{s} + \Delta\mathbf{s} - \mathbf{s}_0). \tag{9.11}$$

9.4 Including a Global Shape Transform

The AAM described above is capable of fitting the shape locally to an object. What needs to be described is how to move shape around the image globally.

This can be done by augmenting the AAM described above with a similarity transformation as show below,

$$\mathbf{N}(\mathbf{x};\mathbf{q}) = \begin{pmatrix} (1+a) & -b \\ b & (1+a) \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} t_x \\ t_y \end{pmatrix}, \tag{9.12}$$

where the parameters $\mathbf{q} = (a, b, t_x, t_y)$ are $a = k\cos\theta - 1$, $b = k\sin\theta$ and $(t_x, t_y$ are the translations in the x and y direction. When $\mathbf{q} = 0$ then $\mathbf{N}$ is the identity transformation $\mathbf{x} = \mathbf{N}(\mathbf{x};\mathbf{q})$.

The above can be reparameterized for incorporation into the AAM framework. Four vectors are defined as,

$$\varphi_{N_1} = \left(x_1^0, x_2^0, \ldots, x_n^0, y_1^0, y_2^0, \ldots, y_n^0\right)^\top \tag{9.13}$$

$$\varphi_{N_2} = \left(-y_1^0, -y_2^0, \ldots, -y_n^0, x_1^0, x_2^0, \ldots, x_n^0\right)^\top \tag{9.14}$$

$$\varphi_{N_3} = (1, 1, \ldots, 1, 0, 0, \ldots, 0)^\top \tag{9.15}$$

$$\varphi_{N_4} = (0, 0, \ldots, 0, 1, 1 \ldots, 1)^\top. \tag{9.16}$$

The similarity transformation can now be expressed as,

$$\mathbf{N}(\mathbf{x};\mathbf{q}) = \mathbf{s}_0 + \sum_{i=1}^{4} q_i \varphi_{N_i}, \tag{9.17}$$

where $(q_1, q_2, q_3, q_4) = (a, b, t_x, t_y)$. This representation enables us to augment the AAM described very easily.

The fitting of the augmented AAM is achieved by minimization of an altered form of (8.31)

$$F_{po}(\mathbf{q}, \mathbf{b}_s) = \left\| I(\mathbf{N}(\mathbf{W}(\mathbf{x};\mathbf{b}_s));\mathbf{q}) - g_0(\mathbf{x}) \right\|^2_{\text{span}(\mathbf{g}_i)^\perp}, \tag{9.18}$$

with respect to both $\mathbf{q}$ $\mathbf{b}_s$ simultaneously.

Fitting both the similarity transformation and the AAM deformation parameters is done in almost the way as described in the previous sections.

9.4.1 Warping

The warp with both the global transformation and the piecewise affine warp is denoted as,

$$\mathbf{N} \circ \mathbf{W}(\mathbf{x};\mathbf{q},\mathbf{b}_s) = \mathbf{N}(\mathbf{W}(\mathbf{x};\mathbf{b}_s);\mathbf{q}). \tag{9.19}$$

The warp is computed as follows: First the shape is deformed by,

$$\mathbf{s} = \mathbf{s}_0 + \sum_{i=1}^{m} b_{s_i} \varphi_{s_i}. \tag{9.20}$$

The next step is to warp every vertex with $\mathbf{N}(\mathbf{x};\mathbf{q})$. This can be done by first computing the destination of a single triangle of $\mathbf{s}_0$ under,

$$\mathbf{s}_0' = \mathbf{s}_0 + \sum_{i=1}^{4} q_i \varphi_{N_i}. \tag{9.21}$$

Using the equations from appendix A.1 the affine warp for this triangle is computed, and subsequently applied to every vertex of $\mathbf{s}$. Thus the destination of $\mathbf{s}_0$ under the warp composed warp $\mathbf{N} \circ \mathbf{W}(\mathbf{x};\mathbf{q},\mathbf{b}_s)$ is found.

9.4.2 Computing the Jacobian

Also needed is the Jacobian of $\mathbf{N} \circ \mathbf{W}(\mathbf{x};\mathbf{q},\mathbf{b}_s)$. It consist of two parts $\left(\frac{\partial \mathbf{N}\circ\mathbf{W}}{\partial \mathbf{q}}, \frac{\partial \mathbf{N}\circ\mathbf{W}}{\partial \mathbf{b}_s}\right)$. However it the computation is straightforward. Since, $\mathbf{W}(\mathbf{x};\mathbf{0}) = \mathbf{N}(\mathbf{x};\mathbf{0}) = \mathbf{0}$,

$$\frac{\partial \mathbf{N} \circ \mathbf{W}(\mathbf{x};\mathbf{0},\mathbf{0})}{\partial \mathbf{q}} = \frac{\partial \mathbf{N}(\mathbf{x};\mathbf{0})}{\partial \mathbf{q}} \tag{9.22}$$

$$\frac{\partial \mathbf{N} \circ \mathbf{W}(\mathbf{x};\mathbf{0},\mathbf{0})}{\partial \mathbf{q}} = \frac{\partial \mathbf{W}(\mathbf{x};\mathbf{0})}{\partial \mathbf{b}_s} \tag{9.23}$$

$$\tag{9.24}$$

Thus each Jacobian can be compute separately. The computation of $\frac{\partial \mathbf{N}\circ\mathbf{W}}{\partial \mathbf{q}}$ is exactly as described for $\frac{\partial \mathbf{N}\circ\mathbf{W}}{\partial \mathbf{b}_s}$ in 9.1 only with φ_{N_i} instead of φ_{s_i}. The combined warp Jacobian is just the concatenation of the two warp Jacobian.

9.4.3 Warp Inversion

The inverse of the warp $\mathbf{N} \circ \mathbf{W}(\mathbf{x};\mathbf{q},\mathbf{b}_s)$ is computed as in section 9.2. Replacing $\mathbf{W}$ with $\mathbf{N} \circ \mathbf{W}$ yields,

$$\mathbf{N} \circ \mathbf{W}(\mathbf{x};\mathbf{q},\mathbf{b}_s)^{-1} = \mathbf{N} \circ \mathbf{W}(\mathbf{x};-\Delta\mathbf{q},-\Delta\mathbf{b}_s), \tag{9.25}$$

to a first order approximation in $\Delta\mathbf{b}_s$ and $\Delta\mathbf{q}$.

9.4.4 Warp Composition

Composing the warp $\mathbf{N} \circ \mathbf{W}(\mathbf{x}; -\Delta\mathbf{q}, -\Delta\mathbf{b}_s)$ with $\mathbf{N} \circ \mathbf{W}(\mathbf{x}; -\Delta\mathbf{q}, -\Delta\mathbf{b}_s)$ is done in the same manner as in section 9.3. Each warp $\mathbf{W}$ and $\mathbf{N}$ is treated separately, and the procedure of section 9.3 is followed, only for $\mathbf{N}$ there is no need to take the mean of several warps since it is the same for all triangles.

9.4.5 Appearance Variation

As for the the warp $\mathbf{W}$ the combined Jacobian must be multiplied with the gradient of the template $\nabla\mathbf{g}_0$ resulting in steepest descent images as described above. To include appearance variation these must be projected into the subspace $\text{span}(\mathbf{g}_i)_\perp$. As described in section 8.5 this is done by subtracting the components in the direction of the vector $\mathbf{g}_i$, as in

$$\begin{aligned} SD_j(\mathbf{x}) &= \nabla\mathbf{g}_0(\mathbf{x})\frac{\partial\mathbf{N}(\mathbf{x};\mathbf{0})}{\partial q_j} - \sum_{i=1}^{m}\left[\sum_{\mathbf{x}} \mathbf{g}_i(\mathbf{x}) \cdot \nabla\mathbf{g}_0(\mathbf{x})\frac{\partial\mathbf{N}(\mathbf{x};\mathbf{0})}{\partial q_j}\right]\mathbf{g}_i(\mathbf{x}) \\ SD_{j+4}(\mathbf{x}) &= \nabla\mathbf{g}_0(\mathbf{x})\frac{\partial\mathbf{N}(\mathbf{x};\mathbf{0})}{\partial q_j} - \sum_{i=1}^{m}\left[\sum_{\mathbf{x}} \mathbf{g}_i(\mathbf{x}) \cdot \nabla\mathbf{g}_0(\mathbf{x})\frac{\partial\mathbf{N}(\mathbf{x};\mathbf{0})}{\partial q_j}\right]\mathbf{g}_i(\mathbf{x}) \end{aligned} \tag{9.26}$$

9.5 The AAM Search

Now the framework for fitting an AAM to an image has been formulated. The actual AAM search proceeds as follows.

An initial model instance is created by use of the mean shape and the mean texture. The topic of choosing an initial position of the AAM, is a research field in itself, and it is not covered in this thesis. A much used method is to divide the image into subregions, and subsequently starting the AAM fit in each location. A measure of the quality of the fit would be to subtract the segmented image region from the texture instance of the model. But, here it is assumed that the initial location of the model instance is known.

From the starting position, the AAM iteratively move around in the image, until a stopping criteria is reached; either convergence to the object in the image or a maximum number of iterations has passed.

An example of an AAM search can be seen in figure 9.6.

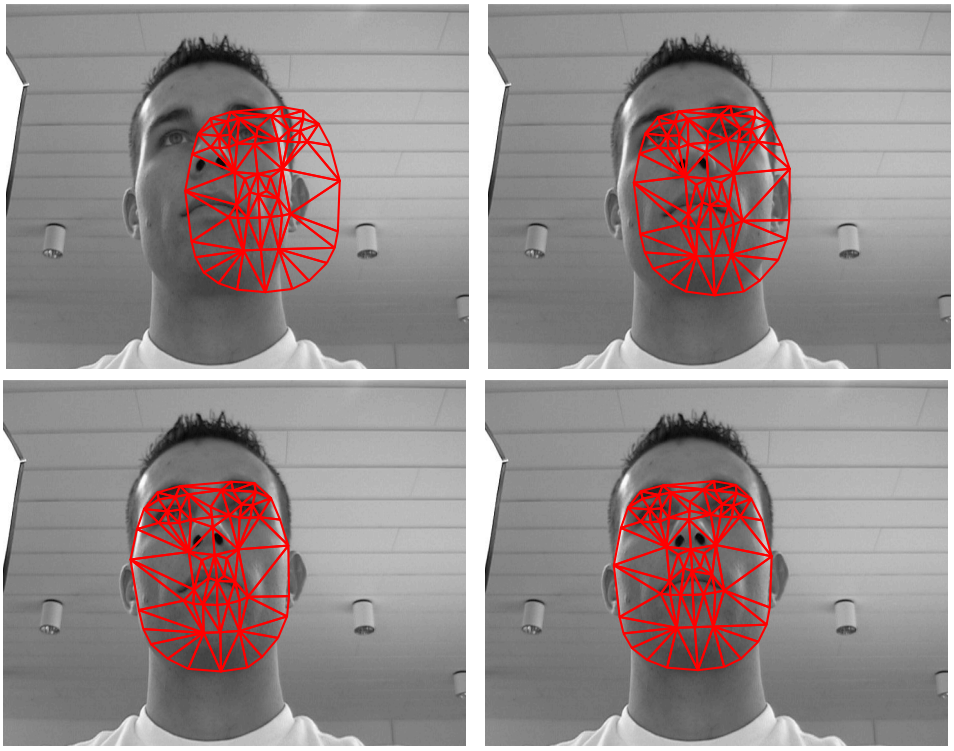


Figure 9.6: An example of an AAM search.

9.6 Summary

This chapter has described the adaptation of the inverse compositional algorithm to the AAM. A global shape transformation has been appended, enabling the model instance to translate, rotate and change scale during the course of the search. What is left is extracting the 3D pose of the head from the fit of the AAM. This is the topic of the next chapter.

Chapter 10

Extracting Head Pose

Extraction of 3D information from a 2D image is a very active field of research. With a single camera, the discipline is called Structure from Motion. The goal is from a sequence of images of an object, to estimate the 3D structure of the object, and the 3D motion of the camera/object. In this chapter a method for extraction 3D information from the fit of a 2D AAM is formulated.

Recovering the structure and motion from for rigid objects, the dominant algorithm is called *factorization*[2]. In the case of non-rigid objects, such as faces, the factorization algorithm must be adapted. Recently Xiao et al.[96] introduced a new approach to the solution. However, the method is quite involved and success is not always guaranteed. In [95] Xiao et al. introduces a method for utilizing the structure from motion algorithms with an AAM to obtain a 3D representation, based on the fit of a 2D AAM. This chapter is a brief overview of the algorithm. For details consult [95].

10.1 Computing 3D Shape from an AAM

If we have a training sequence of images of an object with corresponding shape annotations, a 3D version of the object can be recovered. Stacking all shape vectors in a matrix,

$$\mathbf{S} = \begin{pmatrix} x_1^0 & x_2^0 & \dots & y_n^0 \\ x_1^1 & x_2^1 & \dots & y_n^1 \\ \vdots & \vdots & \vdots & \vdots \\ x_1^N & x_2^N & \dots & y_n^N \end{pmatrix}, \tag{10.1}$$

where x_1^0 indicates vertex 1 frame 0.

The shape $\bar{\mathbf{s}}$ of a non-rigid 3D object can be modeled as a linear combination of a set of K basis shapes,

$$\bar{\mathbf{s}} = \bar{\mathbf{s}}_0 + \sum_{i=1}^{K} b_{3D_i} \bar{\mathbf{s}}_i. \tag{10.2}$$

where $\bar{\mathbf{s}}_0$ corresponds to the perfectly rigid shape. Assuming a weak perspective camera model, the coordinates of the 2D image points observed at each frame f are related to the 3D coordinates by,

$$\mathbf{s}^f = \begin{pmatrix} x_1^f & \dots & x_n^f \\ y_1^f & \dots & y_n^f \end{pmatrix} = \mathbf{R}^f \left(\bar{\mathbf{s}}_0 + \sum_{i=1}^{K} b_{3D_i} \bar{\mathbf{s}}_i \right) + \mathbf{T}^f, \tag{10.3}$$

where $\mathbf{R}_f$ is a 2×3 matrix containing the first two rows of the camera rotation matrix, and $\mathbf{T}_f$ is a 2×1 translation vector. Translation of each shape so its center of gravity is located at origo, eliminates $\mathbf{T}^f$ from the equations.

The matrix $\mathbf{S}$ can be represented as

$$\mathbf{S} = \begin{pmatrix} \mathbf{R}^0 & b_{3D_1}^0 \mathbf{R}^0 & \dots & b_{3D_K}^0 \mathbf{R}^0 \\ \vdots & \vdots & \vdots & \vdots \\ \mathbf{R}^N & b_{3D_1}^N \mathbf{R}^N & \dots & b_{3D_K}^0 \mathbf{R}^N \end{pmatrix} \begin{pmatrix} \bar{\mathbf{s}}_0 \\ \dots \\ \bar{\mathbf{s}}_K \end{pmatrix}. \tag{10.4}$$

Using singular value decomposition on $\mathbf{S}$ it can be factored into,

$$\mathbf{S} = \mathbf{MB}. \tag{10.5}$$

This factorization is not unique and is only determined up to a linear transformation. Any invertible matrix $\mathbf{G}$ can be inserted,

$$\mathbf{S} = \mathbf{MIB} = \tilde{\mathbf{M}}\mathbf{G}\mathbf{G}^{-1}\tilde{\mathbf{B}}. \tag{10.6}$$

Determining $\mathbf{G}$ can be solved by a number of different algorithms, we use one proposed by Brand[15]. Once $\mathbf{G}$ has been determined the corrected versions of $\mathbf{M} = \tilde{\mathbf{M}}\mathbf{G}$ and $\mathbf{B} = \mathbf{G}\tilde{\mathbf{G}}$ can be recovered.

The 3D model of the object is contained in the matrix $\mathbf{B}$. It is a model of the same object as the 2D AAM models. Thus, from (10.3) it is seen that the 2D shape is just a 3D shape multiplied with a camera matrix $\mathbf{R}$. A minimization problem can then be formalized, to obtain $\mathbf{R}$,

$$\min_{\mathbf{R},\bar{\mathbf{b}}_{3D}} \left\| \left(\mathbf{s}_0 + \sum_{i=1}^{m} b_{s_i} \varphi_{\mathbf{s_i}} \right) - \mathbf{R} \left(\bar{\mathbf{s}}_0 + \sum_{i=1}^{m} b_{3D_i} \mathbf{s}_i \right) \right\|^2 = 0. \tag{10.7}$$

The first parenthesis is just the fitted 2D AAM instance obtained using the method described in the previous chapter. From $\mathbf{R}$ the head pose can be extracted.

10.2 Summary

In this chapter, a method for recovering the 3D pose of the head has been formulated.

Chapter 11

Discussion

In this chapter, the previous chapters in this part are summarized. The main forces and drawbacks of the method are discussed.

11.1 Forces

The AAM is a generative model, capable of capturing and synthesizing objects learnt from a training set. Thus the AAM is applicable in a wide variety of problems ranging from segmentation of medical images, face recognition, and of course face tracking. It can model any object with distinct shape and texture.

After a suitable initialization, the algorithm converges in a matter of a few iterations.

The AAM is truly data-drivin, no parameters has to be selected prior to a new segmentation problem.

11.2 Drawbacks

The successful application of an AAM is strongly dependent on an annotated training set. A training set may consist of hundreds of images and annotation may be cumbersome.

Using the PCA and the assumption of gaussian distributed parameters may lead to illegal shapes.

Part II

Eye Tracking

Chapter 12

Introduction

A fundamental part of eye tracking in full-face videos, is the detection and tracking of the face. This topic is solved by the Active Appearance Model described in last part.

In this part various methods of eye trackers are presented. These can be divided into two parts - Segmentation-based and Bayesian tracking. We end up by determining the direction of gaze.

The eye images are extracted from the video frames based on input from AAM. The resulting fit of one frame is shown in figure 12.1. Each eye region is spanned by a number of vertices. A bounding box containing the eye is extracted on which the eye tracking methods are applied.

12.1 Recent Work

Detection of the human eye is a difficult task due to a weak contrast between the eye and the surrounding skin. As a consequence, many existing approaches uses close-up cameras to obtain high-resolution images[36][93]. However, this imposes restrictions on head movements. The problem can be overcome by use of a two camera setup[92][97]. One camera covering the head and controlling a second camera, which focuses on one eye of the person. Matsumoto and Zelinsky[56] utilizes template and stereo matching.

In many existing approaches the shape of iris is modeled as a circle [47][48][56][97]. Since the shape and texture of the object is known, a template model can be used with advantage[43][81]. J. Gracht et al.[91] utilizes an iris template generated by a series of wavelet filtering.

Wang et al.[92] detects the iris using thresholding, morphology and vertical edge operators. An ellipse is fitted to the resulting binary image.

Bagci et al.[8] propose a Hidden Markov Model discretizing the position

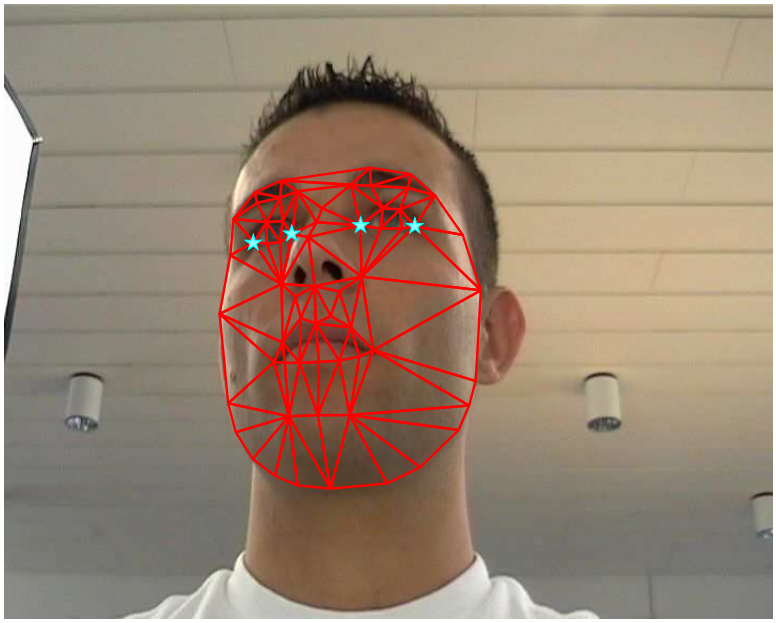


Figure 12.1: The resulting AAM fit of one frame. The eye images are extracted from the video frames based on input from AAM. Each eye region is spanned by a number of vertices. A bounding box containing the eye is extracted on which the eye tracking methods are applied.

of an eye into five states - looking up, down, left, right and forward. The model uses color and geometrical features.

Most algorithms tend to fail when the eyes blink. This can be handled by an eyelid detector. Tian et al. presents a dual-state parametric eye model[90], which is used to detect the different eye states - *open* or *closed*. An open eye is parameterized by a circle and two parabolic arcs describing the eyelids. The closed eye is described by a straight line. The inner eye corners are tracked by a modified version of Lucas-Kanade tracking algorithm.

A probabilistic formulation of eye trackers has the attraction that uncertainty is handled in a systematic fashion. Xie et al.[97] utilizes a Kalman filter with purpose to track the eyes. The eye region is detected by thresholding and the center of an eye is used for motion compensation. The center of this iris is chosen as tracking parameter, while the gray level of the circle modeled eye is chosen as measurement[98]. Hansen and Pece propose an active contour model combining local edges along the contour of the iris[36]. The contour model is utilized by a particle filter.

A generative model explaining the variance of the appearance of the eye is developed by Moriyama et al.[59]. The system defines the structures and

motions of the eye. The structure represents information regarding size and color of iris, width and boldness of eyelid etc. The motion is represented by the position of upper and lower eyelids and 2D position of the iris. Witzner et al. utilizes an AAM[35] .

Based on the center of iris estimate, the gaze direction can be computed utilizing various methods. Stiefelhagen et al.[81] utilizes a neural network with the eye image as input. Witzner et al.[35] uses a Gaussian process interpolation method for inferring the mapping from image coordinates to screen coordinates. Ishikawa et al. [43] exploits a geometric head model, which translates from 2D image coordinates to a direction in space relative to the initial frame.

12.2 Overview

An overview of different eye trackers are presented in the following; from fast heuristic to advanced probabilistic methods. The appearance of the eye can be utilized similarly to the method proposed for face detection in part I. However, to ensure robustness verses changing light conditions, the methods modeling the appearance are kept relatively simple. In chapter 13 template matching, a deformable template model, and a fast heuristic method are presented. In chapter 14 the shape of iris is handled by an active contour method in a Bayesian framework.

The purpose of the eye trackers is to estimate the center of pupil accurately. Based on the pupil location, pose and scale of face, the gaze direction can be determined. Chapter 15 describes the geometric model for gaze determination.

Chapter 13

Segmentation-Based Tracking

One of the most basic, but important, aspects of object tracking is how to find the object under consideration in the scene. Partitioning of an image into object and background is called segmentation. Segmentation of an image is in practice the classification of each image pixel to one of the image parts, which are visually distinct and uniform with respect to some property, such as gray level, gradient information, texture or color.

In this case the object is the eye; or more precisely the center of iris. In many existing approaches the shape of iris is modeled as a circle[47][48][56][97].

This assumption holds when the camera lies on the optical axis of the eye. When the gaze is turned off this axis, the circle is rotated in 3D space, and can then be interpreted as an ellipse in the image plane. Thus, the shape of the contour changes as a function of the gaze direction and the camera pose.

In this chapter various methods for segmentation-based eye tracking are presented. Thresholding is a simple, but widely used, approach for image segmentation. However, choosing the optimal threshold value can be difficult. As a consequence, a double threshold method is utilized in section 13.1.

A template model can be used with advantage, since the shape and texture of the object is known. A template using gray level intensities is described in section 13.2, while a color scheme is found in section 13.3. The appearance of the eye changes together with the gaze direction, face and camera pose. This is utilized in section 13.4, where a template matching model is relaxed to be deformable.

13.1 Thresholding

Thresholding is a traditionally low-level method for segmentation. The value of the threshold decides whether a pixel belongs to object or background[16].

Thus, any pixel with intensity value above the threshold is labeled as object, while values below are labeled as background. This is of great advantage when separating two classes which have intensity levels grouped into two well-separated modes. This kind of simple global thresholding can be expected to be successful when the illumination and other environments are static or at least highly controlled.

Conversely, if the environments changes over time, the intensity values of object and background will also change over time, which will influence on the performance. In that case, no fixed threshold can be chosen as in simple global thresholding. Moreover, only the intensities are considered, thus any relationships between the pixels are not considered. Therefore, this kind of segmentation is very sensitive to noise. Several improved methods exists such as preprocessing by different filters (log, average etc.)[30] with purpose to suppress noise and smooth out the intensities, histogram equalization which spreads out the intensity values in order to increase the dynamic range and thereby enhance the image contrast.

Furthermore, statistical methods like Otsu's method[63] exist, which seeks to minimize the error of classifying a background pixel as object or vice versa. In that sense, we seek to minimize the area under the given image intensity histogram for a region that lies on the other region's side of the threshold. The threshold is chosen to minimize the intraclass variance of the black and white pixels.

13.1.1 Double Threshold

We propose to use adaptive double thresholding to estimate the center of the pupil[77]. The object we seek is more or less completely black and assumed to be darker than the background. Consequently, the description above in section 13.1 is inverted; thus any pixel *below* the threshold is labeled as object. Since the environments changes with time, we choose two threshold values T_1 and T_2 where $T_1 < T_2$. Thus T_2 should at least capture the object, but unfortunately some of the background too. The low threshold T_1 is purposed to capture at least a part of the object. If T_1 is too low, the value is increased adaptively until a given stop criteria, which is set to avoid over-fitting. Survivors of both thresholds are accepted as object. In this way, the low threshold T_1 is used to accept objects accepted by the higher threshold T_2, where it is assumed that the entire object captured. A 1D example is shown in figure 13.1.

Double adaptive threshold is applied on an eye image in figure 13.2. The center of pupil is estimated by calculating the *center of mass*[16] of the object.

One of the main difficulties when segmenting an eye image, is how to

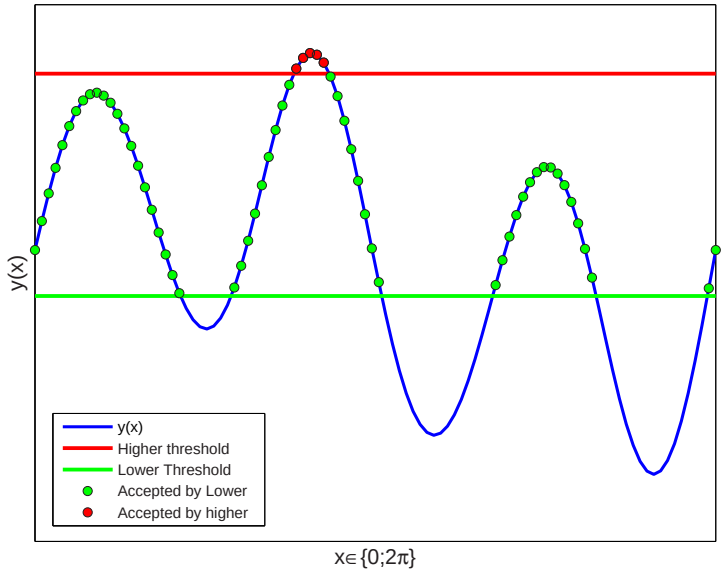


Figure 13.1: The 1D function $y(x)$ is seen to have multiple local maxima. Searching for the object which contain the global maximum, it is seen that neither the high nor low threshold (red or green) seems to be suitable. However, by combining these in a way such that only survivors of both thresholds are accepted as object, the resulting object will be appropriate. The high threshold can be interpreted as a filter regarding the low threshold.

handle the high-intensity corneal reflections seen as white blobs. This phenomenon is explained intuitively by approximating the eye with a convex mirror, which reflects the illumination in one direction. As a consequence, the estimate may be biased dependent on the light conditions. Using knowledge about the iris shape, the estimate can be further improved by weighting the center of gravity estimate or by incorporation of white blobs inside the iris.

13.2 Template Matching

Template matching is a technique for comparing data, in this an case image, with a stored reference template and subsequently assigning a score based on the level of similarity. In its simplest form it is equivalent with cross correlation, which is defined in the spatial domain as,

$$C(s,t) = \sum_x \sum_y I(x,y)T(x-s,y-t), \tag{13.1}$$

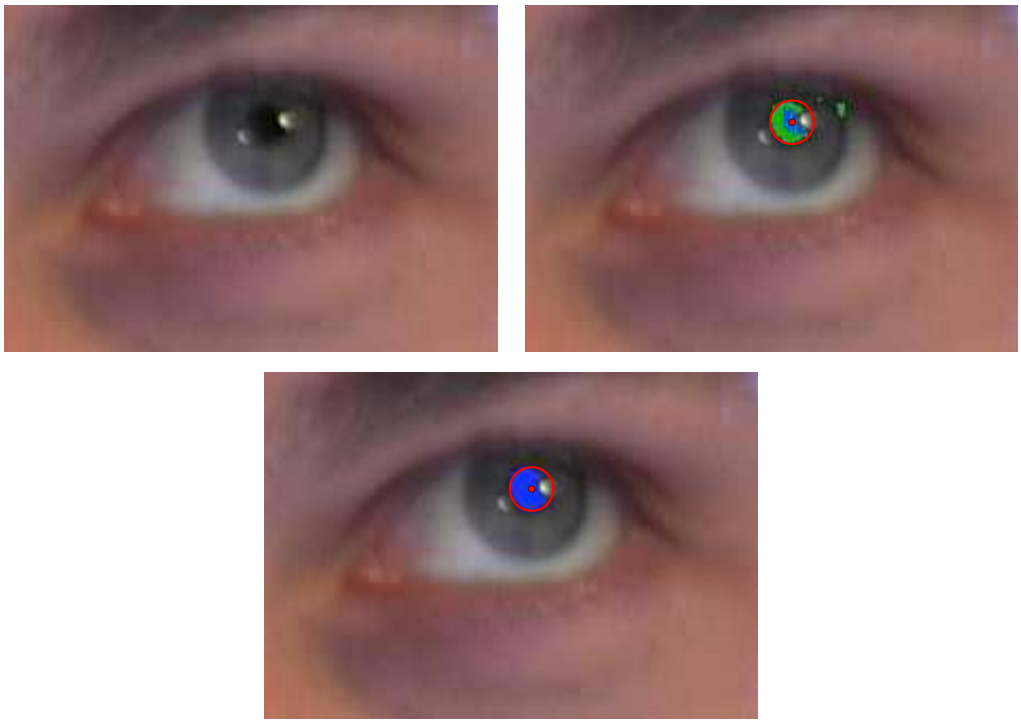


Figure 13.2: The pupil is approximately black and assumed to be darker than the background.(*Top left*) Input image. (*Top right*) The high threshold (green) captures the object but unfortunately some of the background too, while the low threshold captures a small part of the object (blue). (*bottom*) Survivors of both thresholds are accepted as object. The center of pupil is estimated by calculating the *center of mass* of the object.

where $I^{M\times N}$ is an image, T is a template, (x, y) the pixels corresponding to the template, $s = 0, 1, \ldots, M-1$ and $t = 0, 1, \ldots, N-1$. It is usually assumed that the object of interest do not change appearance very much. The template may be an intensity/binary image or a parameterization.

The method is very popular due to the relative simple implementation and fine results in applications such as pattern recognition, tracking etc. However, the sensitivity to typical problems as noise, partial occlusions and varying illumination may be a huge drawback. Furthermore, the computation time may be slow dependent on the size of the template. There are several fast optimized algorithms that can be used to speed up the matching process [70][99].

13.2.1 Iris Tracking

The iris detector proposed by T. Ishikawa et al. [43] consists of two parts. Initially, two different templates are applied to approximately locate the iris. The first template is a black disk template which is matched against the intensity image. The second is a ring template which is matched against the vertical edge image. The radius of both templates can be determined manually or by more sophisticated methods such as from the fit of an AAM. The template matching confidence is then determined by summing the two sets of matching scores. The position with maximum confidence is chosen as the initial estimate of the center of the iris.

This estimate is then refined using an ellipse fitting algorithm[86] to detect the pupil contour. Initially, the edges are detected by scanning radially from the center and outward as seen in figure 13.3. The maximum gradient on each line is defined as an edge. The estimate is improved by fitting an ellipse to the detected edges. The general form of an ellipse is defined as,

$$a_1x^2 + a_2xy + a_3y^2 + a_4x + a_5y = 1, \tag{13.2}$$

where the parameters $a_1, \ldots, a_5$ are fit using least squares. The center, shape and orientation of the ellipse are obtained by converting the general equation to standard form[69]. The standard form, with major axis parallel to the x-axis, is formulated as the coordinates (x, y) satisfying the following equation,

$$\frac{(x-c_x)^2}{\lambda_1^2} + \frac{(y-c_y)^2}{\lambda_2^2} = 1, \tag{13.3}$$

where (c_x, c_y) is the center, λ_1 and λ_2 are the length of major and minor axes. The angle between major and vertical axis θ is applied by the rotation,

$$\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} \cos(\theta) & \sin(\theta) \\ -\sin(\theta) & \cos(\theta) \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix}. \tag{13.4}$$

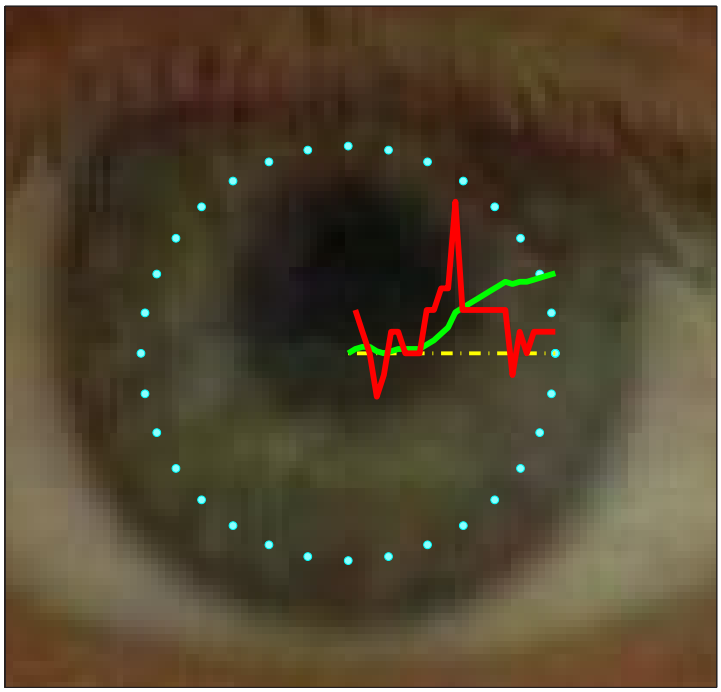


Figure 13.3: The initial estimate of the center of the iris is refined by fitting an ellipse to the radial edges of the pupil, which is found by scanning radially from the center and outward towards the k points on a circle. The position with maximum gradient is chosen as edge. The yellow line shows one of the k lines, while the green line is the relative intensity level and the red line the relative gradient on this line.

The refinement procedure is repeated until convergence, which should in general be no more than 2-3 iterations[43]. The template matching confidence for a given image and the improved estimate of the center of iris is depicted in figure 13.4.

13.2.2 Outlier Detection

Improving the iris location by radial edge detection is unfortunately not trivial. This is caused by the relative weak edges in the image and high-intensity corneal reflections.

Ishikawa et al. [43] propose to increase the robustness by filtering out edges, which are long away from the initial estimate. Nevertheless, this seems to be a poor solution.

If the estimate of the center of iris is acceptable, the Euclidian distance should be approximately equal. Therefore, we propose to constrain on the common distances. The algorithm is shown below, where $c_{xy}^{2\times 1}$ is the esti-

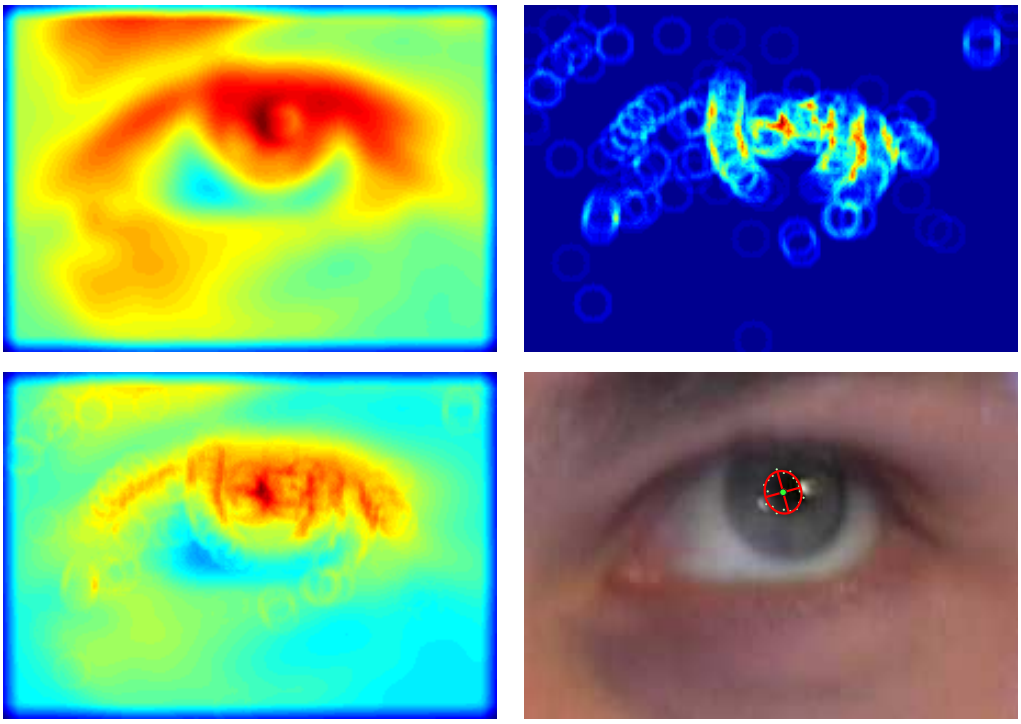


Figure 13.4: Note that warm colors should be interpreted as high values. *(Top left)* Black disk template matched against the intensity image. *(Top right)* Ring (annulus) template matched against the vertical edge image. *(Bottom left)* The template matching confidence is determined by summing the two sets of matching scores *(top left)* and *(top right)*. *(Bottom right)* The position with best confidence is chosen as the initial estimate of the center of the iris, which is refined using an ellipse fitting algorithm. Note that the algorithm is applied on graytone intensity images.

mated center of iris, $\mathbf{X}^{2\times k}$ the estimated edge points on the lines $\mathbf{L}$ spanned by $c_{xy}^{2\times 1}$ and the points $\mathbf{P}^{2\times k}$ on the circle with a given radius. k represents the dimension and is typed as upper case, while lower case indexes represents one specific line.

Initially, the domain of lines are ranging from the center to the points lying on a circle (13.5). The Euclidian distance between center and the estimated edges $\mathbf{X}^{2\times k}$ are calculated (13.6). The difference g^{k-1} between the distances are obtained by differentiating (13.7). If the difference is greater (13.8) or less (13.9) than some tolerance, the domain of $\mathbf{L}$ is adjusted and the edge estimate is recalculated. The algorithms runs until convergence or a given stopping criteria is reached. The outlier detection step is depicted in figure 13.5.

$$\mathbf{L} \in \left[c_{xy}^{2\times 1}; \mathbf{P}^{2\times k}\right]; \tag{13.5}$$

```
while (not Converged)
```

$$d^k = \mathrm{dist}(c_{xy}, \mathbf{X}^{2\times k}); \tag{13.6}$$

$$g^{k-1} = \mathrm{diff}(d^k); \tag{13.7}$$

```
for i = 1 : k − 1
```

```
if (g_i > tol)        (Too far away...)
```

$$\mathbf{L}_i \in [c_{xy}; \mathbf{X}_i]; \tag{13.8}$$

$$\mathbf{X}_i = \mathrm{edge}(\mathbf{L}_i);$$

```
elseif (g_i < tol)    (Too near centroid...)
```

$$\mathbf{L}_i \in [\mathbf{X}_i; \mathbf{P}_i]; \tag{13.9}$$

$$\mathbf{X}_i = \mathrm{edge}(\mathbf{L}_i);$$

```
end
if X_i = ∅            (If no edge, discard point)
```

$$\mathbf{X}_i = [\,];$$

```
end
end
end
```

13.3 Color-Based Template Matching

Template matching using gray level intensities on eye images, suffers from the poor contrast between the skin surrounding the eye and iris. The RGB

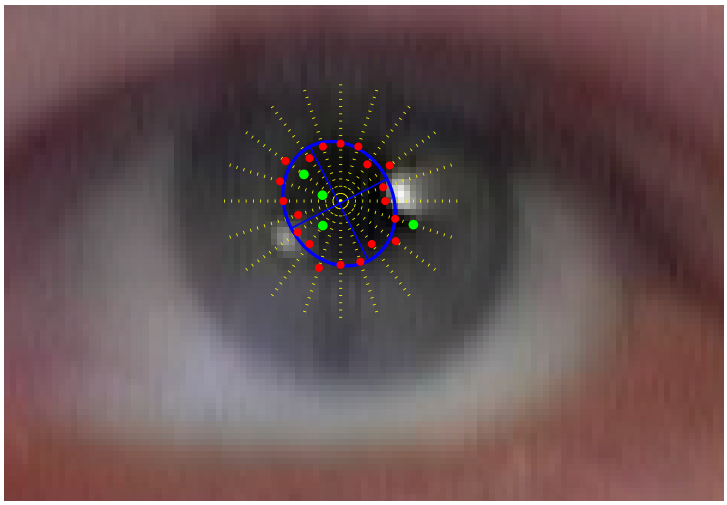


Figure 13.5: The outlier detection in the refinement step constrains the common distances from the center. If the difference between neighbor distances differs from a tolerance level, the domain of a given line is reduced to force the algorithm to choose a more likely position. The lines $\mathbf{L}$ in yellow is spanned by $c_{xy}^{2\times 1}$ and the points $\mathbf{P}^{2\times k}$ on the circle. Green points have been rejected, the red is accepted and the final estimate of pupil contour is depicted in blue.

(**R**ed, **G**reen and **B**lue) colors of the eye region differs significantly from the surrounding skin. As a consequence, the template matching from section 13.2 is modified to color images.

The black disk template is now matched against each of the RGB channels and converted to a confidence for each location in the image by the norm of each pixel.

Now, instead of limiting ourselves to use vertical edges, the Canny's direction of gradient[78] is applied. A color image can be considered as a mapping from a position $\mathcal{R}^2(x, y)$ to a color $\mathcal{R}^3(r, g, b)$. The Jacobian of this mapping is defined as,

$$\mathbf{J} = \begin{bmatrix} \mathrm{d}r/\mathrm{d}x & \mathrm{d}r/\mathrm{d}y \\ \mathrm{d}g/\mathrm{d}x & \mathrm{d}g/\mathrm{d}y \\ \mathrm{d}b/\mathrm{d}x & \mathrm{d}b/\mathrm{d}y \end{bmatrix}. \tag{13.10}$$

The largest eigenvalue λ and the corresponding eigenvector v of the covariance $\mathbf{C} = \mathbf{J}^T\mathbf{J}$, is the direction of maximum change at a given point. The eigenvalue is the square of the change in magnitude in the direction of v. Hence, the expression $(\sqrt{\lambda}) \cdot v$ is equivalent to Canny's direction of gradient.

The final edge estimate is found by thresholding the values computed by Canny, which is further improved by mathematical morphology - *thining*[16].

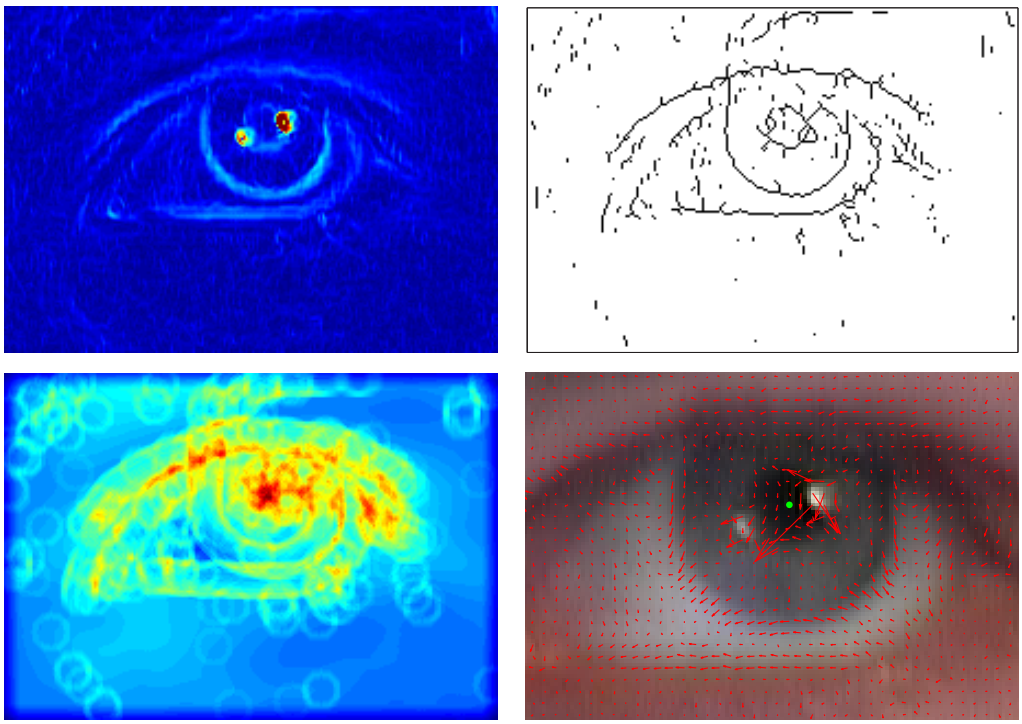


Figure 13.6: *(Top left)* Canny's direction of gradient applied on a eye color image. *(Top right)* The edge detected image is computed by thresholding and subsequently applying the mathematical morphology operator *thining* on the gradient. *(Bottom left)* The template matching confidence is determined by summing the two sets of matching scores - Black disk template matching on the input image and ring (annulus) template matching against the edge detected image in *(top right)*. *(Bottom right)* The final estimate of the center of iris is shown as the green dot, while the red arrows depicts Canny's direction of gradient.

Finally, the ring template is matched against the edge image. The matching scores of the two templates are normalized to prevent the features of one filter to dominate the other and finally summed to get the resulting confidence. The position with maximum confidence is chosen as the estimate of the center of iris.

13.4 Deformable Template Matching

The gray level intensity template matching method is relaxed to be deformable in this section. The objective is, modified, to fit an ellipse to the pupil contour, which is characterized by a darker color compared to the iris. The standard ellipse form, defined in (13.3) and (13.4), is parameterized by,

$$\mathbf{x} = (c_x, c_y, \lambda_1, \lambda_2, \theta)\,. \tag{13.11}$$

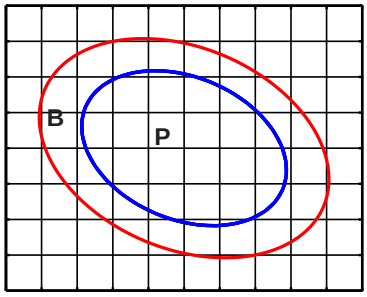


Figure 13.7: Region P is the inner circle, and region B is the ring around it. The regions have the same area. When region I contain the entire object, B must be outside the object, and thus the difference in average pixel intensity is maximal.

The pupil region P is the part of the image I spanned by the ellipse parameterized by $\mathbf{x}$. The background region B is defined as the pixels inside an ellipse, surrounding but not included in P, as seen in figure 13.7. When region P contains the entire object, B must be outside the object, and thus the difference in average pixel intensity is maximal. To ensure equal weighting of the two regions, they must have the same area. The area of the inner ellipse P is $A_P = \pi\lambda_1\lambda_2$. The shape parameters of B should satisfy the constraint on the area $A_{B/P} - A_P = A_P$. As a consequence, the parameters is defined as $\mathbf{x}_B = \left(c_x, c_y, \sqrt{2}\lambda_1, \sqrt{2}\lambda_2, \theta\right)$, while $\mathbf{x}_P$ is defined as (13.11).

The estimate of the pupil contour can now be estimated by minimizing the expression,

$$\mathcal{E} = \mathrm{Av}(P) - \mathrm{Av}(B), \tag{13.12}$$

where $\mathrm{Av}(B)$ and $\mathrm{Av}(P)$ are the average pixel intensity of the background - in this case the pupil - and iris region respectively.

The contrast between the iris and sclera is large. Intuitively, the inner ellipse should surround the iris because of the large contrast in proportion to the white sclera. However, the iris is surrounded by white sclera and a comparatively dark skincolor. This combination results in a weak average value. As a result, each region should contain relative monotone intensities. By letting the inner ellipse fit to the pupil, we ensure a consistent cost function.

The pupil tracker is initialized by previous state estimate, $\mathbf{x}_k = \mathbf{x}_{k-1}$, as we assume the changes from frame to frame are relatively small. The deformable template model is then optimized given the starting point. An example of the optimization of the deformable model is seen left in figure 13.9. However, rapid eye movements may cause the tracking algorithm to break

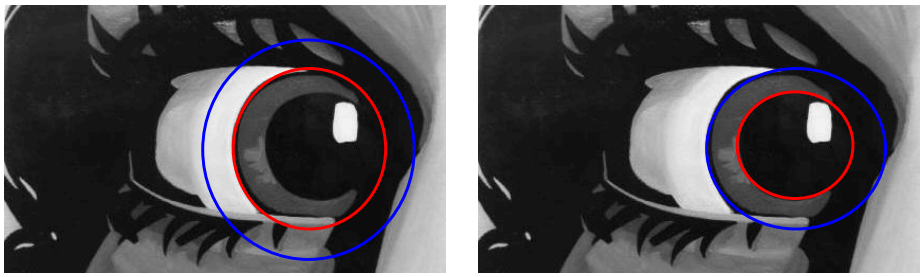


Figure 13.8: (*Left*) Although the contrast between the iris and sclera is large, the cost function is non-descriptive with this choice of P and B. (*Right*) By letting the inner ellipse fit to the pupil, we ensure relative monotone intensities of each region leading to a more consistent cost function.

down, since the starting point is too far from the true state as depicted right in figure 13.9. These movements occurs below the conscious level in order to improve the visual perception[66] and are among the fastest movements the body can make - A rotation speed at over $500\frac{deg}{sec}$[34]. The problem of rapid eye movements can be omitted by applying the Double Threshold method, described in section 13.1.1, as initial guess on the pose.

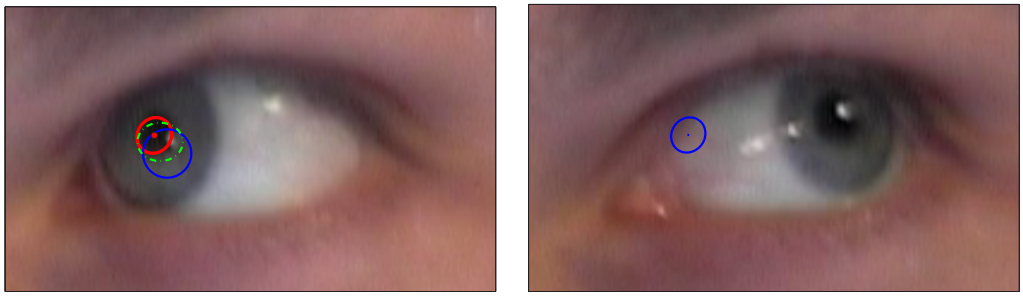


Figure 13.9: (*Left*) The blue ellipse indicates the starting point of the pupil contour. The template is iteratively deformed by an optimizer; one of the iterations are depicted in green. The red ellipse indicates the resulting estimate of the contour. (*Right*) Rapid eye movements may cause the tracker to break down. This can be omitted by applying the Double Threshold method, described in section 13.1.1, as initial guess on the pose.

13.4.1 Optimization

The minimization of the cost function defined in (13.12) can be formulated as a nonlinear optimization problem. The solution can be obtained by various optimization algorithms; among these is Newton's method[46]. The function is denoted $f(\mathbf{x})$, and the truncated Taylor expansion of this function is

approximated by a quadratic model q,

$$q(\mathbf{h}) = \mathrm{f}(\mathbf{x}) + \mathbf{h}^T \mathbf{f}'(\mathbf{x}) + \frac{1}{2}\mathbf{h}^T \mathbf{f}''(\mathbf{x})\mathbf{h}. \tag{13.13}$$

When $\mathbf{x}$ is sufficiently near the minimizer $\mathbf{x}^*$ and $\mathbf{f}''(\mathbf{x}^*)$ is positive definite, then q has a unique minimizer at a point where the gradient of q equals zero,

$$\mathbf{f}'(\mathbf{x}) + \mathbf{f}''(\mathbf{x})\mathbf{h} = 0. \tag{13.14}$$

However, the standard Newton method has some disadvantages; for instance the required analytical second order derivatives of f. This can be overcome by use of the more clever BFGS updating formulas[28]. Analytical derivatives are avoided by use of approximations, which are adjusted dynamically. In the end of search, the approximation converges toward $\mathbf{f}''(\mathbf{x}^*)$. Although convergence towards a stationary point has not been proved yet, BFGS with soft line search is known as a method that never fails[28].

The choice of a suitable initial step size for the optimizer is of great importance when estimating the gradient. An appropriate starting point $\mathbf{x}$, consisting of the five parameters defining an ellipse, is initialized around the pupil. One parameter is varied while the others are kept fixed. The partial derivative with respect to one of the parameters is defined,

$$\frac{\mathrm{f}(c_x + h, c_y, \lambda_1, \lambda_2, \theta) - \mathrm{f}(c_x, c_y, \lambda_1, \lambda_2, \theta)}{h} \simeq \frac{\partial \mathrm{f}}{\partial c_x} + \mathcal{O}(h) + \frac{\mathrm{error}}{h}. \tag{13.15}$$

It is seen from (13.15) that if h is chosen too small, the error will be large. Comparing with figure 13.10, it is seen that a step size around $h = 1$ is suitable in order to make the optimizer more flexible in the beginning and avoiding local minimums. As we get closer to the solution, the step size is decreased by the algorithm. The step size is, however, dependent on the image and the resolution.

13.4.2 Constraining the Deformation

Although a deformable template model is capable of catching changes in the pupil shape, there are also some major drawbacks. Corneal reflections, caused by illumination, may confuse the algorithm and cause it to deform unnaturally. In the worst case, the shape may grow or shrink until the algorithm collapses.

We propose to constrain the deformation of the model in the optimization step by adding a regularization term. Assume the parameters defining an

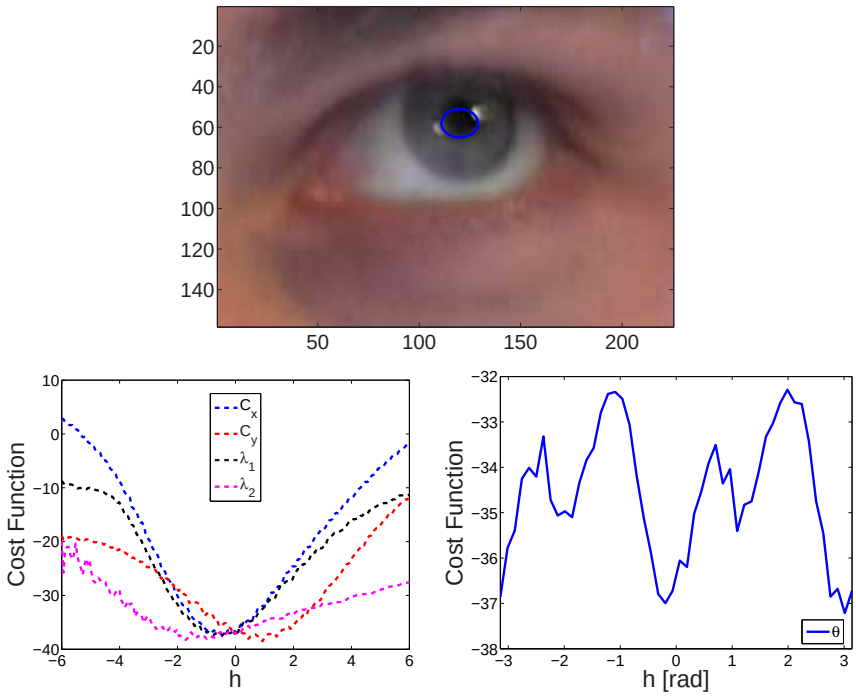


Figure 13.10: *(top)* A point is initialized near the iris. Alternately, one parameter is varied while the four other parameters are kept fixed. The cost function is evaluated. *(Bottom left)* Variation of center coordinates and shape parameters. *(Bottom right)* Variation of the orientation.

ellipse is normally distributed with mean μ and covariance $\mathbf{\Sigma}$. The prior distribution of these parameters are then defined,

$$p(\mathbf{x}) = \mathcal{N}(\mu, \mathbf{\Sigma}) \propto \exp\left(-\frac{1}{2}(\mathbf{x}-\mu)^T\mathbf{\Sigma}^{-1}(\mathbf{x}-\mu)\right), \tag{13.16}$$

where the normalization factor has been omitted. The mean and covariance are estimated in a training sequence. At last the optimization of the deformable template matching method is constrained by adding a regularization term,

$$\mathcal{E} = \mathrm{Av}(P) - \mathrm{Av}(B) + \mathcal{K}\left(1 - p(\mathbf{x})\right), \tag{13.17}$$

where $\mathcal{K}$ is the gain of the regularization term.

The relevance of constraining the deformation is visualized in figure 13.11. A suitable starting point $\mathbf{x}$ is chosen. The pose and orientation are kept fixed, while the shape parameters are varied. In this case the true shape parameters λ_1 and λ_2 are approximately eight. The image confidence as a function of the shape parameters is depicted to the left, while the prior distribution is seen in the middle of figure 13.11. Combining the image confidence with a prior according to (13.17) yields the constrained estimate, which is depicted to the right in figure 13.11.

By use of the shape constraints, we incorporate prior knowledge to the solution. The robustness is increased considerably and the parameters are constrained to avoid the algorithm to break down due to infinite increase or decrease of parameters.

The deformable template matching method is seen applied with and without constraints in figure 13.12. The constrained estimate is seen to be less sensitive to noise due to reflections.

13.5 Summary

A mixture of segmentation-based eye trackers have been presented. Starting by the simple - yet efficient - double thresholding, where two threshold values are chosen to find the dark pixels corresponding to the pupil. The low threshold can be interpreted as a filter regarding the high threshold.

Prior knowledge about the shape and appearance is utilized by the template matching models. A model concerning gray level intensities and a model regarding RGB color images have be described.

A common characteristic of the above mentioned trackers is that they do not incorporate knowledge from last frame. They are based on explicit feature detection using global information.

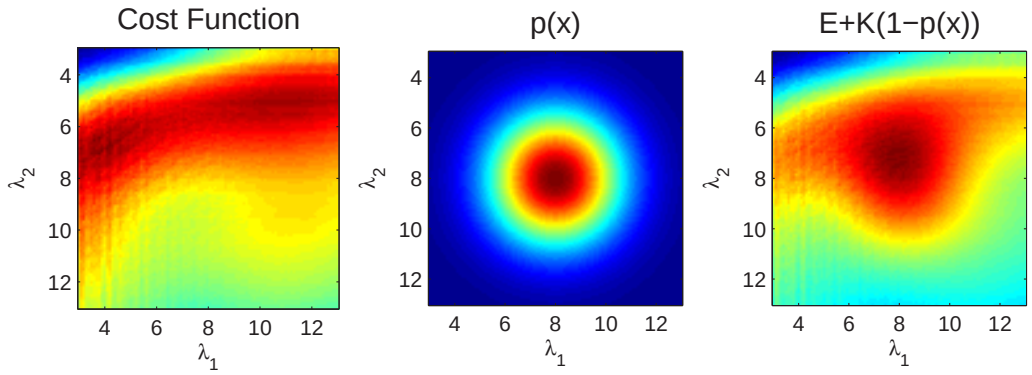


Figure 13.11: Given an appropriate starting point $\mathbf{x}$. The pose and orientation are kept fixed, while the shape parameters are varied. Note that the surface plots are not - as expected - smooth. This is due to rounding in the interpolation when evaluating the image evidence of the deformable template. *(Left)* The image confidence given the state - warmer colors means more likely. *(Middle)* The prior probability is a normal distribution with a given mean value μ and covariance Σ. *(Right)* Combining the image evidence and prior according to (13.17) yields the constrained estimate.

The appearance of the eye changes together with the gaze direction, face and camera pose. This is exploited by a deformable template matching model. The starting point of current frame can be chosen as the estimate from previous frame. However, rapid eye movements may confuse the tracker since the starting point is too far from the true state. This can be omitted by applying the Double Threshold method.

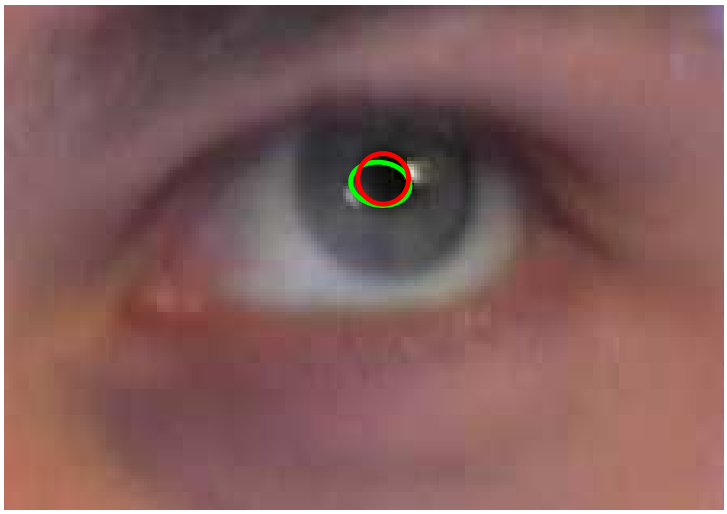


Figure 13.12: The deformable template matching method applied without constraints is seen in green, while the red ellipse depicts the constrained version. The constrained estimate is seen to be less sensitive to noise due to reflections.

Chapter 14

Bayesian Eye Tracking

The segmentation based trackers from chapter 13 are based on explicit feature detection using global information. The iris is circular and characterized by a large contrast to the sclera. Therefore, it seems obvious to use a contour based tracker. The chosen active contour method does not use features explicitly but maximizes feature values underlying the contour in a Bayes sense. Bayesian methods provide a general framework for dynamic state estimation problems. The Bayesian approach is to construct the probability density function of the state based on all the available information. For that reason the method is relative robust to environmental changes.

Moreover, the changes in iris position are very fast. As a consequence, the iris position cannot be assumed to follow a smooth and completely predictable model. Particle filtering is therefore suitable.

14.1 Active Contours

Witzner et al.[36] describes an algorithm for tracking using active contours and particle filtering. A generative model is formulated which combines a dynamic model of state propagation and an observation model relating the contours with the image data. The current state is then found recursively by taking the sample mean of the estimated posterior probability.

The proposed method in this chapter is based on [36], but extended with constraints and robust statistics.

14.2 Assumptions

The active contour method has some basic assumptions; Gray-level values of neighboring pixels are correlated, except when two pixels are known to lie

on each side of an object boundary as depicted in figure 14.2. In addition to this, the shape of the contour is subject to random Gaussian deformation at each sample point on the contours. This can be utilized by marginalizing over the deformation.

14.3 Dynamic Model

The dynamic model describes how the iris moves from frame to frame. As mentioned in chapter 13, the shape of iris can be modeled as a circle. However, when the gaze is turned off the optical axis, the circle is rotated in 3D space, which can be interpreted as an ellipse in the image plane. As a consequence, the iris is modeled as an ellipse and the state vector $\mathbf{x}$ consist of the five parameters defining an ellipse,

$$\mathbf{x} = (c_x, c_y, \lambda_1, \lambda_2, \theta)\,, \tag{14.1}$$

where (c_x, c_y) is the center, λ_1 and λ_2 are the length of major and minor axes, and θ is the angle between major and vertical axis.

To define the problem of tracking, consider the evolution of the state sequence $\mathbf{x}_{t+1} = \mathbf{f}_{t+1}\{\mathbf{x}_t, t \in \mathbb{N}\}$ of a target, given by

$$\mathbf{x}_{t+1} = \mathbf{f}_{t+1}(\mathbf{x}_t, \mathbf{v}_t), \tag{14.2}$$

where $\mathbf{f}_{t+1}$ is a possibly non-linear function of the state $\mathbf{x}_t$ and $\{\mathbf{v}_t, t \in \mathbb{N}\}$ is an independent identically distributed process noise sequence. The objective of tracking is to recursively estimate $\mathbf{x}_{t+1}$ from the measurements,

$$\mathcal{M}_{t+1} = \mathbf{h}_{t+1}(\mathbf{x}_{t+1}, \mathbf{n}_{t+1}), \tag{14.3}$$

where $\mathbf{h}_{t+1}$ is a possibly non-linear function and $\{\mathbf{n}_{t+1}, t \in \mathbb{N}\}$ is an i.i.d measurement noise sequence. Appendix B.1 contains a more detailed description of Bayesian state estimation.

The pupil movements can be very rapid and is therefore modeled as Brownian motions(AR(1)). Thus the evolution of the state sequence (14.2) is modeled,

$$\mathbf{x}_{t+1} = \mathbf{x}_t + \mathbf{v}_t, \quad \mathbf{v}_t \sim \mathcal{N}(0, \mathbf{\Sigma}_t), \tag{14.4}$$

where $\mathbf{\Sigma}_t$ is the time dependent covariance matrix of the noise. The time dependency compensates for scale changes, which affects the amount of movement. Larger movements is expected when the ellipse appears large, since the position of the eye is nearer to the camera. Contrary, when the eye is farther from the camera, smaller movements are expected. Hence, the first two diagonal elements of $\mathbf{\Sigma}_t$ corresponding to c_x and c_y are assumed to be linear dependent on previous sample mean.

14.4 Observation Model

The observation model consists of two parts; a geometric component defining a probability density function over image locations of contours and a texture component defining a pdf over pixel gray level differences given a contour location. The geometric component models the deformations of the iris by assuming Gaussian distribution of all sample points along the contour. The gray level information is gathered by sampling a discrete set of points along the normals of all contour sampling points. Both components are joined and marginalized to produce a test of the hypothesis that there is a true contour present.

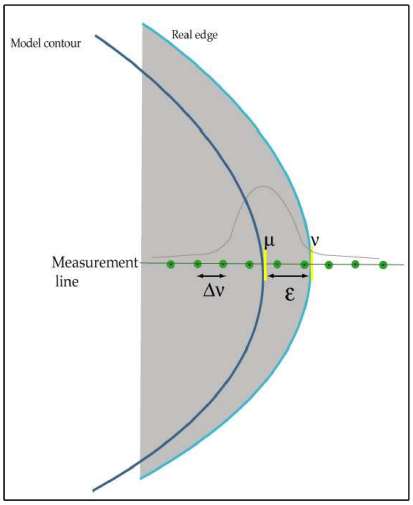


Figure 14.1: Marginalized contour definitions[38]. Given the position of the true boundary μ, the distance from μ to the estimated boundary ν is $\epsilon = \mu - \nu$. Since the geometric object model cannot be assumed to be ideal even though the true position is known, it is assumed that the model is subject to random Gaussian deformation at each sample point on the contours.

We define a *measurement line* $\mathcal{M}$ as a normal to a given point on the contour as seen in figure 14.1. ν is a coordinate on the measurement line, and $\eta(\nu)$ is a binary indicator with value 1 if the boundary of the target is in the interval $[\nu - \Delta\nu/2, \nu + \Delta\nu/2]$, otherwise 0. Given the position of boundary μ, the distance from μ to ν is $\epsilon = \mu - \nu$.

14.4.1 Statistics of Gray-Level Differences

Gray-level values of neighboring pixels are correlated, except when two pixels are known to lie on each side of an object boundary as depicted in figure 14.2.

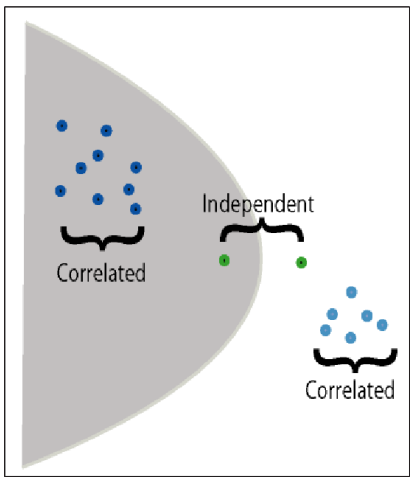


Figure 14.2: Figure from [39]. Gray-level values of neighboring pixels are correlated, except when two pixels are known to lie on each side of an object boundary.

Research on the statistics of natural images has in fact revealed that the pdf of gray-level differences between neighboring pixels is well approximated by a generalized Laplacian[65][40],

$$p_L(\Delta\mathcal{M}) = \frac{1}{Z_L}\exp\left(-\left|\frac{\Delta\mathcal{M}}{L}\right|^{\beta}\right), \tag{14.5}$$

where $\Delta\mathcal{M}$ is the gray level difference, L depends on the distance between two sampled image locations, β is a parameter approximately equal to 0.5 and Z_L is a normalization constant. For $\beta = 0.5$ it can be shown that $Z_L = 4L$[65].

14.4.2 Distributions on Measurement Lines

Given no known edge (object boundary) between two image locations on the measurement line, the pdf of gray levels is by (14.5),

$$p\left[\Delta\mathcal{M}(\nu)|\eta(\nu) = 0\right] = p_L\left[\Delta\mathcal{M}(\nu)\right] \tag{14.6}$$

Assuming independence between gray level differences, the pdf of the observation $\mathcal{M}$ in the absence of an edge is given by

$$p_a(\mathcal{M}) \equiv \prod_i p_L\left[\Delta\mathcal{M}(i\Delta\nu)\right]. \tag{14.7}$$

Contrary, two points observed on opposite sides of an edge are statistically independent. For simplicity, the conditional pdf of gray level differences, separated by an edge, is assumed to be uniform,

$$p\left[\Delta\mathcal{M}(\nu)|\eta(\nu)=1\right] \approx \frac{1}{m}, \tag{14.8}$$

where m is the number of gray levels. If there is a known object boundary at location $j\Delta\nu$, then only one point will correspond to gray level differences across the boundary, the rest will be gray level differences of either object or background. Hence, the pdf of the observation is given by combining (14.7) and (14.8),

$$\begin{aligned} p_c\left(\mathcal{M}|j\Delta\nu\right) &= \frac{1}{m}\prod_{i\neq j} p_L\left[\Delta\mathcal{M}(i\Delta\nu)\right] \\ &= \frac{1}{m}\frac{p_a(\mathcal{M})}{p_L\left(\Delta\mathcal{M}(j\Delta\nu)\right)}. \end{aligned} \tag{14.9}$$

14.4.3 Marginalizing over Deformations

The geometric object model (14.1) cannot be assumed to be ideal, since the idealized contour will never correspond exactly to the object boundary even though the true position is known. Therefore, it is assumed that the model is subject to random Gaussian deformation at each sample point on the contours as seen in figure 14.3.

The prior pdf of deformations is defined

$$p_D(\epsilon) = \frac{1}{Z_D}\exp\left(\frac{-\epsilon^2}{2\sigma^2}\right), \tag{14.10}$$

where $Z_D = \sqrt{2\pi}\sigma$ is a normalization factor.

The likelihood of the observation $\mathcal{M}$ given the contour location μ and the deformation ϵ is

$$p(\mathcal{M}|\mu,\epsilon) = p_c(\mathcal{M}|\mu+\epsilon). \tag{14.11}$$

The joint likelihood of the observation and deformation given the contour, is computed by use of the product rule[55],

$$p(\mathcal{M},\epsilon|\mu) = p_c(\mathcal{M}|\mu+\epsilon)p_D(\epsilon). \tag{14.12}$$

Marginalizing over possible deformations, the likelihood is given by,

$$p_M(\mathcal{M}|\mu) = \int p_c(\mathcal{M}|\mu+\epsilon)p_D(\epsilon)\mathrm{d}\epsilon, \tag{14.13}$$

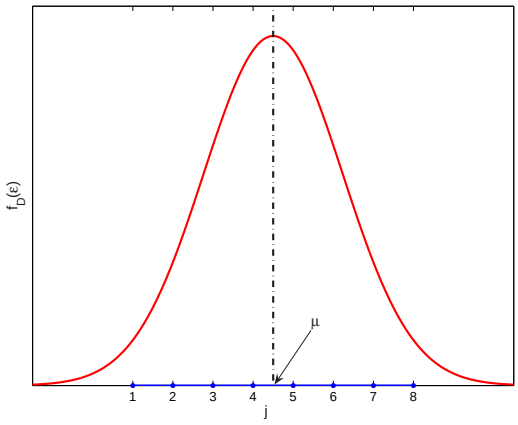


Figure 14.3: The geometric object model cannot be assumed to be ideal. As a consequence, a Gaussian distribution of geometric deformation is assumed. The shape variability on the measurement line is expressed by σ, which should be large enough to reach all the sample points $\Delta\nu$ on the measurement line.

which by combining equation (14.9) and (14.10) yields,

$$p_M(\mathcal{M}|\mu) = \frac{1}{m} p_a(\mathcal{M}) \int \frac{p_D(\epsilon)}{p_L(\Delta\mathcal{M}(\nu))} \mathrm{d}\epsilon. \tag{14.14}$$

The ratio between the likelihoods, regarding the hypothesis that there is a contour at location μ and that there is no contour, can be used for testing the hypothesis of the presence of a contour. The ratio is given by,

$$R(\mathcal{M}|\mu) = \frac{p_M}{p_a} = \frac{1}{m} \int \frac{p_D(\epsilon)}{p_L(\Delta\mathcal{M}(\nu))} \mathrm{d}\epsilon. \tag{14.15}$$

It is suitable to take the logarithm to obtain the log-likelihood ratio,

$$\begin{aligned} h(\mathcal{M}|\mu) &\equiv \log R(\mathcal{M}|\mu) \\ &= -\log(m) + \log \int \frac{p_D(\epsilon)}{p_L(\Delta\mathcal{M}(\nu))} \mathrm{d}\epsilon. \end{aligned} \tag{14.16}$$

The integral can be approximated as a finite sum over discrete set of possible deformations $\epsilon_j = j\Delta\nu - \mu$,

$$h(\mathcal{M}|\mu) = -\log(m) + \log \sum_j \frac{p_D(\epsilon_j)}{p_L(\Delta\mathcal{M}(j\Delta\nu))} \Delta\nu. \tag{14.17}$$

Using the definitions of the generalized Laplacian p_L (14.5), density of deformations p_D (14.10) and choosing the parameter $\beta = 0.5$, the point evaluation

function above becomes,

$$\begin{aligned} h(\mathcal{M}|\mu) &= -\log(m) + \log \sum_j \frac{\frac{1}{Z_D}\exp\left(\frac{-\epsilon_j^2}{2\sigma^2}\right)}{\frac{1}{Z_L}\exp\left(-\sqrt{\frac{|\Delta\mathcal{M}(j\Delta\nu)|}{L}}\right)}\Delta\nu \\ &= h_0 + \log \sum_j \exp\left[\sqrt{\frac{|\Delta\mathcal{M}(j\Delta\nu)|}{L}} - \frac{\epsilon_j^2}{2\sigma^2}\right], \end{aligned} \tag{14.18}$$

where $h_0 = \log Z_L/m - \log Z_D/\Delta\nu$. The full derivation of (14.18) is found in appendix B.2. For a given observation $\mathcal{M}$, the point evaluation increases when a contour is placed at a location that maximizes the function of the absolute values $|\Delta\mathcal{M}|$ under a Gaussian window centered at μ. Thus, the contour of the iris is chosen to maximize the point evaluation function.

14.5 Probabilistic Contour Tracking

The probabilistic formulation has the attraction that uncertainty is handled in a systematic fashion - Increased uncertainty results the particles to be drawn from a wider distribution, while increased confidence results the particles to be drawn from a narrower distribution.

The prediction stage involves using the system model (14.2) to obtain the prior pdf of the state at time $t+1$,

$$p(\mathbf{x}_{t+1}|\mathcal{M}_t) = \int p(\mathbf{x}_{t+1}|\mathbf{x}_t)p(\mathbf{x}_t|\mathcal{M}_t)\mathrm{d}\mathbf{x}_t \tag{14.19}$$

The observation $\mathcal{M}_t$ is independent of the previous state $\mathbf{x}_{t-1}$ and previous observation $\mathcal{M}_{t-1}$ given the current state $\mathbf{x}_t$. At time step $t+1$ a measurement $\mathcal{M}_{t+1}$ becomes available. This is used to update the prior via Bayes' rule,

$$p(\mathbf{x}_{t+1}|\mathcal{M}_{t+1}) \propto p(\mathcal{M}_{t+1}|\mathbf{x}_t)p(\mathbf{x}_{t+1}|\mathcal{M}_t). \tag{14.20}$$

With this in mind, the tracking problem is stated as a Bayesian inference problem by use of (14.19) and (14.20).

Particle filtering is used with the purpose to estimate the filtering distribution $p(\mathbf{x}_t|\mathcal{M}_t)$ recursively. This is done through a random weighted sample set $\mathcal{S}_t^N = \{(\mathbf{x}_t^n, \pi_t^n)\}$, where n is the n^{th} sample of a state at time t weighted by π_t^n. The samples are drawn from the prediction prior $p(\mathbf{x}_{t+1}|\mathcal{M}_t)$, while the sample weights are proportional to the observation likelihood $p(\mathcal{M}_t|\mathbf{x}_t)$ given by the sum of contour hypotheses $\sum_{k=1}^n h(\mathcal{M}_k|\mu)$. The sample set

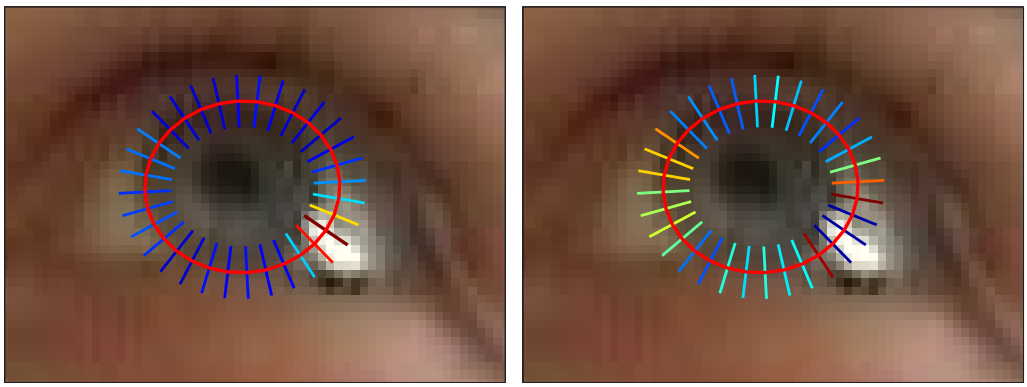


Figure 14.4: The relative normalized weighting of the hypotheses regarding one particle are colored in a temperature scale - Blue indicates low, while red high scores. (*Left*) Corneal reflections cause very distinct edges. Thus some hypotheses are weighted unreasonable high, which may confuse the algorithm. (*Right*) By use of robust statistics outliers are rejected. This results in a better and more robust estimate of the hypotheses regarding the contour.

propagates into a new sample set $\mathcal{S}_{t+1}^N$, which represents the posterior pdf $p(\mathbf{x}_{t+1}|\mathcal{M}_{t+1})$ at time $t+1$. A more detailed explanation of particle filtering is found in appendix B.1.1.

14.6 Constraining the Hypotheses

Despite the fact that a large amount of particles wastes computation time, the proposed active contour method may fail due to different non-ideal conditions in the image.

Corneal reflections, caused by illumination, may confuse the algorithm to weigh some of the hypotheses unreasonably high compared to others. This issue is illustrated left in figure 14.4, where the relative normalized weighting is colored in a temperature scale - Blue indicates low, while red high scores. By using robust statistics, these hypotheses are treated as outliers and therefore rejected.

The contour algorithm may fit to the sclera rather than the iris. This is due to the general formulation of absolute gray level differences $\Delta\mathcal{M}$[16], which seeks to detect contours in a general sense. An example is depicted in figure 14.5, where the image evidence of the contour surrounding the sclera is greater than the one around the iris. It turns out that for a large number of particles, the maximum likelihood estimate prefers the contour around the white sclera when the gaze is turned towards the sides.

As a consequence, we propose to constrain the hypotheses. Intuitively,

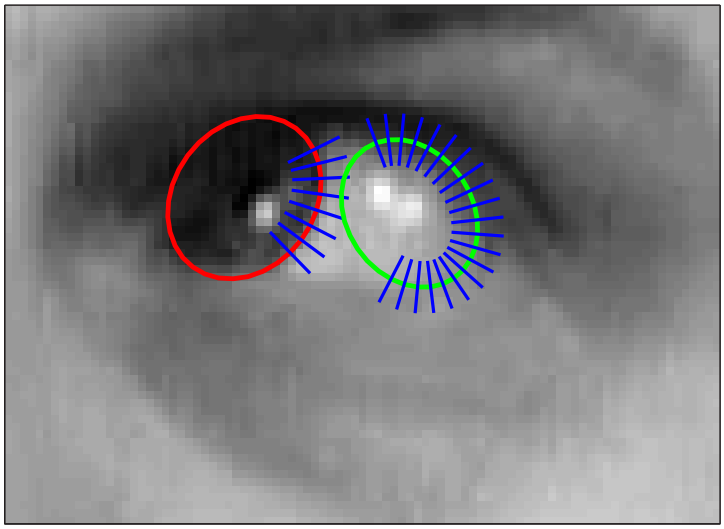


Figure 14.5: This image is depicted as a gray level intensity image, to demonstrate the problem of low gray level difference between iris and the eyelids. Significant hypotheses are shown as blue lines. Due to the general formulation of absolute gray level differences $\Delta\mathcal{M}$, the right contour has a greater likelihood, and the algorithm may thus fit to the sclera.

the average intensity value of the inner ellipse could be compared to some defined outer region as seen in expression (13.12). Nevertheless, this is a weak constraint as discussed in section 13.4 and depicted in figure 13.8. The robustness of the active contour algorithm is increased by weighing the belief of hypotheses and utilizing robust statistics to reject outliers.

We propose to weigh the hypotheses through a sigmoid function, applied on the measurement line $\mathcal{M}$, defined as,

$$\mathcal{W} = \left(1 + \exp\left(\frac{\mu_i - \mu_o}{\sigma_w}\right)\right)^{-1} \tag{14.21}$$

where σ_w adjust the slope of weighting function, μ_i and μ_o are the mean values of the inner and outer sides of the contour respectively. The function is exemplified in figure 14.6. This has the effect of decreasing the evidence when the inner part of the ellipse is brighter than the surroundings. In addition, this relaxes the importance of the hypotheses along the contour around the eyelids, which improves the fit.

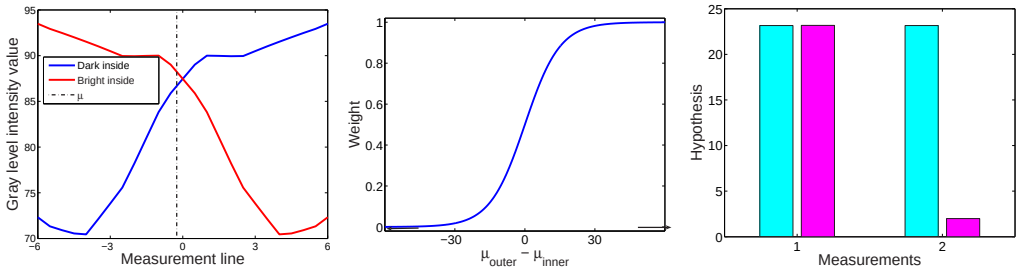


Figure 14.6: (*Left*) The two lines depicts the gray level intensity of two measurement lines - The blue one where the inner part of the ellipse is dark, and the red in the reverse case. (*Middle*) The shifted hyperbolic tangents is utilized as weighting function. Note, the limit values are in range $[-255; 255]$. (*Right*) The cyan bars indicates the hypothesis value before weighting, while the pink is after. Measurement *1* - The blue line - is nearly unchanged, while *2* - the red line - is suppressed.

14.7 Maximum a Posteriori Formulation

The dynamic model may, in certain outlier cases, grow or shrink the contour to a degree, from where the algorithm gets lost. As a consequence, we propose to constrain on the shape of the ellipse in analogy to section 13.4.2. The parameters defining an ellipse is assumed normal distributed with mean μ and covariance $\mathbf{\Sigma}$. The prior distribution of these parameters are then defined,

$$p(\mathbf{x}) = \mathcal{N}(\mu, \mathbf{\Sigma}) \propto \exp\left(-\frac{1}{2}(\mathbf{x}-\mu)^T\mathbf{\Sigma}^{-1}(\mathbf{x}-\mu)\right), \tag{14.22}$$

where the normalization factor has been omitted. The mean and covariance are estimated in a training sequence.

Combining the priors - presented in this section - with the likelihood, results in the *Maximum a Posteriori* formulation (MAP), where the goal is to maximize,

$$p(\mathbf{x}|\mathcal{M}) \propto p(\mathcal{M}|\mathbf{x})p(\mathbf{x}). \tag{14.23}$$

By incorporation of prior knowledge about the shape, with the prediction prior and observation likelihood (14.20), the robustness increases considerably and the parameters are constrained to avoid the algorithm to break down due to infinite increase or decrease of parameters.

14.8 Optimization by EM

Increasing the number of particles leads to better performance, but increases the number of computations as well. The chosen number should therefore be a trade-off between accuracy and computation time.

The number of particles can be reduced by applying the *expectation-maximization* algorithm to a subset of samples. Since the log-likelihood is obtained by marginalization, optimization by the EM method leads to more robust convergence.

14.8.1 Motivation

Suppose that the true location ν^* of boundary were known, then the image evidence should be ignored since it would give no additional information. The likelihood of the contour algorithm could be simplified to the likelihood of the deformation (14.10),

$$p(\nu^*|\mu) = p_D(\nu^* - \mu). \tag{14.24}$$

However, the true locations are unfortunately not known and must therefore be estimated. This is obtained by applying the EM algorithm, which is described in appendix B.3.

14.8.2 Applying EM

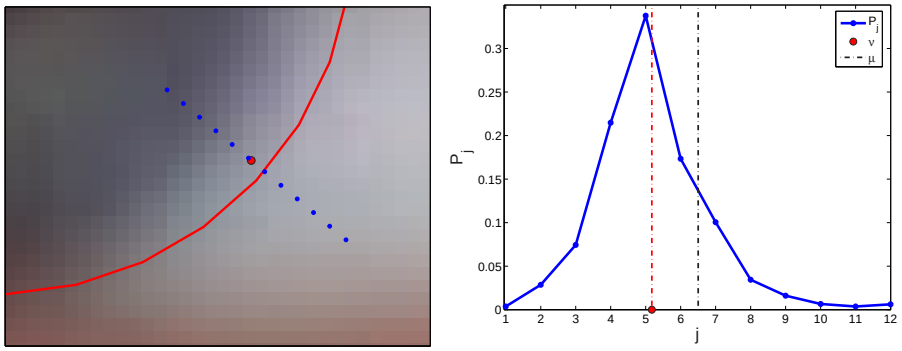


Figure 14.7: The true location of iris is estimated in the E-step by use of *center of mass* of the observations. p_j is the probability that the contour on the respective measurement line is in the interval $[(j - \frac{1}{2})\,\Delta\nu, (j + \frac{1}{2})\,\Delta\nu]$. The new estimate $\hat{\nu}$, depicted in (*right*), is seen in (*left*) to be better than the original μ.

The unobserved true contour location ν^* is not known. It can be estimated by use of *center of mass* of the observation $\widehat{\nu}$, which is also referred as the *Bayes least squares* estimate of the deformation[65] (see figure 14.7),

$$\widehat{\nu} \equiv \sum_j p_j j \Delta\nu, \tag{14.25}$$

where p_j is the probability that the contour on the respective measurement line is in the interval $[\left(j-\frac{1}{2}\right)\Delta\nu, \left(j+\frac{1}{2}\right)\Delta\nu]$. This is obtained by normalization to unity of the sum over all locations on the given measurement line and is defined,

$$p_j \equiv p(\epsilon_j|I,\mu)\Delta\nu = \frac{p(I,\epsilon_j|\mu)}{\sum_i p(I,\epsilon_i|\mu)}. \tag{14.26}$$

By use of (14.12) consisting of (14.9) and (14.10), the above expression is rewritten,

$$\begin{aligned} p_j &= \frac{p_D(\epsilon_j)\frac{1}{m}p_a(\mathcal{M})/p_L\left[\Delta\mathcal{M}(j\Delta\nu)\right]}{\sum_i p_D(\epsilon_i)\frac{1}{m}p_a(\mathcal{M})/p_L\left[\Delta\mathcal{M}(i\Delta\nu)\right]} \\ &= \frac{p_D(\epsilon_j)p_L^{-1}\left[\Delta\mathcal{M}(j\Delta\nu)\right]}{\sum_i p_D(\epsilon_i)p_L^{-1}\left[\Delta\mathcal{M}(i\Delta\nu)\right]}. \end{aligned} \tag{14.27}$$

The improved estimate of the contour location can now be computed, and the log-likehood of the deformation is given by,

$$\begin{aligned} \log p(\widehat{\nu}|\mu) &= \log p_D(\widehat{\nu}-\mu) \\ &= -\log Z_D - g(\widehat{\nu}-\mu), \end{aligned} \tag{14.28}$$

where

$$g(\widehat{\nu}-\mu) = \frac{(\widehat{\nu}-\mu)^2}{2\sigma^2}. \tag{14.29}$$

Estimating the contour location by center of mass (14.25), incorporates the current estimate through (14.27). As a consequence, an estimate of μ is required to obtain $\widehat{\nu}$, and an estimate of $\widehat{\nu}$ is required in order to optimize μ. Thus, the iterative scheme from section B.3 can be interpreted as,

EM Optimization step

E step: Estimate the true contour location as the *center of mass* of the observation $\widehat{\nu}$, using the image evidence $\Delta\mathcal{M}$ and last estimate of location μ^{k-1} from $\mathbf{x}^{k-1}$.

M step: Optimize the state parameters $\mathbf{x}$ by minimizing the squared deformation $g(\widehat{\nu}-\mu)$ to obtain the re-estimated contour location μ^k.

The presented active contour algorithm optimized by EM, is depicted in figure 14.8.

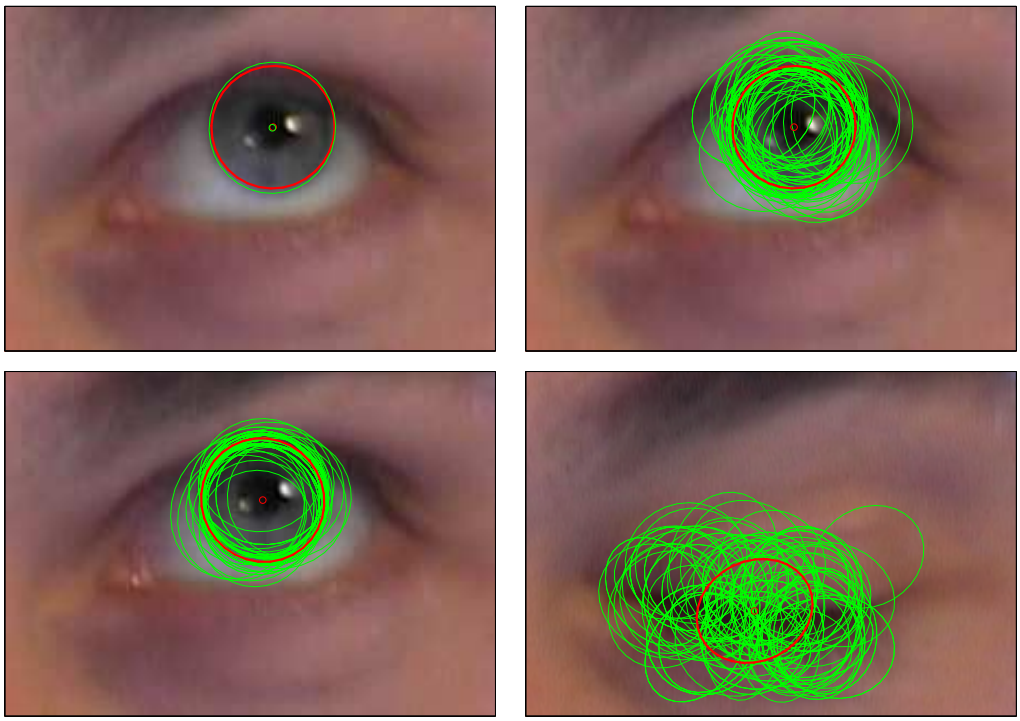


Figure 14.8: The current state is found by taking the sample mean of the posterior probability, which is estimated recursively; in this case only 50 particles are drawn from $\mathbf{x}_k = \mathbf{f}_k(\mathbf{x}_{k-1}, \mathbf{v}_{k-1})$. (*Top left*) The estimated state $\mathbf{x}_k$ at iteration k is depicted as the green curve, while the red is the optimized EM contour. (*Top right*) The green curves illustrates 50 drawn particles at iteration k, the red curve represents the EM contour. (*Bottom left*) The particles are drawn from a narrower distribution, since the location of iris has been nearly static for some frames. (*Bottom right*) Even though the eye is closed, the algorithm still makes a suitable estimate. The uncertainty increases, resulting in particles to be drawn from a wider distribution.

14.9 Optimization by Deformable Template Matching

Similar to the *expectation-maximization* algorithm, the deformable template matching model from section 13.4 may be used to refine the state estimate. The precision of the location of pupil center is equivalent to - or even better than - the iris center. Hence, given the state $\mathbf{x}$ that maximizes the posterior distribution (14.20), an appropriate starting point regarding the deformable template matching method can be chosen to,

$$\mathbf{x} = (c_x, c_y, \alpha\lambda_1, \beta\lambda_2, \theta)\,, \tag{14.30}$$

where α and β are scaling parameters. An example of the optimization of the deformable model is seen left in figure 13.9.

14.10 Parameters of the Method

In order to apply the active contour algorithm, a number of parameters regarding the observation and dynamic model must be set.

The observation model expressed through the point evaluation function (14.18) has 3 parameters; the scale parameter L of the pdf of gray level differences, the standard deviation σ of shape deformations and the sampling interval $\Delta\nu$ on the measurement lines.

The parameter L is estimated from the average square root of gray level differences measured over a representative subset of images $\mathbf{E}\left[\sqrt{\Delta\mathcal{M}}\right]$. By use of the convenient value $\beta = 0.5$, the maximum-likelihood estimate of L is obtained explicit by differentiating (14.5) and finding the root; L_{ML} $=\mathbf{E}\left[\sqrt{\Delta\mathcal{M}}\right]^2/4$. The parameters of the generalized Laplacian distribution L and β are directly related to the variance and kurtosis[40]. The latter measures the peakedness or tail prominence of the distribution in proportion to the Gaussian distribution in which $\beta = 0.0$[16].

The standard deviation parameter σ expresses the shape variability on the measurement line. Thus, σ is defined to reach all coordinates on this line within $\pm 3\sigma$ drawn from a Gaussian distribution as seen in figure 14.3.

The sampling interval $\Delta\nu$ should be scale dependent. A convenient value is given by $\Delta\nu = \max(1, \sigma/4)$.

The parameters of the dynamical model are constrained to a reasonable physical range of the object being tracked. The evolution of the state sequence is given by $\mathbf{x}_k = \mathbf{f}_k(\mathbf{x}_{k-1}, \mathbf{v}_{k-1})$. The initial state $\mathbf{x}_0$ can be set either

manually or by one of the segmentation based methods from chapter 13. The state evolution is then estimated from the samples controlled by the system variance $\mathbf{v}_{k-1}$.

In the optimization step, each frame is run until some stop criterion.

14.11 Summary

The active contour model under consideration does not use features explicitly but maximizes feature values underlying the contour in a Bayes sense. Two basic assumptions are made; there are no correlation between gray-level values on each side of an object boundary, and marginalization over shape variations.

A hypothesis regarding the presence of a contour is formulated as the ratio between the contour existence and non-existence likelihoods. The current state is then found recursively by maximizing the estimated posterior probability.

The model is utilized by particle filtering for iris tracking since the changes in iris position are very fast. Furthermore, the estimate of the state is optimized by the EM algorithm.

Extensions to the original active contour method are proposed to improve robustness and accuracy:

- Weighing of the hypotheses to relax their importance along the contour around the eyelids. Moreover, it penalizes contours surrounding bright objects.
- Robust statistics to remove outlying hypotheses stemming from corneal reflections.
- Constraining the deformation of the contour regarding the magnitude of the axes defining the ellipse.
- Refinement of the fit by a deformable template model of the pupil.

The pseudo code of the EM Active Contour method is presented in appendix B.4. The active contours with the deformable template refinement method is similar - The optimization step is simply replaced.

Chapter 15

Gaze Determination

Gaze is very important for human communication and also plays an increasing role for human computer interaction. Gaze can play a role, e.g., in understanding the emotional state for humans[3][4], understanding the perception of infants[27], synthesizing emotions[32], and for estimation of attentional state[82]. Specific applications include devices for the disabled, e.g., using gaze as a replacement for a computer mouse and driver awareness monitoring to improve traffic safety[43].

We use a geometric head model for gaze determination[43]. There is nothing novel about this model. It is simply a translation from 2D image coordinates to a direction in space relative to the first frame. Thus, suppose the anatomical constants are measured somehow, and the pose and scale of face, eye corners, and pupil location are known, then the exact gaze direction can be computed.

However, this is not the case. The method for gaze estimation is described below.

15.1 The Geometric Model

Some basic assumptions are made; the eyeball is spherical and the eye corners have been estimated. The latter is not a trivial task. In fact it is more difficult to detect eye corners than the iris or pupil. This task can be solved by use of AAM.

We begin by defining some anatomical constants of the eye as depicted in figure 15.1b.

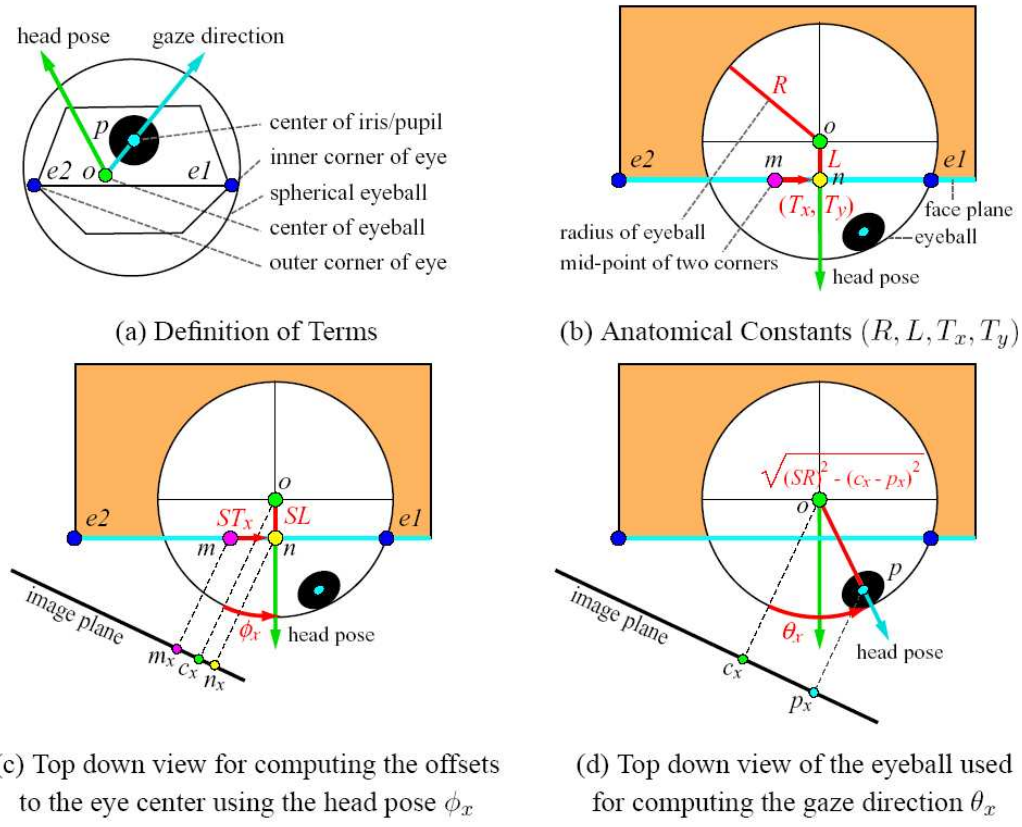
Anatomical Constants

$R0$: Radius of the eyeball when the scale of the eye is one.

(T_x, T_y): The offset in the image between the mid-point of the two eye corners and the center of eyeball, when the face is frontal and the scale is one.

L: The depth of the center of the eyeball relative to the plane containing the eye corners.

The anatomical constants are pre-computed on a training sequence by use of the least squares solution[43].



(a) Definition of Terms

(b) Anatomical Constants (R, L, T_x, T_y)

(c) Top down view for computing the offsets to the eye center using the head pose ϕ_x

(d) Top down view of the eyeball used for computing the gaze direction θ_x

Figure 15.1: Geometric model for gaze estimation[43].

In order to estimate the gaze, we need to compute the center and radius of the eyeball.

The mid-point (m_x, m_y) between the inner corner $(e1_x, e1_y)$ and the outer corner $(e2_x, e2_y)$ are estimated by,

$$\begin{pmatrix} m_x \\ m_y \end{pmatrix} = \begin{pmatrix} \frac{e1_x + e2_x}{2} \\ \frac{e1_y + e2_y}{2} \end{pmatrix}. \tag{15.1}$$

The scale of the face is estimated by the AAM. A more simple approach, is to compute the distance between the eye relative to the head pose angle ϕ_x,

$$S = \frac{\sqrt{(e1_x - e2_x)^2 + (e1_y - e2_y)^2}}{\cos\phi_x}. \tag{15.2}$$

The disadvantage is, however, that numerical errors are introduced when two points subtracted are very close.

The center of the eyeball is determined as the mid-point corrected by two terms; Even though the face is frontal, the midpoint cannot be assumed equivalent to the eye center - which cannot be assumed to lie in the plane of the eye corners (see figure 15.1c),

$$\begin{pmatrix} o_x \\ o_y \end{pmatrix} = \begin{pmatrix} m_x \\ m_y \end{pmatrix} + S \begin{pmatrix} T_x \cos\phi_x \\ T_y \cos\phi_y \end{pmatrix} + SL \begin{pmatrix} \sin\phi_x \\ \sin\phi_y \end{pmatrix}. \tag{15.3}$$

The radius of the eyeball is estimated from the scale and anatomical constant, $R = SR_0$.

At last, the gaze direction (ϕ_x, ϕ_y) can be computed as,

$$\begin{pmatrix} \sin\theta_x \\ \sin\theta_y \end{pmatrix} = \begin{pmatrix} \frac{p_x - o_x}{\sqrt{R^2 - (p_y - o_y)^2}} \\ \frac{p_y - o_y}{\sqrt{R^2 - (p_x - o_x)^2}} \end{pmatrix}. \tag{15.4}$$

An example where the face is frontal is depicted in figure 15.2.

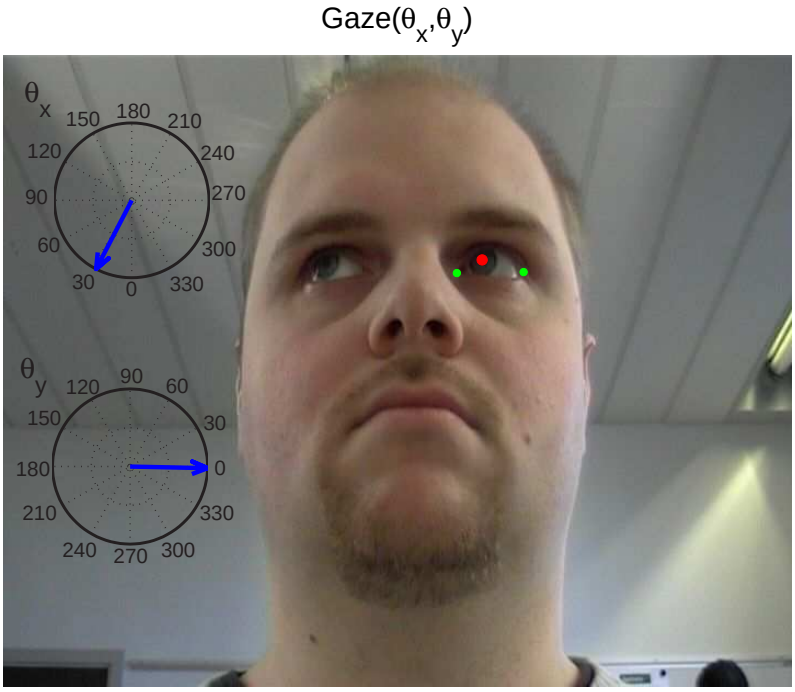


Figure 15.2: The gaze direction (θ_x, θ_y) is computed based on (15.4). The eye corners is obtained from the AAM and depicted in green. The estimate of the center of pupil is obtained from one of the methods described in this part.

Chapter 16

Discussion

Several approaches for eye tracking has been presented in chapter 13 and 14. The main difference is the propagation model - That is, how the system dynamics are propagated given previous state estimates. While the segmentation based tracking uses the last estimate as starting point for a segmentation method, or even no knowledge of old states at all, the bayesian tracker predicts the state distribution given previous state.

16.1 Segmentation-Based Tracking

The different segmentation-based eye trackers proposed, all have their strengthes and weaknesses. The double threshold method is very fast, even though it does not use knowledge from previous estimates. Since corneal reflections may overlap the pupil to some extent, the estimate is expected to be biased. On the other hand, the method may be used to get a rough estimate very fast. This can be applied with advantage as starting point for better, but computational more expensive, tracking algorithms.

The template matching utilizes the knowledge of the appearance of iris. No memory regarding previous states is used. As a consequence, the method is computational expensive in proportion to double threshold, but more precise. However, in the lack of sharp edges, the robustness decreases. Moreover, the method tends to fail during blinking. The template cannot handle deformations.

This leads to the deformable template matching method, which in contrast to the above methods, avoids using explicit features nor fixed templates. Instead it maximizes the difference between an inner and outer template. Thus, the only requirement is a suitable starting point and that the pupil is darker than iris. Combining the method with priors regarding the shape,

and robust statistics in cases where the illumination causes dominant corneal reflections, results in an excellent template model. However, rapid eye movements may confuse the tracker since the starting point is too far from the true state. For that reason, we propose to use double threshold in order to choose an appropriate starting point. If several starting points are found, the state corresponding to the optimal image confidence is preferred.

16.2 Bayesian Eye Tracking

The active contour model is utilized by particle filtering for iris tracking since the changes in iris position are very fast. The method is based on two basic assumptions; there are no correlation between gray-level values on each side of an object boundary, and marginalization over shape variations.

Furthermore, the estimate of iris location is optimized by the EM algorithm and the deformable template model. The disadvantage of EM is the slow convergence of the optimization since the contour parameters are not gradient-based. Furthermore, the number of calculations to obtain one single observation is huge, leading to demanding computational time. In contrast, the deformable model is optimized independent of the measurements lines, but utilizes the image evidence in a Newton optimization.

The power of bayesian tracking is the capability to handle typical image processing problems such as moderately variations of scale, occlusions and changes in illumination. Variations of scale may occur due to the relative movement between camera and eye. In the case of partial occlusions, the object pose is well estimated depending on the geometry of the observable lines. Changes in illumination is handled by re-estimation of λ from the average square root of gray level differences measured in the current image.

The contour model under consideration avoids explicit feature detection by maximizing feature values underlying the contour in a Bayes sense. This makes the method fairly robust to noise. Moreover, the likelihood is smoothed by marginalization over possible shape variations.

In general, highly textured regions may limit the performance of the contour algorithm. This is, however, not the case with the available data.

16.3 Comparison

The main difference between the trackers described in this part is how the system dynamics are propagated given previous state estimates. Previous estimate may be used as starting point for segmentation-based trackers with

the exception of rapid eye movements. In this case, the algorithm gets stuck and it may take several frames to find the way back; in worst case the algorithm may break down.

The strength of particle filtering is that increased uncertainty results in particles to be drawn from a wider distribution. On the other hand, increasing the belief results in particles to be drawn from a narrower distribution. For that reason the active contour method is quite robust. The cost is, however, a computational demanding algorithm compared to the deformable model.

Since the deformable model seeks to locate the pupil contour, instead of iris, fewer irregularities are expected as long as the eyes are open. Applying priors regarding the shape ensures the algorithm not to break down when the eyes are closing. By use of double threshold as starting point ensures the method to be capable of handling rapid movements.

Combining the active contour utilized by particle filtering and the deformable model to refine the estimate, results in a precise and robust tracker.

Part III

Experimental Results

Chapter 17

Experimental Design

In this part we evaluate the proposed methods of the eye tracking system.

The system presented is run on a 2.4GHz Pentium 4. We have implemented all methods described in the parts above in Matlab 7.0. Computationally demanding subroutines are implemented in C++.

17.1 System Overview

The system design is broken down into five components; The digital video camcorder, a computer interface, face detection and tracking, eye tracking, and gaze determination. Each component is described briefly below.

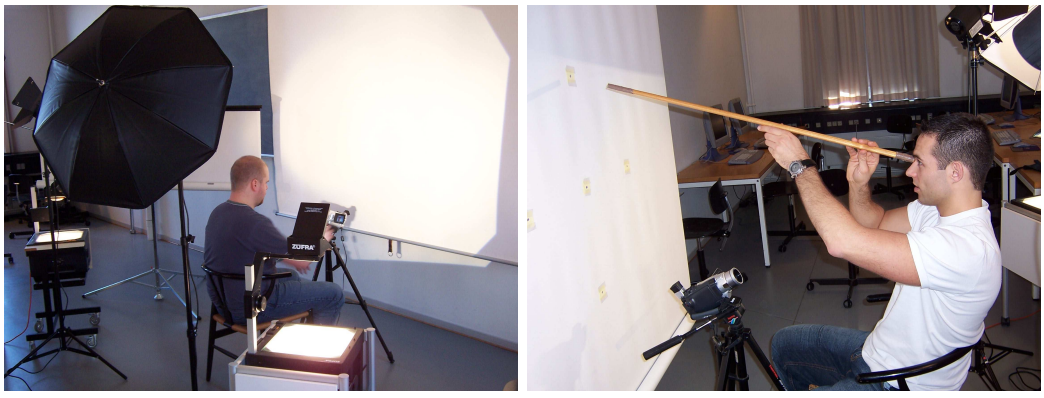


Figure 17.1: Video recording setup. (*Left*) The person recorded is indirectly illuminated. (*Right*) The gaze direction is turned at a collection of markers placed at some controlled points in 3D space. The DV-Cam is positioned in front of the person.

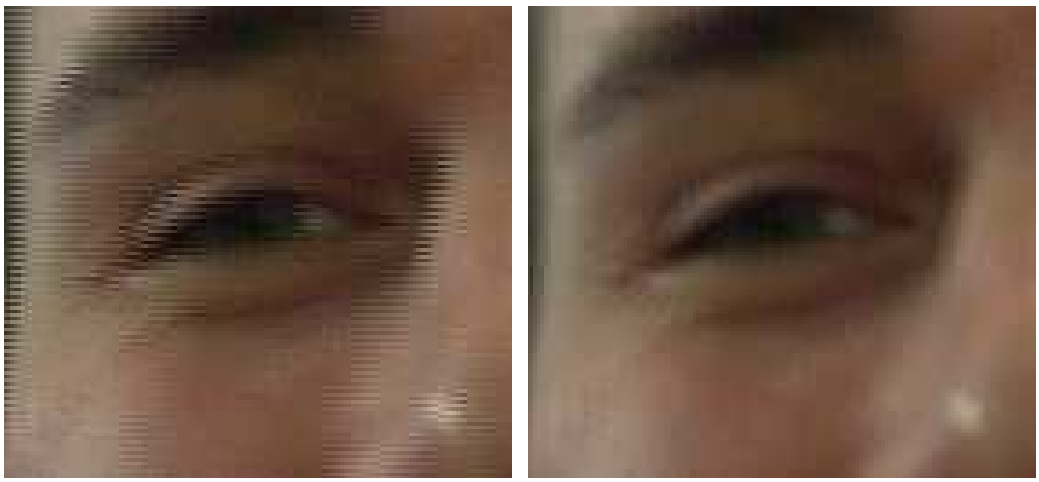


Figure 17.2: (*Left*) The frames extracted from videos are interlaced. Every two consecutive images are intermixed with the purpose to minimize flickering. (*Right*) The corresponding deinterlaced image.

17.1.1 Camera

The DV-Cam records video sequences of the face. All variations, in the yaw, pitch and roll angles are within ±30 degrees relative to frontal images. The setup is shown in figure 17.1.

17.1.2 Computer Interface

This component contains a framegrabber, which extracts frames from video sequences, and a deinterlacer. A DV-Cam does not, as intuitively expected, record 25 frames per second (fps) when recording a movie. Instead it records 50 pictures per second, intermixing every two consecutive images, with half the height, into one frame[1]. The purpose of interlacing is to minimize flicker inherent in lower-bandwidth images. An example of a interlaced and the corresponding deinterlaced is depicted in figure 17.2. This responsibility of this component is fully handled by an the Open Source software framegrabber *VirtualDub*.

17.1.3 Face Detection and Tracking

The active appearance model is utilized for face detection and tracking given the video frame sequence. This module outputs the location of the eye region and provides the pose of the head.

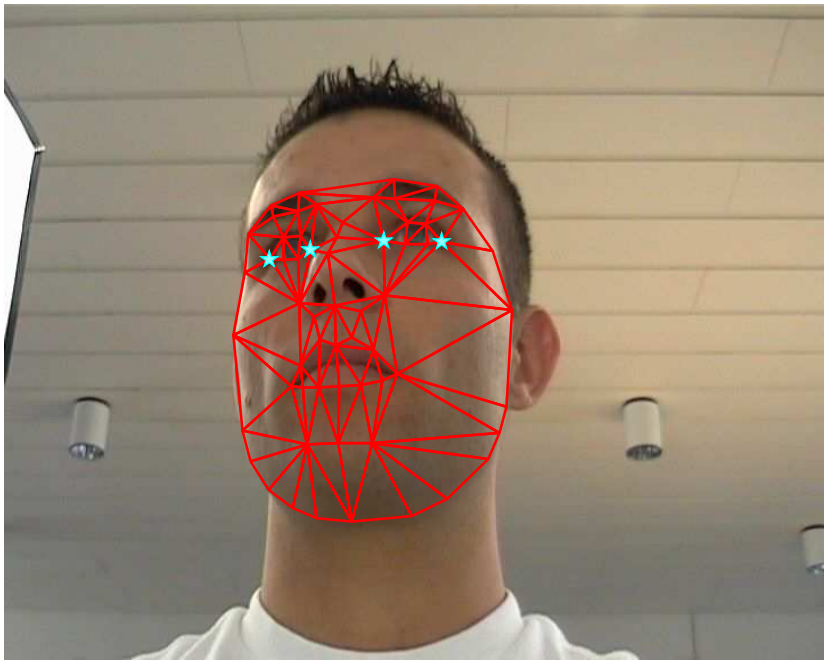


Figure 17.3: The eye images are extracted from the input video frames based on the fit of AAM. Each eye is modeled by six vertices. A bounding box containing the eye is extracted by choosing a region slightly larger than the modeled eye.

17.1.4 Eye Tracking

The eye images are extracted from the video frames based on input from AAM. The resulting fit of one frame is shown in figure 17.3. Each eye region is spanned by six vertices. A bounding box containing the eye is extracted on which the eye tracking methods are applied.

17.1.5 Gaze Determination

The gaze direction can be determined by combining the anatomical constants with the pose and scale of face, eye corners, and pupil location.

17.2 System

Each module functions as an independent unit. Thus, the eye tracking algorithm can be exchanged with no consequence for the face tracker, and vice versa. As a consequence, the face tracking and eye tracking modules are tested independently.

17.3 Data

The algorithms are tested on various datasets designed to illuminate different problems regarding eye tracking. The datasets are presented in the following chapters.

17.4 Algorithm Evaluation

To assess performance of the developed algorithms, the error is recorded as the difference between an annotated ground truth and the output of the algorithms. Specifically, we use the *point to point error* defined as the Euclidian distance between corresponding landmarks regarding the AAM, or between the pupil location and ground truth regarding eye tracking,

$$D_{pt.pt.}(\mathbf{x}_{gt}, \mathbf{x}) = \frac{1}{n} \sum_{i=1}^{n} \sqrt{(x_i - x_{gt,i})^2 + (y_i - y_{gt,i})^2}. \tag{17.1}$$

17.5 Overview

A variety of experiments are presented in the following. The results are found in chapter 18 and 19 for the AAM and eye trackers respectively.

Chapter 18

Face Detection and Tracking

In this chapter the ability of the active appearance model to detect and track faces in sequences of images is tested. As stated in 17.1, the tasks of the AAM in the eye tracking system are; to provide the pose of the face, and to find the region of the eyes. A bounding box given by the eye corners is extracted and passed on to the eye tracking algorithms presented in part II together with the estimated pose. All this, requires the AAM to accurately find and track the face.

18.1 Constructing the AAMs

Before proceeding, a set of active appearance models must be created. Models of three different datasets have be made:

Dataset 1 : A person specific AAM constructed from a image sequence of 5000 frames of a talking man[20]. Every tenth frame of the first 1000 frames have been used for training.

Dataset 2 : A person specific AAM constructed from a sequence of 340 frames of one of the authors.

Dataset 3 : A multi-person AAM constructed from 240 images of 40 different human faces[60].

Figure 18.1 depicts examples of all three training set with training set 1 to the top left, 2 to the top right, and 3 at the bottom of the figure.

The number of shape eigenvectors to use is chosen by thresholding on the total variance as described in section 5.2.2. The number of modes explaining 95% of the total variance are retaining for each model. For Dataset 1, this yields 5 component, Dataset 2 consist of 2 modes, and the multi-person

Figure 18.1: Examples of training images. Training set 1 to the top left, 2 to the top right, and 3 at the bottom.

Dataset 3 uses 21 eigenvectors. It is seen that the number of modes for the multi-person dataset is significantly bigger than for the two single-person datasets. Since Dataset 3 must model head rotation and a lot of inter-person differences in size and shape of the head, a large number of components is needed.

In the following sections the capability of the AAM as an image segmentation and feature extraction method is investigated.

18.2 Convergence

To measure the convergence of the AAM, model instances are placed in various images. The vertices of the instances should hopefully converge to the ground-truth points corresponding to the image.

Since the AAM fitting algorithm is based on a local optimization method, it has the risk of getting stuck in a non-optimal local minima. Therefore, the quality of the fit is very dependent on the starting point of the search. To get a 'fair' measurement, a number of searches are conducted from different starting points. The starting points are distributed as circles with expanding radius. The circles center lies in the center of gravity of the ground truth points. Figure 18.3 depicts the starting points. The points are distributed with radii $[1, 5, 10, \ldots, 35]$. Each circle consists of 25 starting points. An experiment is conducted for each starting point, and the point-to-point error, see (17.1), is recorded at each iteration of the AAM search algorithm.

To illuminate any differences between single person and multi person AAMs, the experiment is made on five images from the single-person AAM, Dataset 2, and five from the multi-person Dataset 3, not used in the model training. A test image from each of the test sets is depicted in figure 18.2.

The searches are performed by translation of the mean shape of the model to the starting point, and commencing the search.

Figure 18.4 depicts the iterations of an AAM search, starting at the top left image. Notice that the top three images mostly consist of translation of the mean shape, corresponding to optimization of the global shape transformation described in 9.4. In the bottom three images, the optimization is mostly in the deformation of the mean shape.

18.2.1 The Average Frequency of Convergence

To investigate the fitting performance of the AAM, a measure is introduced; the *average frequency of convergence*. We count the number of times for each circle the algorithm converges. Figure 18.5 depicts the average frequency of

Figure 18.2: Two of the test images used in the convergence test. The left image corresponds to Dataset 2, the right to Dataset 3.

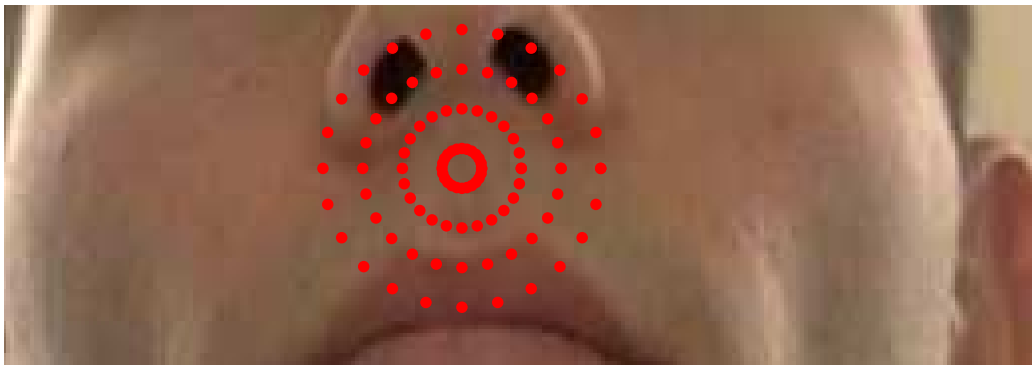


Figure 18.3: Convergence Test starting points. The red dots indicate the starting location of the AAM instance

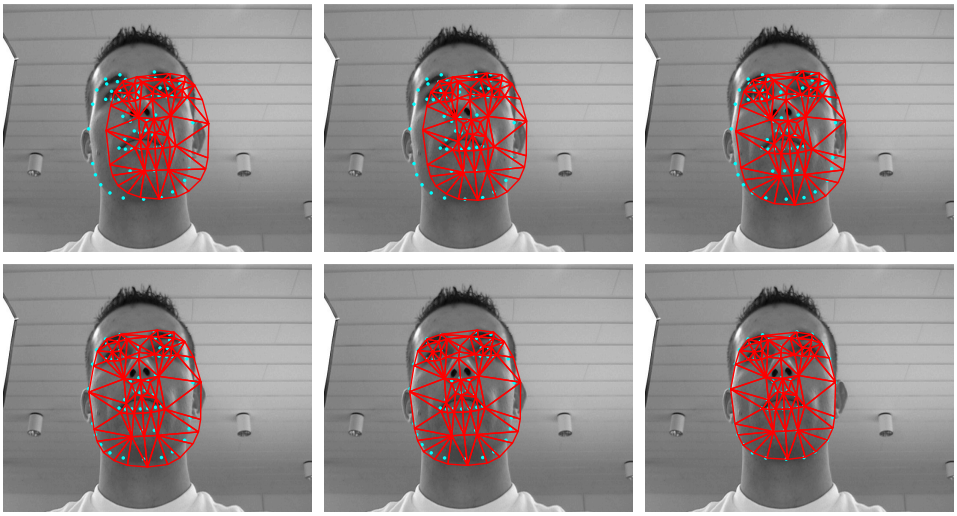


Figure 18.4: Iterations of an AAM search The start of the search, depicted in the top three images, mostly consist of translation of the mean shape, corresponding to optimization of the global shape transformation. In the bottom images the shape is deformed to fine tune the fit of the face. mean shape.

convergence for the single-person AAM of Dataset 2 and the multi-person AAM of Dataset 3.

The blue curve represents the single-person AAM, and as seen all searches up until a circle radius of 20 pixels converge to the ground truth. Then it slowly decays to roughly 50% convergence at a radius of 35 pixels.

The multi-person AAM, represented by the red curve, is seen to perform very poorly. The frequency of convergence decays very fast with the radii of the circles and at a radius of 30 pixel it does not converge at all.

One source for the poor performance might be traced to the fact, that the inverse compositional AAM algorithm minimizes an error function, based on the error between the mean texture and the image. See (8.31). If this mean texture is calculated in a single-person AAM, the mean texture is a good approximation to the face of the person. However, in multi-person AAMs, the mean texture might be far from the face in the image. Figures 18.7 and 18.8 shows four surface plots of the error surface. The surfaces are made by varying the parameters corresponding to the two first shape eigenvectors, b_{s_1} and b_{s_2}, while zeroing the rest. The mean shape is translated to the center of gravity of the ground truth, and placed in the corresponding test images. Then the mean shape is warped according to the values of b_{s_1} and b_{s_2}, see figure 18.6 for an example. The error function value, (8.31), is calculated and used as 'z'-value in the surface plot.

The plots depicts the value of the error function as a function of the

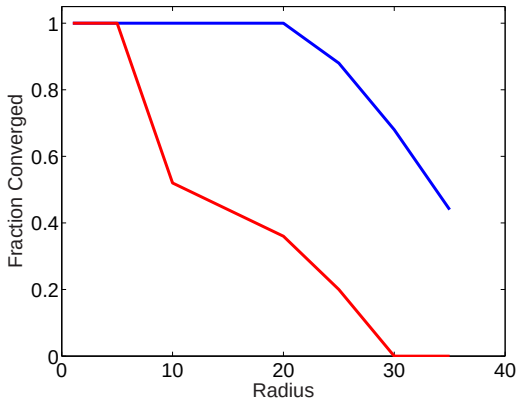


Figure 18.5: Convergence test. The fraction of converged AAM searches. The blue curve indicate the single-person dataset, and the red curve the multi-person.

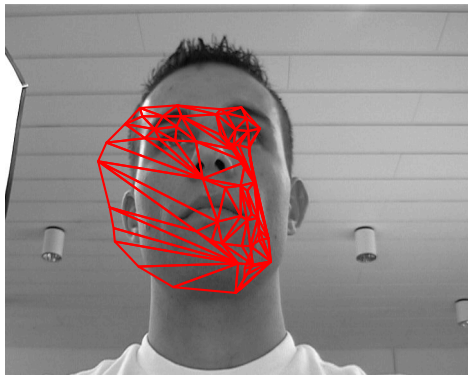


Figure 18.6: A shape corresponding b_{s_1} and b_{s_2} equal to -3 standard deviations used in the calculation of the error surfaces.

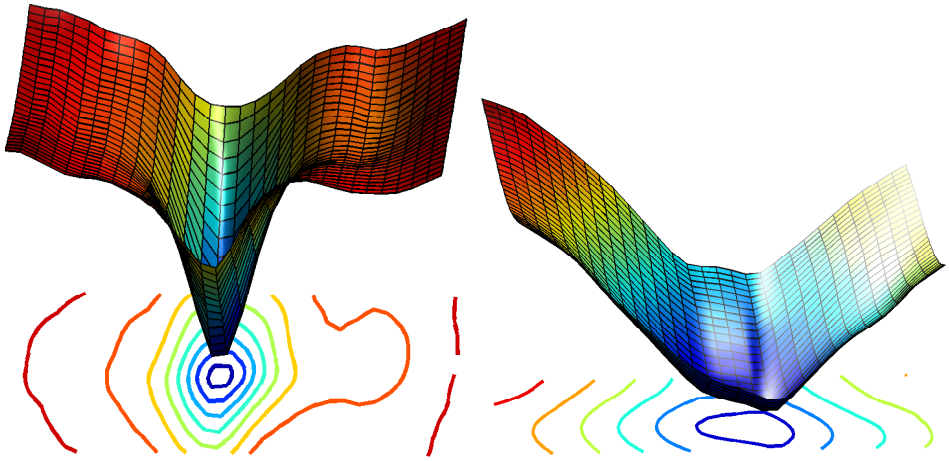


Figure 18.7: Error surfaces for (8.31) depicting the value of the error function as a function of two parameters. They are made by varying the parameters corresponding to the two first shape eigenvectors, b_{s_1} and b_{s_2}, while zeroing the rest. The mean shape is translated to the center of gravity of the ground truth corresponding to the test image.. The parameters are varied between ± 3 standard deviations of the parameters. The left surface correspond to the AAM built from Dataset 2, and the right correspond to Dataset 3.

parameters. The surfaces of figure 18.7, corresponds to variations of the parameters between ± 3 standard deviations. It is seen that the error function is very well suited for gradient descent optimization. Both contain no obvious local minima that might confuse the optimizer. The surface plots of figure 18.8, corresponds to variations of the parameters between ± 7 standard deviations. It is seen that the error surfaces corresponding to the multi-person dataset is more challenging with local minima in abundance. Since the multi-person dataset lives in a 21-dimensional space, claiming that a local minimum in the surface of figure 18.8, is truly a local minimum, is not possible. However, the surface gives an indication that a local minimum *might* be there.

Since the inverse compositional AAM uses unconstrained non-linear optimization, there is nothing to hinder it obtaining values of the parameters outside the values of ± 3 standard deviations. Thus in the course of the optimization the model instances might be illegal shapes. If the optimization were to be constrained to ± 3 standard deviations, the error surfaces, given by the parameters b_{s_1} and b_{s_2} at least, are smooth and the steepest descent direction points towards the correct minimum.

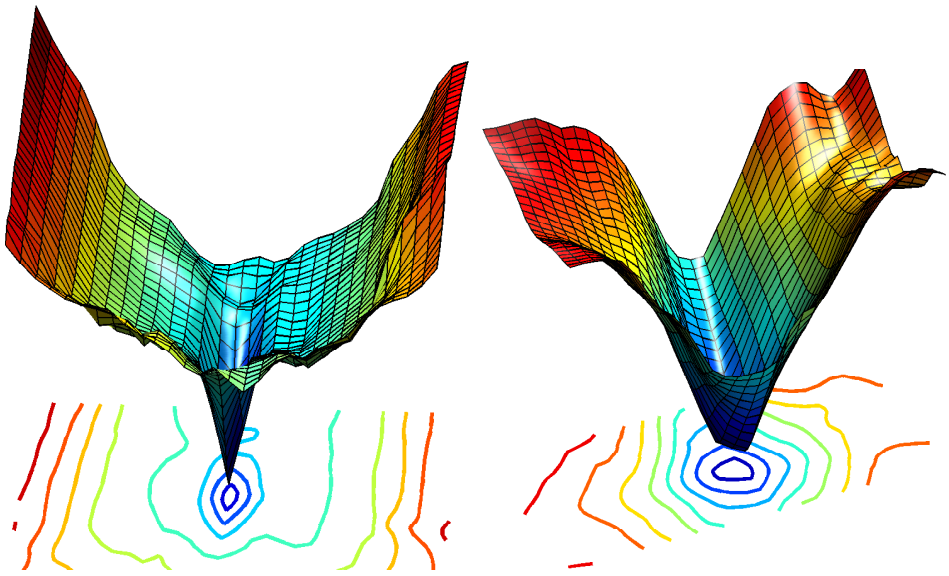


Figure 18.8: Error surfaces for (8.31) corresponding to the test images depicting the value of the error function as a function of two parameters; b_{s_1} and b_{s_1}. The parameters are varied between ± 7 standard deviations of the parameter. The left surface correspond to the AAM built from Dataset 2, and the right correspond to Dataset 3.

18.3 Tracking

The model made from Dataset 1 is used to test the tracking capabilities. The dataset consist of a vast amount of annotated video frame. The person in the video is a man engaged in conversation, laughing and gesturing, but primarily focusing his gaze towards the camera. This dataset could resemble a human-computer interaction session, where the users head is non-stationary but still maintaining his primary gaze direction towards the screen.

Figure 18.9 depicts the AAM fit of six frames of the test set. For each frame the mean Euclidian distance to the ground truth points are calculated.

Figure 18.10 illustrates the error as a function of frame number shown in the blue curve. From frame number 25 to 40 a peak in the error is visible. In figure 18.11 the related frames can be seen. The error is caused by the person pursing his lips. The AAM model trained has trouble modeling this, and thus the fit of the mouth is very bad. This can be seen in figure 18.12 which highlights the error in a close up view. However, this is an eye tracking application and not a lip reader so how about calculating the error in the fit of the eye corners. This error is plotted in the red curve in figure 18.10, and it is seen that the lip pursing does influence the eye fit. This is a problem

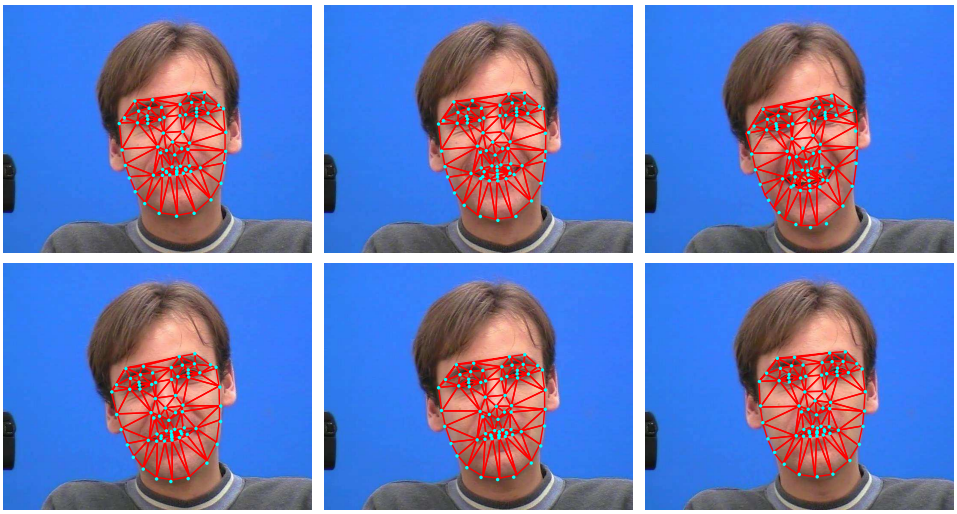


Figure 18.9: Six frames from the tracking test. The ground truth point are depicted as cyan dots. Notice how the AAM fits the face through the image sequence, although especially along the cheek the point-to-point correspondence is not exact.

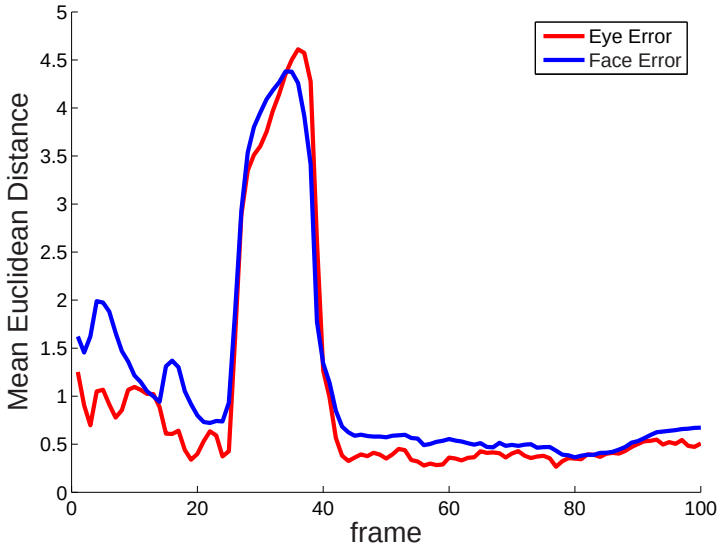


Figure 18.10: The mean Euclidean distance between the ground truth and the vertices of the AAM instance. The red curve indicates the error between the ground truth eye corners and the corresponding AAM instance vertices. The blue curve indicates the mean error for all vertices and corresponding ground truth points. Notice the peak in error between frame number 25 and 40. This corresponding images can be seen in figure 18.11.

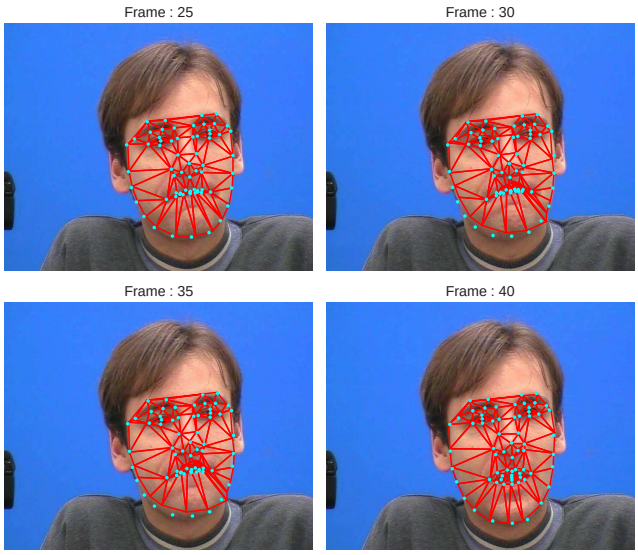


Figure 18.11: The frames related to the peak in the mean Euclidean error shown in figure 18.10.

inherent in the AAM. It uses PCA to describe the space of the training examples. Thus there might be a coupling between lip and eye movements, and so an error in one place of the face can propagate into another region. This is visible in figure 18.12 which depicts a close up of one of the eyes, and the region around the mouth. Notice the contour along the cheek has moved into the face region as a consequence of the poor fit. However, the error induced is not larger than a successful extraction of the eye region can be made.

18.4 Discussion

The responsibilities of the AAM in the eye tracking system, are to provide coordinates of the corners of the eyes, and to estimate the pose of the face by finding feature points in the face. In this chapter the capability of the AAM to overcome these tasks have been investigated.

18.4.1 Convergence

First the ability to find feature points in the face has been tested. This is done by testing the frequency of convergence of the AAM search algorithm. A comparison between an AAM built on a dataset of a single person, and an AAM built on a multi-person data set. Figure 18.5 shows the frequency

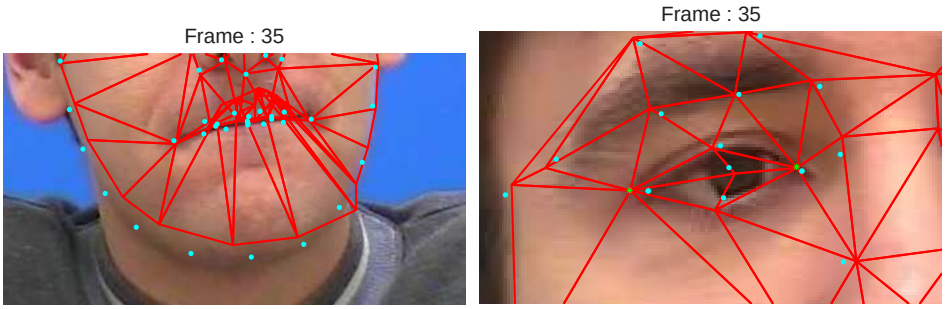


Figure 18.12: Close up frames related to the peak in the mean Euclidean error shown in figure 18.10. The AAM search seems caught in a local minimum along the cheek as depicted in the left image. Also note the inverted triangles around the mouth. This is another of the problems, using a general unconstrained optimizer. The error of the vertices along the cheek propagate to the eye as well, as seen in the right image.

of convergence. The AAM, using the inverse compositional algorithm, fits very good with single-person models but very poorly with multiple-person models. As discussed previously this may be caused by the surface of the error function contains more local minima for the multi-person AAM than for single person AAM. Also the minimization takes place in a 2-dimensional space for Dataset 2 versus a minimization in a 21-dimensional space for the multi-person AAM.

Several steps can be made for avoiding these local minima. A multiresolution approach has been described in [21] and used with success in an AAM framework. The surface plots of figures 18.7 and 18.8 indicate that the surface is more smooth and the local is very distinct when the area defined by ± 3 standard deviations. A second way to overcome the local minima would be to constrain the parameters to the range from -3 to 3 standard deviations. In [10] Baker and Matthews presents an extension of the inverse compositional algorithm which optimizes with constraints on the parameters.

A second reason for the poor performance can be that the distribution of training shape is far from Gaussian. Figure 18.13 shows a plot of the training shapes of Dataset 3 projected onto the space spanned by the first two eigenvectors. The black ellipse indicate ± 3 standard deviations. As seen a gaussian distribution is not a very good model of the distribution of the training examples. It seems that the training data is parted in to three distinct groups. Thus, any instance draw from a position between the groups could constitute an illegal shape. The magenta rings in figure 18.13 are the projections of a model instance during the five iterations of an AAM search. The result is depicted in figure 18.14 and it is clearly seen that the model instance is an illegal shape, even when the parameters are in the range of

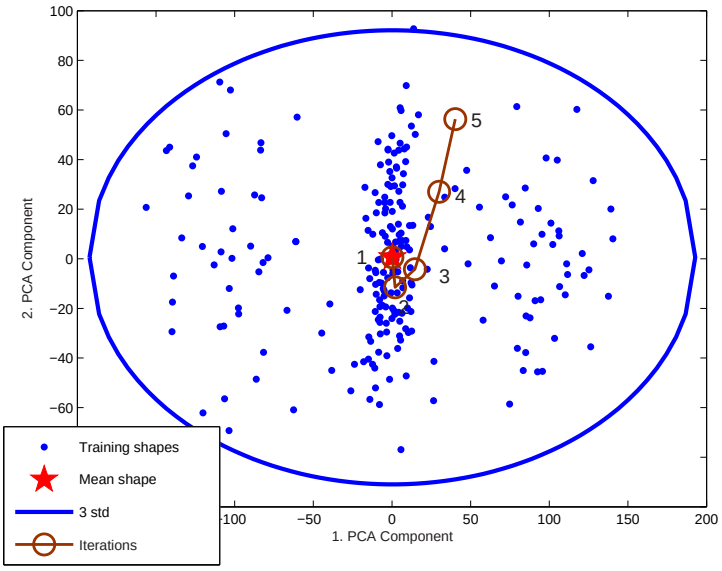


Figure 18.13: The training shapes from Dataset 3 projected into the first and second eigenvector. The blue dots indicate training examples, and the red star is the mean. The magenta rings indicate the projection of a model instance at each iteration of an AAM search. Figure 18.14 shows the resulting shape overlain the test image.

± 3 standard deviations. To overcome this problem, other distributions, for instance a mixture of gaussians, could be used.

18.4.2 Tracking

Tracking using the AAM is just an application of the AAM search on a sequence of images. The starting point in one frame is the convergence of the last frame. In this test, the ability to keep the convergence on a moving face, is tested. As seen from the test in section 18.3 the AAM is perfectly capable of tracking a face in situations of normal human behavior. However, the variations in the mimic of a human face are endless, and the AAM is only trained on a finite dataset, so outliers may occur. As seen in figure 18.10, the AAM is capable of recovering the tight fit.

18.5 Improvements

Using gaussian pyramids can solve some of the problems of the AAM relating to local minima of the surface of the error function. A set of downsampled versions of the image is used in a hierarchial scheme for fitting the AAM. The AAM is applied to the image with lowest resolution first continuing on

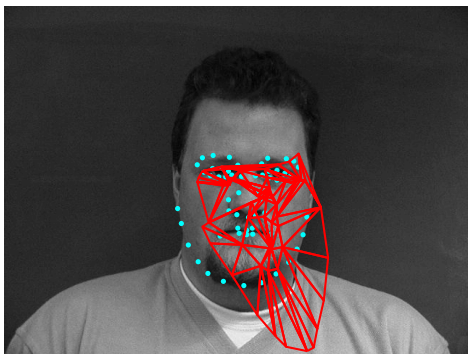


Figure 18.14: An AAM search gone wild. The AAM was initialized at the center of gravity of the ground truth data.

to the original image. The fit from the lower resolution image is propagated as initializer at the next level.

A further improvement would be to include priors on the shape parameters constraining them to be within certain boundaries.

Using gaussian mixture models[21] to model the distribution of the parameters better can also hinder illegal shapes occurring.

Matthews and Baker proposes using more sophisticated non-linear optimization methods, such as the Levenberg-Marquardt algorithm.

18.5.1 Summary

In this chapter, the ability of the AAM to function as a feature detector and tracker of facial features has been tested. From the tests it has been shown that the AAM is a suitable tool in the overall eye tracking system presented in this thesis.

Chapter 19

Eye Tracking

Detection of the human eye is a difficult task due to a weak contrast between the eye and the surrounding skin. As a consequence, many existing approaches use close-up cameras to obtain high-resolution images[36]. However, this imposes restrictions on head movements. The problem can be overcome utilizing a multiple camera setup[56][92].

The eye trackers from chapter 13 and 14 are evaluated below. In particular we propose a robust algorithm for swift eye tracking in low-resolution video images. We compare this algorithm with a proven method EM active contour algorithm [36] and relate the pixel-wise error to the precision of the gaze determination.

The importance of image resolution is investigated by comparison of two image frame setups with different resolution. One containing close up images - denoted as high-resolution images, although the resolution is [351x222] pixels; or 0.078 megapix. Another one containing a down-sampled version hereof ensuring identical conditions such as illumination, reflections, and eye movements. The low-resolution images corresponds to the full-face image setup, utilizing a standard digital camcorder of [720x576] pixels, seen in figure 17.3.

A couple of examples from the 378 frame video sequence is seen in figure 19.1. When the camera lies on the optical axis of the eye, the contour can be modeled as a circle. However, when the gaze is turned off the optical axis, the circle is rotated in 3D space, which can be interpreted as an ellipse in the image plane. Thus, the shape of the contour changes as a function of the gaze direction, which is seen figure 19.1.

Figure 19.1: Examples from the dataset. (*Top figures:*) High-resolution data [351x222] pixels; (*Top left:*) The iris and pupil have a diameter of 33 and 83 pixels respectively. They can both be approximated by circles when the gaze is straightforward. (*Top right:*) When the gaze is turned off the optical axis, the circle is rotated in 3D space, which can be interpreted as an ellipse in the image plane. Thus, the resolution is $(57, 80)$ and $(26, 40)$ in the x- and y-direction respectively. (*Bottom figures:*) The downsampled version of the upper figures; The resolution is decreased to [88x53] pixels.

Notation

A number of figures, where the names are shortened, are presented in the following. To ease understanding, the shortened names are listed below:

AC	Active contours.
Cons	Constraining the shape of contours.
EM	Expectation-maximization optimization of contours.
DT	Deformable Template Matching optimization of contours.
Thres	Double Thresholding.
TM	Template Matching.
TMref	Template Matching refined by the ellipse fitting algorithm.
TMrgb	Color-Based Template Matching.
Deform	Deformable Template Matching initialized by double thresholding.

19.1 Performance of Segmentation-Based Eye Trackers

Recall the eye trackers presented in chapter 13: The heuristic double thresholding, template matching, and deformable template matching. These methods estimate the center of the pupil. For each frame the error is recorded as the difference between a hand annotated ground truth and the output of the algorithms. This may lead to a biased result due to annotation error. However, this bias applies to all algorithms and a fair comparison can still be made. The mean error is shown in figure 19.2.

The refined version of template matching actually improves the precision as intended. The cost is, however, longer computation time and a worsening in robustness due to the lack of shape constraints.

The color-based template matching method exploits the fact that colors of the eye region differs significantly from the surrounding skin. Nevertheless, the method may be confused due to the heavy edges found e.g. in the eyebrows. This can be overcome, but the robustness of the method is not satisfactory in general.

Double tresholding is clearly the fastest method with a framerate close to 70 frames per second for low-resolution images. This is more than needed in realtime videos (25fps). The relationship between pixels are not considered by thresholding, instead each pixel are classified independently of its neighbors and is therefore sensitive to noise. The accuracy is, unfortunately, not good enough to utilize the method as a "stand-alone" eye tracker. Instead, it can be used to initialize other methods such as the deformable template matching method. This combination results in a fast, accurate eye tracker

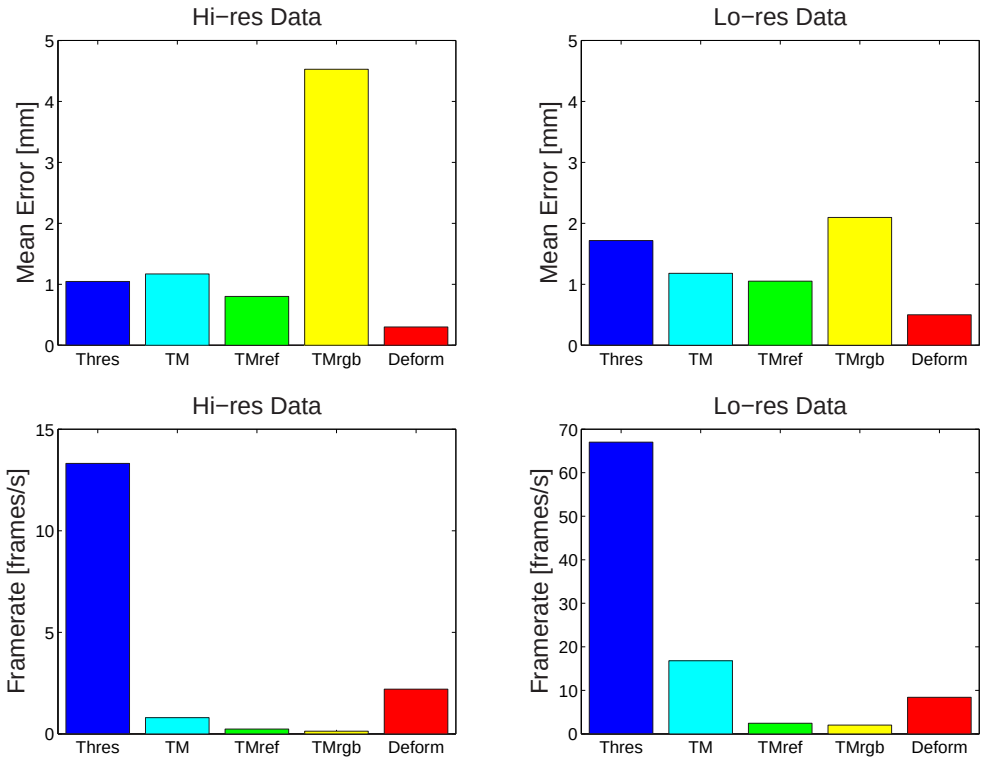


Figure 19.2: Performance of the segmentation-based trackers. (*Top figures:*) Mean error; The lowest error is obtained by use of deformable template matching, while the color-based template matching has the highest. (*Bottom figures:*) The framerates of the methods are evaluated as the number of frames processed pr. second. Double Thresholding is the fastest method.

as depicted in figure 19.2.

The precision of the template matching - basic and refined - and the deformable template matching is accurate almost independent of resolution and they are very fast for low resolution images. The precision of the heuristic double thresholding worsens when decreasing resolution. The edges confusing the color-based template method is smoothed out to some extent, consequently the performance is improved. Interpreting the results, the deformable template matching method should be chosen if one focus on high accuracy. On the other hand, if one require as less computation time as possible, the basic template matching should be utilized.

19.2 Performance of Bayesian Eye Trackers

A Bayesian approach for eye tracking is presented in chapter 14. The eye is tracked from a video sequence utilizing the active contour method [36][64] and particle filtering, but extended with constraints, robust statistics, and a novel refinement method - Deformable template matching.

A few examples of the tracker is depicted in the theory part in figure 14.8. Tracking the eyes during blinking is a challenging task. An example of this phenomena is illustrated in figure 19.3. The extension of the contour method, increases the robustness to outliers and relaxes the importance of the hypotheses along the contour around the eyelids. The resulting estimate of the iris center is seen, in bottom right figure 19.3, to be fine.

The performance of the active contour methods are evaluated by averaging over 10 runs of the video frame sequences. Figure 19.4 depicts the error as a function of the number of particles used, for low resolution and high resolution images respectively. The errors of the three different active contour algorithms are shown; basic, with EM refinement, and with deformable template refinement. In addition, the constraints regarding the mentioned methods are evaluated.

The proposed constraints on the active contour generally improves the accuracy of the fit. However, utilizing the constraints with more than 50 particles in high-resolution images, worsens the precision. This is caused by the fact that the contour shape changes with the gaze direction; in the extreme directions of gaze, the contour is deformed to an extent which is penalized by the constraints. Thus, there is no need to constraint the deformation, when the number of particles is sufficiently large on high-resolution images. The constraints should therefore be utilized on low-resolution images, and to avoid a break down of the algorithm due to poor data. In contrast, the constraints on the deformable template never worsens the accuracy. The two

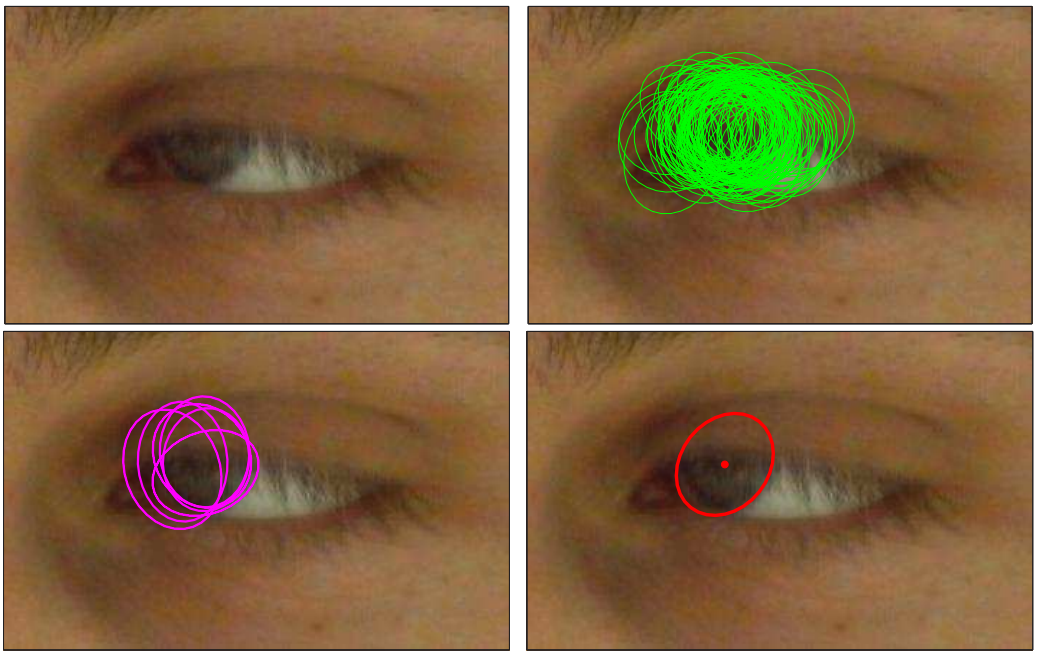


Figure 19.3: The active contour algorithm utilizes particle filtering. (*Top left:*) A frame challenging the eye trackers; eye blinking. (*Top right:*) A set of particles is drawn from a prediction prior. (*Bottom left:*) By evaluation of the hypotheses regarding the particles, a new re-sampled set of particles is obtained. This is the estimated posterior state distribution. (*Bottom right:*) The current state is then found by taking the sample mean of the estimated posterior probability. Notice that the algorithm is capable to estimate the center of iris, despite the fact that a major part of the iris is occluded by the eyelid.

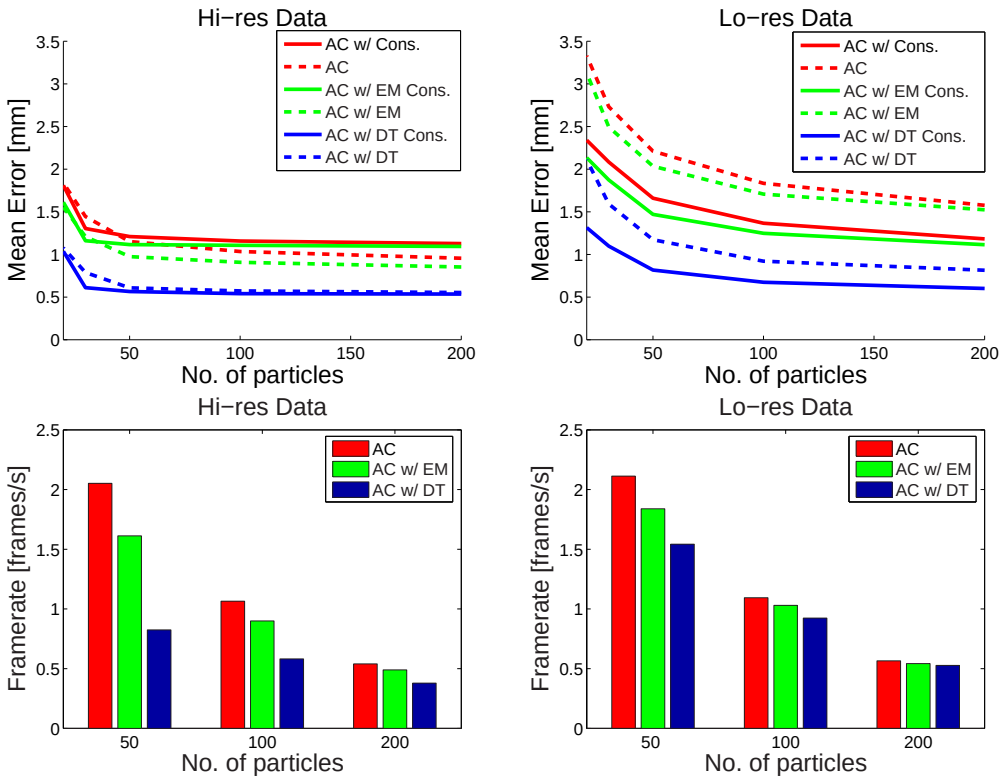


Figure 19.4: Performance of the active contour averaged across 10 runs of each method. (*Top figures:*) The error of the active contour algorithms as a function of the number of particles and resolution. Constraining the shape deformation clearly improves the accuracy of the contour method in low-resolution and high-resolution images with few particles. Utilizing the constraints with more than 50 particles in high-resolution images, worsens the precision. In general, the deformable template refinement has the best precision. (*Bottom figures:*) The framerate of the methods are evaluated as the number of frames processed pr. second. The framerate is highly dependent on the number of particles, and the basic contour is considerably faster than the refined versions. On the other hand, the number of particles needed is reduced significantly by use of refinement.

error curves converges to the same value when the number of particles is increased. Contrary to the iris, the pupil is not partly occluded except when blinking. Therefore, the method is more accurate although the number of particles is lower.

The refinement by the deformable template outperforms the EM method - Even when comparing low-resolution image to EM refinement on high-resolution images. The cost is an increased number of computations, which is resolution dependent. The basic, or even EM refined, active contour is faster than the deformable refinement utilizing few particles in high-resolution data. Conversely, increasing the number of particles or by use of low-resolution images, this refinement is not significant slower.

In general the methods perform better in high-resolution images compared to low-resolution images, where the dependency on the number of particles is increased. The cost of increasing the number of particles is an increased number of computations - leading to a lower framerate. The deformable template is, however, only dependent on the resolution. Hence, the framerate is increased when the number of pixels is decreased.

Corneal reflections caused naturally by illumination challenges the eye tracking methods. Typically, this phenomenon influences on the accuracy - The estimate is biased to some extent. A frame sequence exemplifying handling of corneal reflections is shown in figure 19.5. The light conditions result in strong reflections in bottom left figure. By weighing the hypotheses as proposed in section 14.6 and utilizing robust statistics regarding the deformable template, increases the robustness and accuracy of the estimate of the iris center. An illustrative example is found in figure 14.4.

19.2.1 Ability to Handle Eye Movements

The particles are drawn from a narrower distribution in frame sequences where the center location of iris is nearly static. In this case, sudden eye movements may confuse the contour method. The re-sampling of particles, broadens the state distribution and thus recovering the fit in a few frames.

The performance of the contour method under dynamic and static states is shown in figure 19.6. The performance is certainly affected by eye movements, when utilizing few particles. The error of the dynamic frames is in general a bit larger, but vanishes when the number of particles is increased. The deformable template has, surprisingly, a lower error in these frames. This is caused by the fact, that the error is in average larger in the extremes of the gaze direction; the eye is typically static in these states. Hence, the error function is biased to some extent. The influence on the low-resolution data is less compared to the high-resolution. This is due to the relative smaller

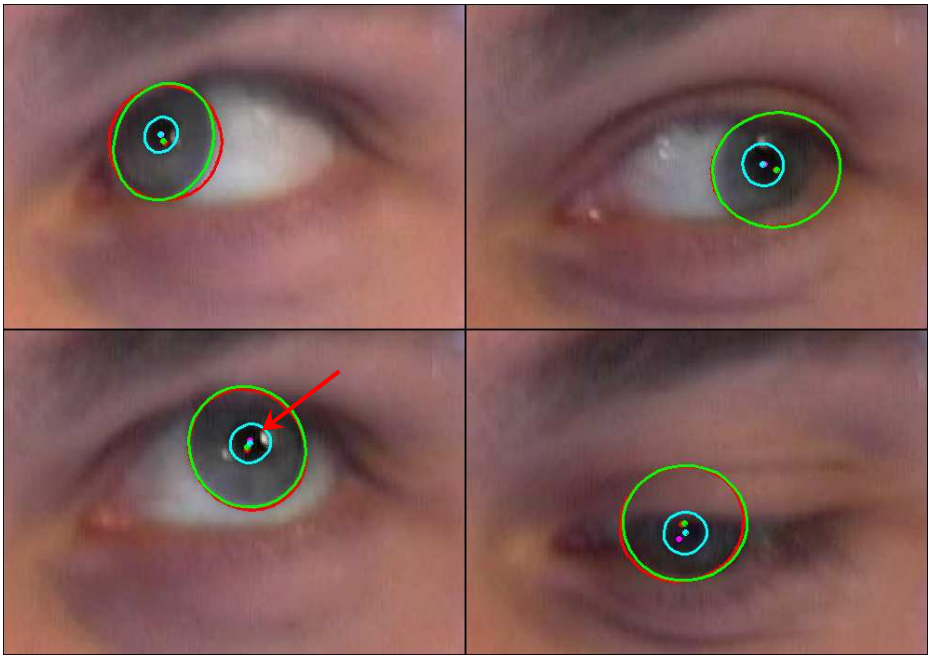


Figure 19.5: The resulting fit on four frames from a sequence - the red contour indicates the basic active contour, green indicates the EM refinement and the cyan indicates the deformable template initialized by the heuristic method. The images illustrates the benefit of fitting to the pupil rather than the iris. Using robust statistic the influences from corneal reflections on the deformable template fit are ignored as depicted in the left bottom image. In addition, weighting the hypotheses improves the fit when a part of the iris is covered or during blinking.

motion in pixels, but the error is obviously greater on average.

19.3 Comparison of Segmentation-Based and Bayesian Eye Trackers

The bayesian and segmentation-based trackers have both pros and cons regarding accuracy, robustness and speed. The mean error of the center of iris is computed and the results of the presented methods are shown in table 19.1. Additionally, the influence on the gaze and framerate is presented. The active contour uses 200 particles ensuring optimal accuracy, but decreasing the framerate. The number of particles is a trade-off between accuracy and computation time as depicted in figure 19.4. The deformable template model initialized by the heuristic method - Double thresholding - is the most accurate tracker. Additionally, initializing by active contours leads to high

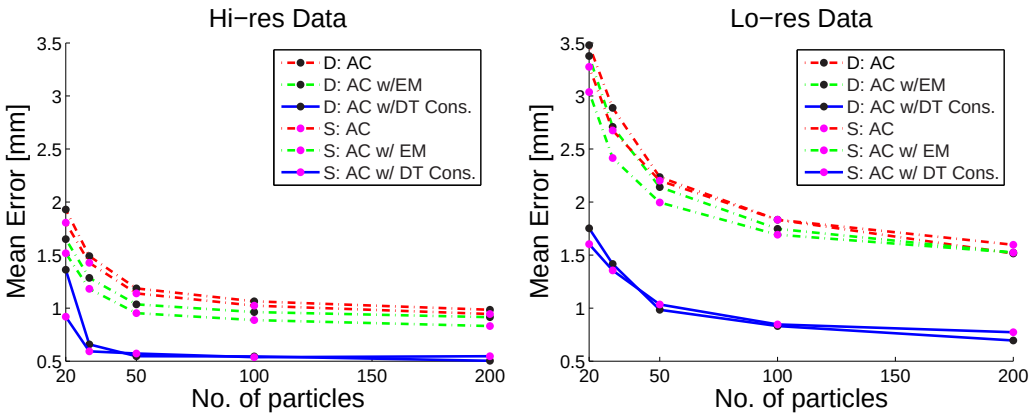


Figure 19.6: Investigation of performance of the active contour tracker, when the state propagation is dynamic (**D**) or static (**S**) respectively. The performance is certainly affected by eye movements, when utilizing few particles. The error of the dynamic frames is in general a bit larger, but vanishes when the number of particles is increased. The deformable template has, surprisingly, a lower error in these frames. This is caused by the fact, that the error is in average larger in the extremes of the gaze direction; the eye is typically static in these states. Hence, the error function is biased to some extent. The influence on the low-resolution data is less compared to the high-resolution. This is due to the relative smaller motion in pixels, but the error is obviously greater on average.

precision. The highest framerate is obtained using double thresholding and basic template matching.

The color-based template matching is not evaluated further due to poor performance. Neither is the heuristic threshold method, since the method is utilized to initialize the deformable template method.

19.3.1 Influence of Gaze Direction

Intuitively, the error is highly dependent on the gaze direction. When the gaze direction is inward or outward, a part of the iris is covered by the eyelid, hence, fewer points are available for contour estimation - challenging the algorithms. This fact is investigated and depicted in figure 19.7 and 19.8.

Notice, that the number of particles needed for the active contour method, is considerably lower for deformable template refinement. The pupil is, in contrast to the iris, not covered with the only exception when blinking. Therefore, the method is more accurate although the number of particles is lower.

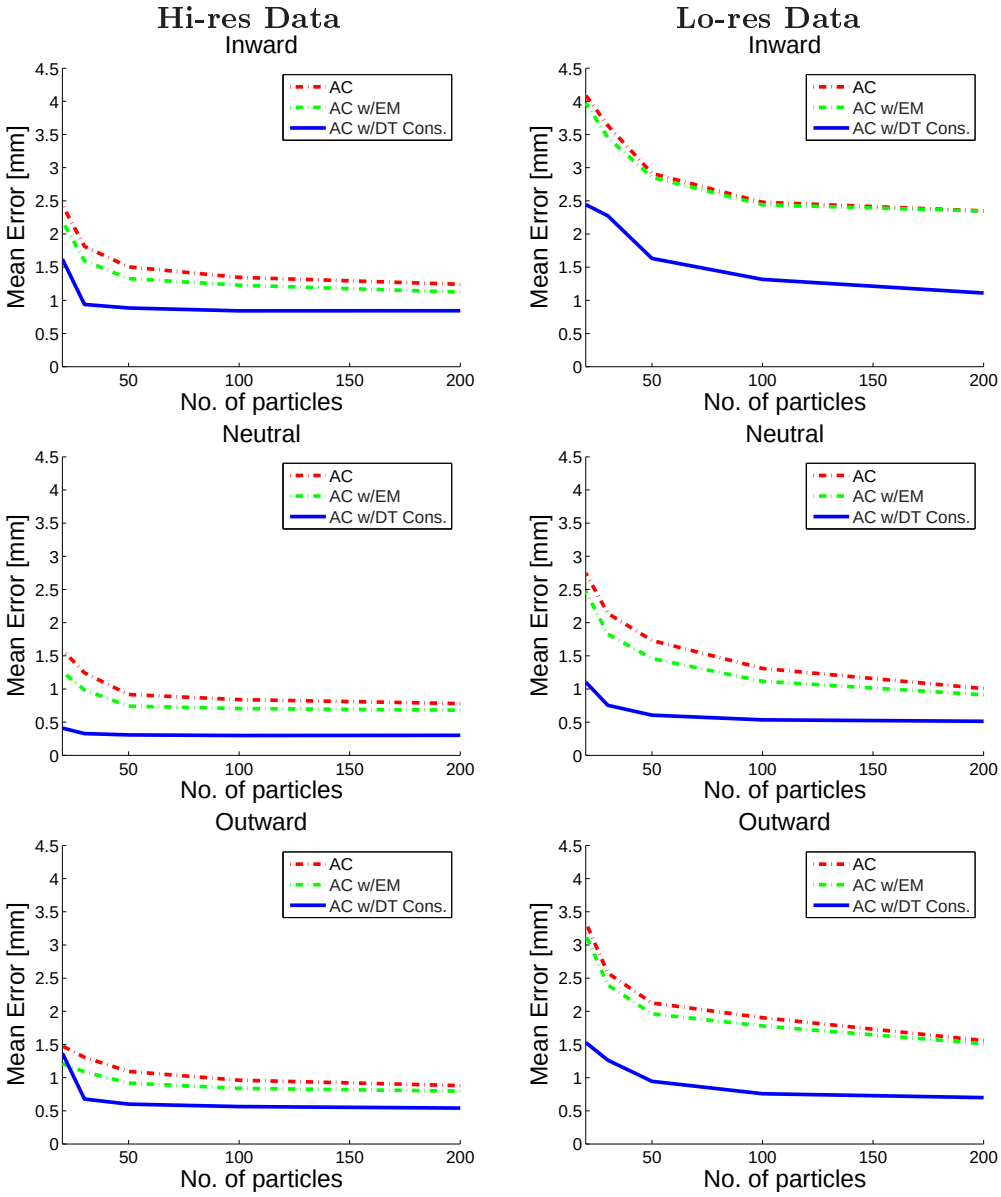


Figure 19.7: Investigation of performance of the active contour tracker, when the gaze direction is inward (towards the nose), neutral, and outward (away from the nose). (*Left Column:*) High-resolution data. (*Right Column:*) Low-resolution data. The error is in general highly dependent on the gaze direction. When the gaze direction is inward or outward, a part of the iris is covered by the eyelid, hence, fewer points are available for contour estimation - challenging the algorithms. Notice, that the number of particles needed, is considerably lower for deformable template refinement. Contrary to the iris, the pupil is not partly occluded except when blinking. Therefore, the method is more accurate although the number of particles is lower.

	Hi-res Data			Lo-res Data		
Method	$E(x,y)$[mm]	$E(\theta)$	[frame/s]	$E(x,y)$[mm]	$E(\theta)$	[frame/s]
AC w/Cons.	1.2	5.2	0.54	1.2	5.5	0.57
AC	0.95	4.1	0.54	1.5	7.3	0.57
AC w/EM Cons.	1.1	5.1	0.49	1.2	5.2	0.55
AC w/EM	0.85	3.7	0.49	1.5	6.9	0.55
AC w/DT Cons.	0.53	2.4	0.38	0.60	2.8	0.49
AC w/DT	0.55	2.5	0.38	0.81	3.7	0.49
Thres.	0.95	4.4	13.	1.7	8.0	67.
TM	1.2	5.5	0.80	1.2	5.5	17.
TMref	0.79	3.7	0.23	1.1	4.9	2.4
TMrgb	4.5	21.	0.13	2.0	9.5	2.0
DT	0.30	1.4	2.2	0.49	2.3	8.4

Table 19.1: Speed and precision comparison of the algorithms - red indicates remarkably fine results, while blue poor results. The active contour uses 200 particles ensuring optimal accuracy, but decreasing the framerate. The number of particles is a trade-off between accuracy and computation time as depicted in figure 19.4. The deformable template model initialized by the heuristic method - Double thresholding - is the most accurate tracker. Additionally, initializing by active contours leads to high precision. Double thresholding and basic template matching have the highest framerate.

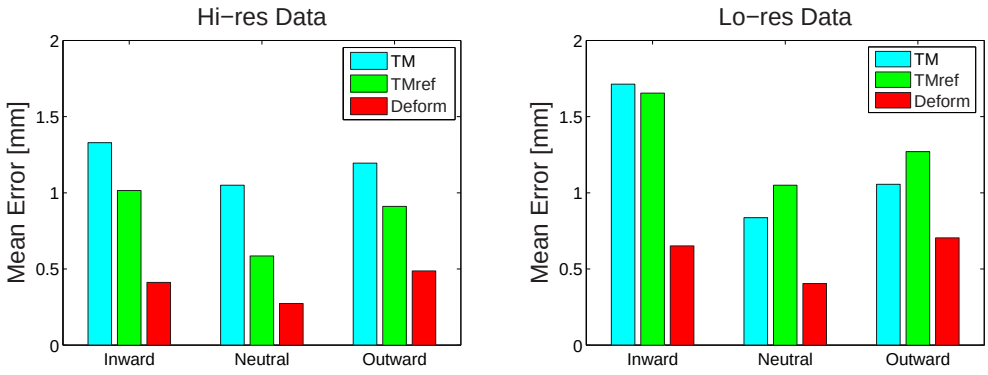


Figure 19.8: Investigation of performance of the best segmentation-based trackers, when the gaze direction is inward (towards the nose), neutral, and outward (away from the nose). (*1*) High-resolution data. (*2*) Low-resolution data. The error is highly dependent on the gaze direction. A part of the iris is covered by the eyelid, when the gaze direction is inward or outward. As a consequence, fewer points are available for contour estimation. The pupil is, in contrast to the iris, not covered with the only exception when blinking. Therefore, the method is more accurate although the number of particles is lower.

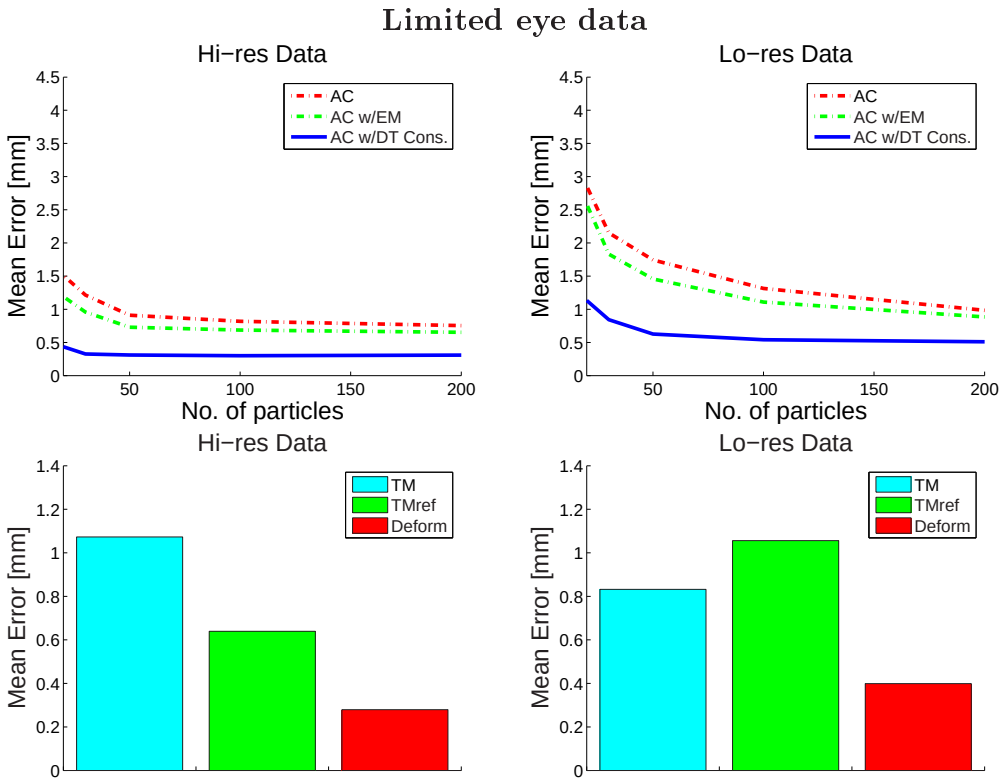


Figure 19.9: Utilizing eye tracking in human computer interaction, limits the needed freedom of gaze. A person sitting in front of (60cm from) a 19" monitor is able to reach every point by gaze within 17 degrees. Illustrated is the performance of the eye trackers applied to a limited dataset with gaze directions within 20 degrees. The error is of same magnitude as for neutral gaze illustrated in figure 19.7 and 19.8. The precision is found in table 19.2

.

19.3.2 Human Computer Interaction

Above is demonstrated that the error is highly dependent on the gaze direction. When the gaze direction reaches the extremes in the horizontal direction, the field of view is approximately within ±50 degrees. This is far beyond what is needed in many applications, e.g. human computer interaction. Utilizing eye tracking in human computer interaction, limits the needed freedom of gaze. Suppose a person sitting in front of (60cm from) a 19" monitor. Every point can be reached with a field of view within 17 degrees.

The dataset is now limited to gaze directions within 20 degrees. The error is of approximately same magnitude as for neutral gaze illustrated in figure

	Hi-res Data			Lo-res Data		
Method	$E(x,y)$[mm]	$E(\theta)$	$E(poi)$[cm]	$E(x,y)$[mm]	$E(\theta)$	$E(poi)$[cm]
AC	0.76	3.5	4.1	0.98	4.5	5.2
AC w/EM	0.65	3.0	3.5	0.88	4.1	4.8
AC w/DT Cons.	0.30	1.4	1.6	0.50	2.3	2.6
TM	1.1	5.0	5.9	0.80	3.7	4.3
TMref	0.64	3.0	3.5	1.1	4.9	5.7
DT	0.28	1.3	1.5	0.40	1.8	2.1

Table 19.2: The error on the center of iris, gaze and inaccuracy on the screen (**poi** - point of interest) is compared on a limited dataset for human computer interaction - red indicates remarkable fine results, while blue poor results. The active contour uses 200 particles as previous (see table 19.1). High-resolution data is in general more accurate than low-resolution. Nevertheless, the deformable template model initialized by the heuristic method - Double thresholding - is not as dependent on the resolution as the other methods. In addition, the deformable template model initialized by active contours is an accurate eye tracker. Surprisingly, the basic template matching method performs better for low resolution - than high resolution.

19.7 and 19.8. The performance of the limited data is depicted in figure 19.9, and the error on the center of iris, gaze and inaccuracy on the screen is found in table 19.2.

High-resolution data is in general more accurate than low-resolution. Nevertheless, the deformable template model initialized by the heuristic method - Double thresholding - is not as dependent on the resolution as the other methods. In addition, the deformable template model initialized by active contours is an accurate eye tracker. Surprisingly, the basic template matching method performs better for low - than high resolution. The ring template filter, described in section 13.2 applied on low-resolution, performs better due to the relative broader filter. Increasing the width of the ring template, regarding high-resolution data proportionally to the low-resolution, decreases the performance. The increased amount of gradients confuses a broad filter.

19.4 Gaze Estimation

A dataset is collected under relatively controlled conditions as seen in figure 17.1. The gaze direction is turned at a collection of markers placed at some controlled points in 3D space. By recording multiple sequences of different head and eye positions, the anatomical constants can be computed, which is used for gaze determination. All variation is in the yaw, pitch and roll direction within ± 30 degrees relative to frontal images.

In spite of the relative controlled conditions while recording data, the gaze

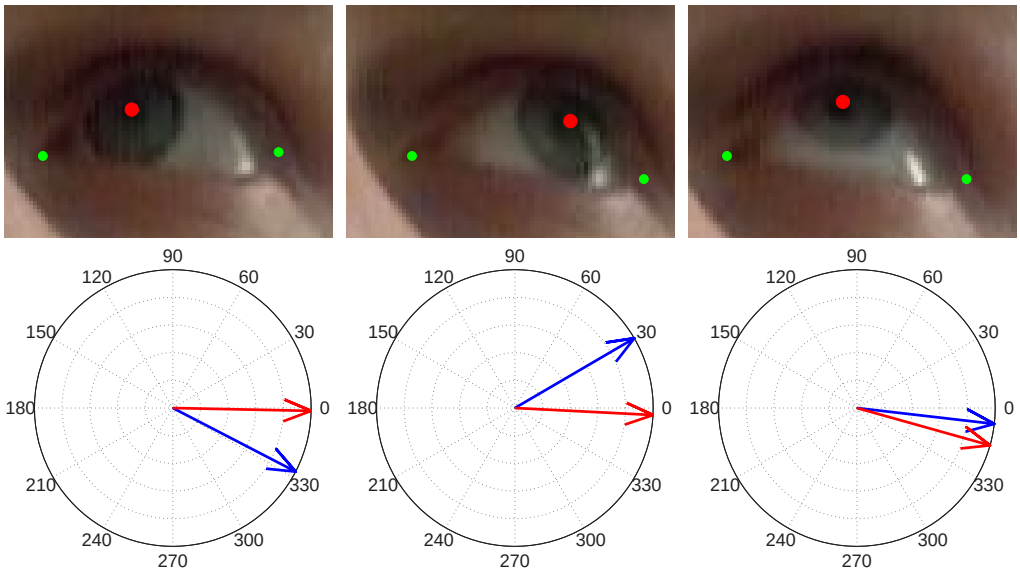


Figure 19.10: Examples of gaze determination. (*Top*) The eye corners detected by AAM (green) and pupil location (red) is utilized for gaze determination. (*Bottom*) The corresponding directions - The blue arrow corresponds to θ_x, while the red arrow corresponds to θ_y.

direction cannot be evaluated fairly against the ground truth. This is partly caused by the overall uncertainty, and partly due to the physical difficult separation of face pose and pupil location for a person; e.g. how to separate the horizontal head and eye direction $(\phi_x, \theta_x) = (20, 0)$ with $(\phi_x, \theta_x) = (17, 3)$. Information concerning position and yaw can be obtained by use of a gyroscope and an accelerometer[85]. However, utilizing eye tracking in human computer interaction, the mapping from the pose of eye to point on screen - calibration - is computed similarly to the one described.

The gaze error found in the above tables are theoretic, but are still usable for evaluation of the methods.

19.5 Discussion

Several eye tracking methods has been presented and evaluated through a set of experiments. The methods have both pros and cons regarding accuracy, robustness and speed.

The chosen number of particles utilized by active contours is a trade-off between accuracy and computation time. The number of particles can be reduced by applying the EM algorithm to refine the posterior state. Additionally, refining by deformable template improves the fit.

The proposed constraints on the hypotheses result in a better and more robust estimate of the contour. Ignoring these may confuse the algorithm to shrink and fit to the corneal reflections in outlier frame sequences. When the size of the contour is decreased, it cannot grow to the size of iris again.

Constraints regarding the shape of the contour increases the robustness, but utilizing the constraints with more than 50 particles in high-resolution images, worsens the precision. Consequently, there is no need to constrain the deformation, when the number of particles is sufficiently large on high-resolution images. The constraints should therefore be utilized on low-resolution images, and to avoid a break down of the algorithm due to poor data. In contrast, the constraints on the deformable template never worsens the accuracy.

Among the segmentation-based trackers, the deformable template matching method should be chosen if one focus on high accuracy. On the other hand, if one require as low computation time as possible and the images are of low-resolution, the basic template matching should be utilized.

In general, the highest framerate is obtained using double thresholding and basic template matching. The most accurate tracker is the deformable template model initialized by the heuristic method - double thresholding. It is shown that the deformable template model is accurate independent of resolution and it is very fast for low resolution images. This makes it useful for head pose independent eye tracking. Additionally, initializing by active contours leads to high precision. The improved performance is caused by the object being tracked - the pupil. In contrast to the iris, the pupil is not occluded by the eyelids with the only exception during blinking. Intuitively, the active contour method should be modified to estimate the contour of the pupil. However, the contour between pupil and iris is weak resulting in an unstable tracker. Test has shown that if the variance of the state propagation is broad enough, the contour algorithm, sooner or later, expands the contour to fit to the iris.

19.5.1 Interpretation of Performance

The accuracy of the gaze determination is satisfactorily compared to other proven methods. Ishikawa et al.[43] reports an average error of 3.2 degrees. Moreover, their proposed method - combining an AAM with a refined template matching method for iris detection - is evaluated in a car. A frame is exemplified in figure 19.11, where the yellow circle corresponds to a 5.0 degree gaze radius.

Tobii Technology AB reports an average error of 0.5 degrees in front of a 17" monitor[89]. This is a commercial system using infrared illumination.

Figure 19.11: Image from [43]. A driver follows a person walking outside by gaze. The yellow circle corresponds to a 5.0 degree gaze radius.

But, how accurate do we expect the eye tracker to be? In fact, the gaze is not a stringent line in space. The human eye perceives the immediate surrounding of its point of gaze through its peripheral vision, thus an error of 1 degree obtained from the tracker is lost in the noise of how the human eye works anyway[83].

Example

While staring at this **word**, other words are clearly seen. Without moving the eyes, a couple of words in front of, behind of, and on the line below can probably be read too. It is, however, harder to make out specific words that are a couple paragraphs away. Hence, with a margin of error of plus or minus 1 degree of visual angle, this error falls within the margin of error of the natural function of the human eye.

> *"... it is completely natural for people to focus just above or just below the line of text that they are actually reading."*
>
> - C. Johnson et al.[83].

Part IV

Discussion and Future Work

Chapter 20

Summary of Main Contributions

The main objectives set forth was to:

> *Develop a fast and accurate eye tracking system enabling the user to move the head naturally in a simple and cheap setup.*

The objective was divided into three components - Face detection and tracking, eye tracking and gaze determination. The gaze precision, however, is totally dependent on the quality of the face and eye tracking components. Thus, improving gaze precision, has to be done at the two lower levels.

In this thesis, a fully functional eye tracking system has been developed. It complies to the objectives set for the thesis:

- A face tracker based on a new, fast, and accurate Active Appearance Model of the face. It segments the eye region, and provides the pose of the head.
- Several eye tracking algorithms - segmentation-based and bayesian - has been proposed and tested. They provide fast and accurate estimate of the pupil location.
- Determination of gaze direction is obtained by exploiting a geometric model. With this, the true objective of the eye tracking system is accomplished.

20.1 Face Detection and Tracking

Regarding face detection and tracking, a complete functional system has been implemented. The theory and application of the Active Appearance Model have been described, with the main points:

- The building of an Active Appearance Model of faces.
- The model fitting algorithm which uses a new, faster, analytical gradient descent based optimization rather than the usual ad-hoc methods.
- A 3D model of the face is used to extract head pose from the fit of the AAM.

20.2 Eye Tracking

Several eye tracking algorithms has been proposed, described and tested. The main difference is the propagation model - that is, how the system dynamics are propagated given the previous state estimates. While the segmentation based tracking uses the last estimate as starting point for a segmentation method, or even no knowledge of old states at all, the bayesian tracker predicts the state distribution given previous state. The main contributions are:

Segmentation-Based Tracking

- A fast adaptive double thresholding method. The high threshold can be interpreted as a filter regarding the low threshold.
- Template matching of two templates are merged.
- Template matching including a refining step and extended with outlier detection.
- Color-based template matching utilizing information from color gradients.
- Deformable template matching capable of handling corneal reflections by utilizing robust statistics. Additionally, we constrain the deformation. The method is based on a well-proven optimization algorithm - Newton with BFGS updating.

Bayesian Eye Tracking

The proven active contour algorithm[36] is extended to improve robustness and accuracy:

- Weighing of the hypotheses to relax their importance along the contour around the eyelids. Moreover, it penalizes contours surrounding bright objects.

- Robust statistics to remove outlying hypotheses stemming from corneal reflections.
- Constraining the deformation of the contour regarding the magnitude of the axes defining the ellipse.
- Refinement of the fit by a deformable template model of the pupil.

Chapter 21

Propositions for Further Work

In this chapter naturally extensions to the algorithms, developed during this master thesis work, are proposed.

- The Levenberg-Marquardt non-linear optimization algorithm would naturally extend the existing AAM algorithm using the Gauss-Newton algorithm. This would enable faster convergence, stemming from larger initial steps in the optimization.
- Utilizing prior knowledge of the shape of a face, could be incorporated in the algorithm in the form of priors on the parameters.
- Implementing an optimization scheme using gaussian pyramids would be a fast way to improve the fitting.
- A new shape model could be tested. One which utilizes global knowledge of the face, such as inter-relationship between the the face and the mouth, the location of eyebrows etc., to improve the accuracy and speed of the fit.
- Extending the iris contour model to a full shape model of the eye, may provide additional accuracy to iris detection. Hence, hypotheses occluded by the eyelids can be rejected.
- Optimization of the speed regarding the eye tracking can be obtained through a variable number of utilized particles. Thus, increasing the number of particles due to increased uncertainty.
- The constraints on the deformation can be extended, exploiting the estimation of eye corners obtained from the AAM. Consequently, the method should constrain the contour to be circular when the gaze direction is neutral, but ellipsoid elsewhere.

Chapter 22

Conclusion

As computers has become faster, the way we apply them become increasingly complex. This opens a wide range of possibilities, for using computers as a tool for enhancing the quality of life, learning human behavior, and increasing the general safety. Today eye tracking is a technology in the making, and we are just opening Pandoras box. Ensuring the success of eye tracking applications, wide accessibility is required. This proposes a dilemma; low cost equals low performance. To overcome this problem, sophisticated data analysis and interpretation are required.

In this thesis, we have proposed an eye tracking system, suitable for use with low cost consumer electronics. A system capable of tracking the eyes, while putting no restraint on the movement of the head. Novel algorithms, along with extensions of existing ones, have been introduced, implemented and compared to a proven, *state of the art*, eye tracking algorithm.

An innovative approach, based on a deformable template initialized by a simple heuristic, leads to the best performance. The algorithm is stable towards rapid eye movements, closing of the eye lids, and extreme gaze directions. The improved accuracy is due to tracking of the pupil rather than the iris. This is particularly the case when a part of the iris is occluded. Additionally, it is shown that the deformable template model is accurate, independent of the resolution of the image, and it is very fast for low resolution images. This makes it useful for head pose independent eye tracking. The precision of the estimated gaze direction is satisfactory, bearing in mind how the human eye works.

In preparation of this thesis, countless lines of code has been written, an endless amount of figures has been printed, and thorough investigations has been conducted leading up to the algorithms presented. However, many stones has been left unturned; a few mentioned in chapter 21.

After six months ... we have just *opened* our eyes...

Appendix A

Face Detection and Tracking

A.1 Piecewise Affine Warps

In this framework, a warp is defined by the relationship between two triangulated shapes, as seen in figure A.1. The left mesh is a triangulation Each triangle in the left mesh has a corresponding triangle in the right mesh, and this relationship defines an affine transformation.

Figure A.2 depicts two triangles, where the right triangle is a warped version of the left. Denote this warp $\mathbf{W}(\mathbf{x};\mathbf{b}_s)$. If $\mathbf{x}_1$, $\mathbf{x}_2$ and $\mathbf{x}_3$ denotes the vertices of the left triangle, the coordinate of a pixel $\mathbf{x}$ is written as,

$$\begin{aligned} \mathbf{x} &= \mathbf{x}_1 + \beta(\mathbf{x}_2 - \mathbf{x}_1) + \gamma(\mathbf{x}_3 - \mathbf{x}_1) \\ &= \alpha\mathbf{x}_1 + \beta\mathbf{x}_2 + \gamma\mathbf{x}_3, \end{aligned} \tag{A.1}$$

where $\alpha = 1 - (\beta - \gamma)$, $\alpha + \beta + \gamma = 1$ and $0 < \alpha, \beta, \gamma < 1$. Warping a pixel $\mathbf{x} = (x, y)^\top$ is now given by transferring the relative position within the

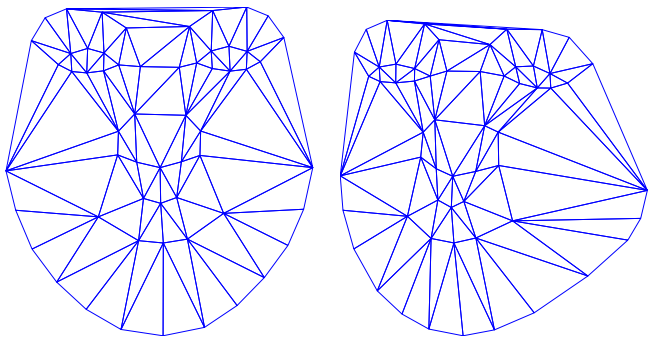


Figure A.1: Left: The mean shape triangulated using the Delaunay algorithm. Right: A training shape triangulated.

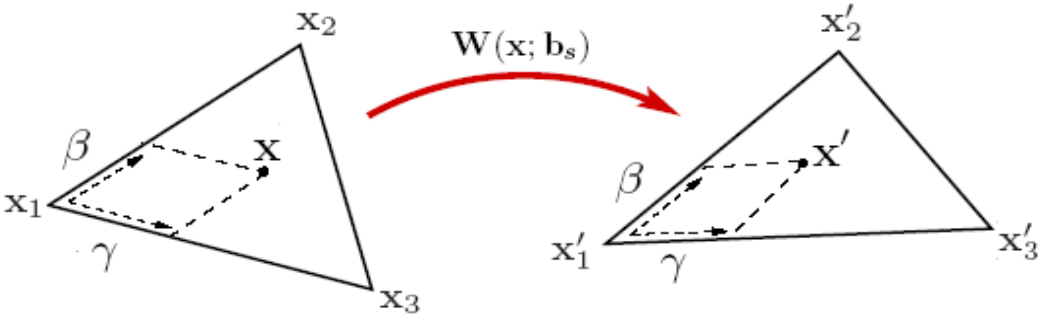


Figure A.2: Piecewise affine warping[57]. A pixel $\mathbf{x} = (x, y)^\top$ inside a triangle in the base mesh can be decomposed into $\mathbf{x}_1 + \beta(\mathbf{x}_2 - \mathbf{x}_1) + \gamma(\mathbf{x}_3 - \mathbf{x}_1)$. The destination of $\mathbf{x}$ under the warp $\mathbf{W}(\mathbf{x};\mathbf{b}_s)$ is $\mathbf{x}'_1 + \beta(\mathbf{x}'_2 - \mathbf{x}'_1) + \gamma(\mathbf{x}'_3 - \mathbf{x}'_1)$.

triangle spanned by $[\mathbf{x}_1\,\mathbf{x}_2\,\mathbf{x}_3]$ determined by α,β and γ, onto the triangle spanned by $[\mathbf{x}'_1\,\mathbf{x}'_2\,\mathbf{x}'_3]$,

$$\mathbf{x}' = \mathbf{W}(\mathbf{x};\mathbf{b}_s) = \alpha\mathbf{x}'_1 + \beta\mathbf{x}'_2 + \gamma\mathbf{x}'_3. \tag{A.2}$$

Determining α,β and γ for a given $\mathbf{x} = (x, y)^\top$ is done by solving (A.1)[79],

$$\begin{aligned}
\alpha &= 1 - (\beta - \gamma) \\
\beta &= \frac{yx_3 - x_1y - x_3y_1 - y_3x + x_1y_3 + xy_1}{-x_2y_3 + x_2y_1 + x_1y_3 + x_3y_2 - x_3y_1 - x_1y_2} \\
\gamma &= \frac{xy_2 - xy_1 - x_1y_2 - x_2y + x_2y_1 + x_1y}{-x_2y_3 + x_2y_1 + x_1y_3 + x_3y_2 - x_3y_1 - x_1y_2}.
\end{aligned} \tag{A.3}$$

The warp $\mathbf{W}(\mathbf{x};\mathbf{b}_s)$ can be parameterized as,

$$\mathbf{W}(\mathbf{x};\mathbf{b}_s) = \begin{pmatrix} a1 + a2 \cdot x + a3 \cdot y \\ a4 + a5 \cdot x + a6 \cdot y \end{pmatrix}. \tag{A.4}$$

The parameters $(a_1, a_2, a_3, a_4, a_5, a_6)$ can be found from the relationship of two triangles, T_1 and T_2, with vertices denoted as (i, j, k) and $(1, 2, 3)$ respectively. Combining (A.1), (A.3) and (A.4) yields the values of the parameters,

$$
\begin{aligned}
a1 &= \frac{x_i + ((-x_1y_3 + x_3y_1 + x_1y_2 - x_2y_1)x_i + (x_1y_3 - x_3.*y_1)x_j + (-x_1y_2 + x_2y_1)x_k)}{-x_2y_3 + x_2y_1 + x_1y_3 + x_3y_2 - x_3y_1 - x_1y_2} \\
a2 &= \frac{((y_3 - y_2)x_i + (y_1 - y_3)x_j + (y_2 - y_1)x_k)}{-x_2y_3 + x_2y_1 + x_1y_3 + x_3y_2 - x_3y_1 - x_1y_2} \\
a3 &= \frac{((-x_3 + x_2)x_i + (x_3 - x_1)x_j + (-x_2 + x_1)x_k)}{-x_2y_3 + x_2y_1 + x_1y_3 + x_3y_2 - x_3y_1 - x_1y_2} \\
a4 &= \frac{y_i + (y_i(-x_1y_3 + x_3y_1 + x_1y_2 - x_2y_1) + (x_1y_3 - x_3y_1)y_j + y_k(-x_1y_2 + x_2y_1))}{-x_2y_3 + x_2y_1 + x_1y_3 + x_3y_2 - x_3y_1 - x_1y_2} \\
a5 &= \frac{(y_i(y_3 - y_2) + (y_1 - y_3)y_j + y_k(y_2 - y_1))}{-x_2y_3 + x_2y_1 + x_1y_3 + x_3y_2 - x_3y_1 - x_1y_2} \\
a6 &= \frac{(y_i(-x_3 + x_2) + (x_3 - x_1)y_j + y_k(-x_2 + x_1))}{-x_2y_3 + x_2y_1 + x_1y_3 + x_3y_2 - x_3y_1 - x_1y_2}.
\end{aligned} \tag{A.5}
$$

Appendix B

Bayesian Eye Tracking

B.1 Bayesian State Estimation

Bayesian methods provide a general framework for dynamic state estimation problems. The Bayesian approach is to construct the probability density function of the state based on all the available information.

Kalman filtering[94] finds the optimal solution given a linear problem with Gaussian distributed noise.

For nonlinear problems there are no analytic expression for the required pdf. The extended Kalman filter[6] linearizes about the predicted state. However, a more sophisticated approach is Particle filtering[6][31], which is a sequential Monte Carlo method. This is a generalization of the traditional Kalman filtering methods. A brief description is found in the following section B.1.1.

B.1.1 Particle Filtering

Let $\{\mathbf{x}^i_{0:k}, w^i_k\}^{N_s}_{i=1}$ denote a random measure characterizing the posterior pdf $p(\mathbf{x}_{0:k}|\mathbf{z}_{1:k})$, where $\{\mathbf{x}^i_{0:k}, i = 0, \ldots, N_s\}$ is a set of support points with associated weights $\{w^i_k, i = 1, \ldots, N_s\}$ and $\mathbf{x}_{0:k} = \{\mathbf{x}_j, j = 0, \ldots, k\}$ is the set of all states up to k. The weights are normalized so they sum to one. The true posterior density can then be approximated as a set of weighted samples,

$$p(\mathbf{x}_{0:k}|\mathbf{z}_{1:k}) \approx \sum_{i=1}^{N_s} w^i_k \delta(\mathbf{x}_{0:k} - \mathbf{x}^i_{0:k}). \tag{B.1}$$

The weights w^i_k are chosen using the principle of importance sampling.

$$w^i \propto \frac{p(\mathbf{x}^i_{0:k}|\mathbf{z}_{1:k})}{q(\mathbf{x}^i_{0:k}|\mathbf{z}_{1:k})}. \tag{B.2}$$

$$q(\mathbf{x}_{0:k}|\mathbf{z}_{1:k}) = q(\mathbf{x}_k|\mathbf{x}_{0:k-1}, \mathbf{z}_{1:k})q(\mathbf{x}_{0:k-1}|\mathbf{z}_{1:k-1}) \tag{B.3}$$

Using Bayes theorem the posterior distribution can be written as,

$$p(\mathbf{x}_{0:k}|\mathbf{z}_{1:k}) = \frac{p(\mathbf{z}_k|\mathbf{x}_{0:k}, \mathbf{z}_{1:k-1})p(\mathbf{x}_{0:k}|\mathbf{z}_{1:k-1})}{p(\mathbf{z}_k|\mathbf{z}_{1:k-1})}. \tag{B.4}$$

Using the sum rule,

$$p(\mathbf{x}_{0:k}|\mathbf{z}_{1:k-1}) = \sum p(\mathbf{x}_{0:k}|\mathbf{x}_{0:k-1}, \mathbf{z}_{1:k-1})p(\mathbf{x}_{0:k-1}|\mathbf{z}_{1:k-1}) \tag{B.5}$$

Inserting (B.5) in (B.4) yields,

$$\begin{aligned}
&p(\mathbf{x}_{0:k}|\mathbf{z}_{1:k}) \\
&\quad = \frac{p(\mathbf{z}_k|\mathbf{x}_{0:k}, \mathbf{z}_{1:k-1}) \sum p(\mathbf{x}_{0:k}|\mathbf{x}_{0:k-1}, \mathbf{z}_{1:k-1})p(\mathbf{x}_{0:k-1}|\mathbf{z}_{1:k-1})}{p(\mathbf{z}_k|\mathbf{z}_{1:k-1})} \\
&\quad = \frac{\sum p(\mathbf{z}_k|\mathbf{x}_{0:k}, \mathbf{z}_{1:k-1})p(\mathbf{x}_{0:k}|\mathbf{x}_{0:k-1}, \mathbf{z}_{1:k-1})}{p(\mathbf{z}_k|\mathbf{z}_{1:k-1})} p(\mathbf{x}_{0:k-1}|\mathbf{z}_{1:k-1}) \\
&\quad = \frac{p(\mathbf{z}_k|\mathbf{x}_{0:k}, \mathbf{z}_{1:k-1})p(\mathbf{x}_{0:k}|\mathbf{x}_{0:k-1}, \mathbf{z}_{1:k-1})}{p(\mathbf{z}_k|\mathbf{z}_{1:k-1})} \times p(\mathbf{x}_{0:k-1}|\mathbf{z}_{1:k-1}) \\
&\quad = \frac{p(\mathbf{z}_k|\mathbf{x}_k)p(\mathbf{x}_k|\mathbf{x}_{k-1})}{p(\mathbf{z}_k|\mathbf{z}_{1:k-1})} p(\mathbf{x}_{0:k-1}|\mathbf{z}_{1:k-1}) \\
&\quad \propto p(\mathbf{z}_k|\mathbf{x}_k)p(\mathbf{x}_k|\mathbf{x}_{k-1})p(\mathbf{x}_{0:k-1}|\mathbf{z}_{1:k-1}).
\end{aligned} \tag{B.6}$$

Inserting (B.3) and (B.6) into (B.2) the weight update equation follows,

$$\begin{aligned}
w_k^i &\propto \frac{p(\mathbf{z}_k|\mathbf{x}_k^i)p(\mathbf{x}_k^i|\mathbf{x}_{k-1}^i)p(\mathbf{x}_{0:k-1}^i|\mathbf{z}_{1:k-1})}{q(\mathbf{x}_k^i|\mathbf{x}_{0:k-1}^i, \mathbf{z}_{1:k})q(\mathbf{x}_{0:k-1}^i|\mathbf{z}_{1:k-1})} \\
&= w_{k-1}^i \frac{p(\mathbf{z}_k|\mathbf{x}_k^i)p(\mathbf{x}_k^i|\mathbf{x}_{k-1}^i)}{q(\mathbf{x}_k^i|\mathbf{x}_{0:k-1}^i, \mathbf{z}_{1:k})}
\end{aligned} \tag{B.7}$$

If the Markov property holds for q, $q(\mathbf{x}_k|\mathbf{x}_{0:k-1}, \mathbf{z}_{1:k}) = q(\mathbf{x}_k|\mathbf{x}_{k-1}, \mathbf{z}_k)$, then

$$w_k^i \propto w_{k-1}^i \frac{p(\mathbf{z}_k|\mathbf{x}_k^i)p(\mathbf{x}_k^i|\mathbf{x}_{k-1}^i)}{q(\mathbf{x}_k^i|\mathbf{x}_{k-1}^i, \mathbf{z}_k)} \tag{B.8}$$

B.2 Derivation of the Point Evaluation Function

The point evaluation function presented in section 14.4.3 is used for testing the hypothesis of the presence of a contour. This function is defined (14.17),

$$h(\mathcal{M}|\mu) = -\log(m) + \log \sum_j \frac{p_D(\epsilon_j)}{p_L(\Delta\mathcal{M}(j\Delta\nu))} \Delta\nu. \tag{B.9}$$

Using the definitions of the generalized Laplacian p_L (14.5), density of deformations p_D (14.10) and choosing the parameter $\beta = 0.5$, the point evaluation function above becomes,

$$\begin{aligned}
h(\mathcal{M}|\mu) &= -\log(m) + \log \sum_j \frac{\frac{1}{Z_D}\exp\left(\frac{-\epsilon_j^2}{2\sigma^2}\right)}{\frac{1}{Z_L}\exp\left(-\sqrt{\frac{|\Delta\mathcal{M}(j\Delta\nu)|}{L}}\right)}\Delta\nu \\
&= -\log(m) + \log\left(\frac{Z_L\Delta\nu}{Z_D}\right) + \log\sum_j \frac{\exp\left(\frac{-\epsilon_j^2}{2\sigma^2}\right)}{\exp\left(-\sqrt{\frac{|\Delta\mathcal{M}(j\Delta\nu)|}{L}}\right)} \\
&= \log\left(\frac{Z_L}{m}\right) + \log\left(\frac{\Delta\nu}{Z_D}\right) + \log\sum_j \frac{\exp\left(\frac{-\epsilon_j^2}{2\sigma^2}\right)}{\exp\left(-\sqrt{\frac{|\Delta\mathcal{M}(j\Delta\nu)|}{L}}\right)} \\
&= \log\left(\frac{Z_L}{m}\right) - \log\left(\frac{Z_D}{\Delta\nu}\right) + \log\sum_j \exp\left(\sqrt{\frac{|\Delta\mathcal{M}(j\Delta\nu)|}{L}}\right)\exp\left(\frac{-\epsilon_j^2}{2\sigma^2}\right) \\
&= h_0 + \log\sum_j \exp\left[\sqrt{\frac{|\Delta\mathcal{M}(j\Delta\nu)|}{L}} - \frac{\epsilon_j^2}{2\sigma^2}\right], \qquad \text{(B.10)}
\end{aligned}$$

where $h_0 = \log Z_L/m - \log Z_D/\Delta\nu$.

B.3 The EM Algorithm

The expectation-maximization algorithm is an iterative optimization method with purpose to find the model parameters $\mathbf{x}$ describing a set of observed data $\mathcal{M}$ and unobserved data ν^*. We wish to optimize the log likelihood of the parameters,

$$\begin{aligned}
\mathcal{L}(\mathbf{x}) &= \log p(\mathcal{M}|\mathbf{x}) \qquad \text{(B.11)} \\
&= \log \int p(\mathcal{M}, \nu^*|\mathbf{x})\mathrm{d}\nu^* \qquad \text{(B.12)} \\
&= \log \int q(\nu^*)\frac{p(\mathcal{M}, \nu^*|\mathbf{x})}{q(\nu^*)}\mathrm{d}\nu^*. \qquad \text{(B.13)}
\end{aligned}$$

By use of Jensen's inequality[13][29] for any distribution of hidden states $q(\nu^*)$, we have

$$\mathcal{L}(x) \geq \mathcal{F}(q, \mathbf{x}), \qquad \text{(B.14)}$$

where $\mathcal{F}(q, \mathbf{x})$ is a lower bound on the log likelihood.

$$\begin{aligned}
\mathcal{F}(q, \mathbf{x}) &= \int q(\nu^*) \log \frac{p(\mathcal{M}, \nu^* | \mathbf{x})}{q(\nu^*)} \mathrm{d}\nu^* && \text{(B.15)} \\
&= \int q(\nu^*) \log \frac{p(\nu^* | \mathcal{M}, \mathbf{x}) p(\mathcal{M} | \mathbf{x})}{q(\nu^*)} \mathrm{d}\nu^* && \text{(B.16)} \\
&= \int q(\nu^*) \log \frac{p(\nu^* | \mathcal{M}, \mathbf{x})}{q(\nu^*)} \mathrm{d}\nu^* + \log p(\mathcal{M} | \mathbf{x}). && \text{(B.17)}
\end{aligned}$$

The first term of (B.17) is always negative. Therefore, by comparing (B.17) and (B.11), it is easily seen that the expression $\mathcal{L}(\mathbf{x}) \geq \mathcal{F}(q, \mathbf{x})$ indeed holds.

The lower bound is equal to $\mathcal{L}(\mathbf{x})$ for $q(\nu^*) = p(\nu^* | \mathcal{M}, \mathbf{x})$. This is proven by inserting into (B.16),

$$\begin{aligned}
\mathcal{F}(\hat{q}, \mathbf{x})_{\hat{q}=p(\nu^* | \mathcal{M}, \mathbf{x})} &= \int p(\nu^* | \mathcal{M}, \mathbf{x}) \log \frac{p(\nu^* | \mathcal{M}, \mathbf{x}) p(\mathcal{M} | \mathbf{x})}{p(\nu^* | \mathcal{M}, \mathbf{x})} \mathrm{d}\nu^* && \text{(B.18)} \\
&= \int p(\nu^* | \mathcal{M}, \mathbf{x}) \log p(\mathcal{M} | \mathbf{x}) \mathrm{d}\nu^* && \text{(B.19)} \\
&= \log p(\mathcal{M} | \mathbf{x}) && \text{(B.20)} \\
&= \mathcal{L}(\mathbf{x}). && \text{(B.21)}
\end{aligned}$$

However, the distribution of hidden states cannot be obtained explicit. Consequently, local lower bounds $\mathcal{F}(q, \mathbf{x})$ are optimized alternately with respect to q and $\mathbf{x}$ while keeping the other fixed. This leads naturally to the iterative scheme for maximum-likelihood parameter estimation,

E step: Optimize $\mathcal{F}(q, \mathbf{x})$ with respect to the distribution over hidden data given the fixed parameters,

$$q^k(\nu^*) = \text{argmax } \mathcal{F}(q(\nu^*), \mathbf{x}^{k-1})$$

M step: Optimize $\mathcal{F}(q, \mathbf{x})$ with respect to the parameters given the fixed hidden distribution,

$$\mathbf{x}^k = \text{argmax } \mathcal{F}(q^k(\nu^*), \mathbf{x}) = \text{argmax } \int q^k(\nu^*) \log p(\mathcal{M}, \nu^* | \mathbf{x}) \mathrm{d}\nu^*,$$

which is equivalent to optimizing the complete expected data likelihood $p(\mathcal{M}, \nu^* | \mathbf{x})$, since q does not depend on $\mathbf{x}$.

This scheme is iterated until $\mathcal{F}(q, \mathbf{x})$ attains equality with $\mathcal{L}(\mathbf{x})$ or at least to some stop criteria is satisfied. At no iteration the likelihood can decrease.

B.4 EM Active Contour algorithm

The pseudo code of the presented EM active contour algorithm is presented below.

Initialization

- Load initial image
- Set initial particle $\mathbf{x}_0$ and state noise $\mathbf{v}_0$
- Choose number of particles N_s
- Draw initial particles distributed by $\mathcal{N}(\mathbf{x}_0, \mathbf{v}_0)$

Particle filtering Estimate the posterior density $p(\mathbf{x}_k|\mathcal{M}_{1:k})$

- **for** $i = 1 : N_s$
  - Propagate particles through $\mathbf{x}_k^i = \mathbf{f}_k(\mathbf{x}_{k-1}, \mathbf{v}_{k-1})$
  - Evaluate importance weights $w_k^i = p(\mathcal{M}_k|\mathbf{x}_k^i)$
    - Measure gray level differences (GLD) along the normal to each point on the contour μ
    - Evaluate hypothesis $h(\mathcal{M}|\mu)$ for each measurement line
    - Evaluate likelihood of each particle as the sum of hypothesis
    - The likelihood is coupled with priors regarding the probability of a present shape and intensity.
    - Assign the particle a weight w_k^i.
- **end**
- Normalization of weights, $\sum_{i=1}^{N_s} w_k^i = 1$
- Calculate effective particle set size $\widehat{N}_{eff}$
- **if** $\widehat{N}_{eff} < N_T$
  - Resample particles
- **end**
- $\mathbf{x}_k = \sum_{i=1}^{N_s} w_k^i \mathbf{x}_k^i$

Optimization by EM

- **while** norm$(\mu_k^j - \mu_k^{j-1}) <$ tol
  - E-step: Estimate the true contour location $\widehat{\nu}$ according to (14.25) given the image evidence and last estimate μ_k^{j-1}
  - M-step: Minimize the squared deformation (14.29) in a least squares sense to obtain the re-estimated contour location μ_k^j
  - j = j+1
- **end**

Appendix C

Heuristics for Speeding Up Gaze Estimation

During the six months master thesis period, a paper was prepared and accepted at the *Swedish Symposium on Image Analysis, Malmö, 10-11 march 2005* (SSBA 2005). As documentation of the workload herein, the paper is presented below. The paper recapitulates much of the work documented in the thesis.

Heuristics for speeding up gaze estimation

Denis Leimberg, Martin Vester-Christensen, Bjarne Kjær Ersbøll and Lars Kai Hansen
Department of Informatics and Mathematical Modelling
Technical University of Denmark
denis@kultvizion.dk, martin@kultvizion.dk, be@imm.dtu.dk, lkh@imm.dtu.dk

Abstract

A deformable template method for eye tracking on full face images is presented. The strengths of the method are that it is fast and retains accuracy independently of the resolution. We compare the method with a state of the art active contour approach, showing that the heuristic method is more accurate.

1 Introduction

Gaze is very important for human communication and also plays an increasing role for human computer interaction. Gaze can play a role, e.g., in understanding the emotional state for humans [1, 2], synthesizing emotions [3], and for estimation of attentional state [7]. Specific applications include devices for the disabled, e.g., using gaze as a replacement for a computer mouse and driver awareness monitoring to improve traffic safety [5].

It has been noted that the high cost of good gaze detection devices is a major road block for broader application of gaze technology, hence, there is a strong motivation for creating systems that are simple, inexpensive, and robust [4].

Figure 1: Examples of the dataset. The region surrounding the eyes can be found in various ways. We use a head tracking algorithm[5] based on Active Appearance Models. A subimage is extracted and subsequently processed by the eye tracking algorithms.

Detection of the human eye is a difficult task due to a weak contrast between the eye and the surrounding skin. As a consequence, many existing approaches use close-up cameras to obtain high-resolution images[4]. However, this imposes restrictions on head movements. Wang et al.[8] use a two camera setup to overcome the problem.

The present paper is inspired by the line of thinking mentioned above. We focus on some of the image processing issues. In particular we propose a robust algorithm for swift eye tracking in low-resolution video images. We compare this algorithm with a proven method[4] and relate the pixel-wise error to the precision of the gaze determination.

2 Deformable Template Matching

In many existing approaches the shape of the iris is modeled as a circle. This assumption is well-motivated when the camera pose coincides with the optical axis of the eye. When the gaze is off the optical axis, the circular iris is rotated in 3D space, and appears as an ellipse in the image plane. Thus, the shape of the contour changes as a function of the gaze direction and the camera pose. The objective is then to fit an ellipse to the pupil contour, which is characterized by a darker color compared to the iris. The ellipse is parameterized,

$$\mathbf{x} = (c_x, c_y, \lambda_1, \lambda_2, \theta)\,, \tag{1}$$

where (c_x, c_y) is the ellipse centroid, λ_1 and λ_2 are the lengths of the major and minor axis respectively. θ is the orientation of the ellipse.

The pupil region P is the part of the image I spanned by the ellipse parameterized by $\mathbf{x}$. The background region B is defined as the pixels inside an ellipse, surrounding but not included in P, as seen in figure 2. When region P contains the entire object, B must be outside the object, and thus the difference in average pixel intensity is maximal. To ensure equal weighting of the two regions, they have the same area.

The pupil contour can now be estimated by minimizing the cost function,

$$\mathcal{E} = \mathrm{Av}(P) - \mathrm{Av}(B), \tag{2}$$

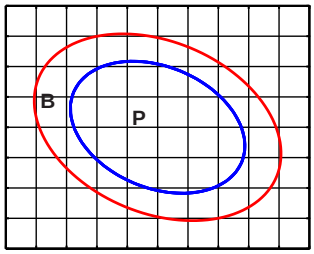


Figure 2: The deformable template model. Region P is the inner circle, and region B is the ring around it.

where $\mathrm{Av}(B)$ and $\mathrm{Av}(P)$ are the average pixel intensities of the background - in this case the iris - and pupil region respectively.

The model is deformed by Newton optimization given an appropriate starting point. Due to rapid eye movements[6], the algorithm may break down if one uses the previous state as initial guess of the current state, since the starting point may be too far from the true state. As a consequence, we use a simple 'double threshold' estimate of the pupil region as starting point.

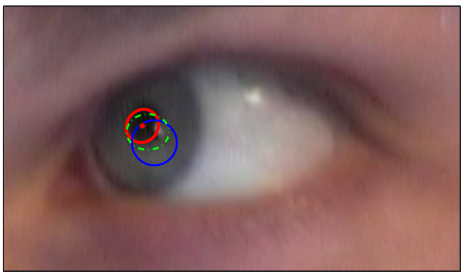


Figure 3: The blue ellipse indicates the starting point of the pupil contour. The template is iteratively deformed by an optimizer; one of the iterations is depicted in green. The red ellipse indicates the resulting estimate of the contour.

An example of the optimization of the deformable model is seen in figure 3.

2.1 Constraining the Deformation

Although a deformable template model is capable of tracking changes in the pupil shape, there are also some major drawbacks. Corneal reflections, caused by illumination, may confuse the algorithm and cause it to deform unnaturally. In the worst case the shape may grow or shrink until the algorithm collapses.

We propose to constrain the deformation of the model in the optimization step by adding a regularization term.

3 EM Contour Tracking

The iris is circular and characterized by a large contrast to the sclera. Therefore, it seems obvious to use a contour based tracker. Witzner et al.[4] describe an algorithm for tracking using active contours and particle filtering. A generative model is formulated which combines a dynamic model of state propagation and an observation model relating the contours to the image data. The current state is then found recursively by taking the sample mean of the estimated posterior probability.

The proposed method in this paper is based on [4], but extended with constraints and robust statistics.

3.1 The Dynamic Model

A dynamic model describes how the iris moves from frame to frame. Since the pupil movements are quite rapid at this time scale, the dynamics are modeled as Brownian motion (AR(1)),

$$\mathbf{x}_{t+1} = x_t + v_t, \quad v_t \sim \mathcal{N}(0, \mathbf{\Sigma}), \tag{3}$$

where $\mathbf{x}$ is the state from (1) and $\mathbf{\Sigma}$ is covariance matrix of the noise v_t.

3.2 The Observation Model

The observation model consists of two parts. A geometric component modeling the deformations of the iris by assuming a Gaussian distribution of all sample points along the contour. Secondly a texture component defining a pdf over pixel gray level differences given a contour location. Both components are joined and marginalized to produce a test of the hypothesis that there is a true contour present. The contour maximizing the combined hypotheses is chosen.

3.3 Active Contour Tracking

The tracking problem can be stated as a Bayesian inference problem by use of the recursive relation,

$$p(\mathbf{x}_{t+1}|\mathcal{M}_{t+1}) \propto p(\mathcal{M}_t|\mathbf{x}_t)p(\mathbf{x}_{t+1}|\mathcal{M}_t) \tag{4}$$

$$p(\mathbf{x}_{t+1}|\mathcal{M}_t) = \int p(\mathbf{x}_{t+1}|\mathbf{x}_t)p(\mathbf{x}_t|\mathcal{M}_t)\mathrm{d}\mathbf{x}_t \tag{5}$$

where $\mathcal{M}_t$ is the observations. Particle filtering is used to estimate the optimal state in a new frame.

3.4 Constraining the Hypotheses

We propose to weigh the hypotheses through a sigmoid function. This has the effect of decreasing the evidence when the inner part of the ellipse is brighter than the surroundings. An example is depicted in figure 4. In addition, this relaxes the importance of the hypotheses along the contour around the eyelids, which improves the fit.

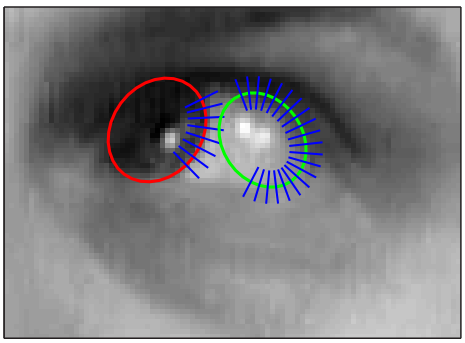


Figure 4: This figure illustrates the importance of the gray level constraint. Due to the general formulation of absolute gray level differences, the right contour has a greater likelihood, and the algorithm may thus fit to the sclera. Note the low contrast between iris and skin.

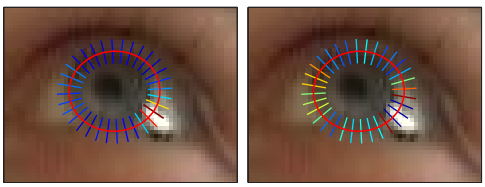


Figure 5: The relative normalized weighting of the hypotheses - Blue indicates low, while red indicates high scores. (*1*) Corneal reflections cause very distinct edges. Thus some hypotheses are weighted unreasonably high, which may confuse the algorithm. (*2*) This is solved by using robust statistics to remove outlying hypotheses.

3.5 Robust Statistics

By using robust statistics, hypotheses which obtain unreasonably high values compared to the others, are treated as outliers and therefore rejected, as seen in figure 5.

4 Results

A number of experiments have been performed with the proposed methods. We wish to investigate the importance of image resolution. Therefore the algorithms are evaluated on two datasets. One containing close up images, and one containing a down-sampled version hereof.

The algorithms estimate the center of the pupil. For each frame the error is recorded as the difference between a hand annotated ground truth and the output of the algorithms. This may lead to a biased result due to annotation error. However, this bias applies to all algorithms and a fair comparison can still be made.

Figure 6 and 7 depicts the error as a function of the number of particles used, for low resolution and high resolution images respectively. The errors for three different active contour(AC) algorithms are shows; basic, with EM refinement, with deformable template(DT) refinement. The error of the deformable template(DT) algorithm, initialized by double threshold, is inserted into the plot.

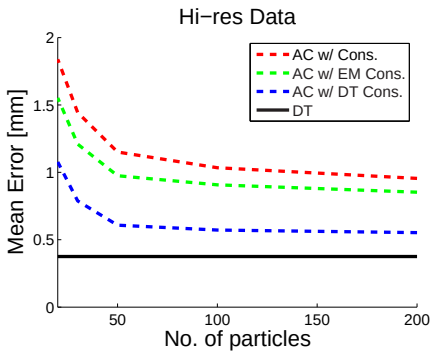


Figure 6: The error of the algorithms as a function of the number of particles for the high resolution data.

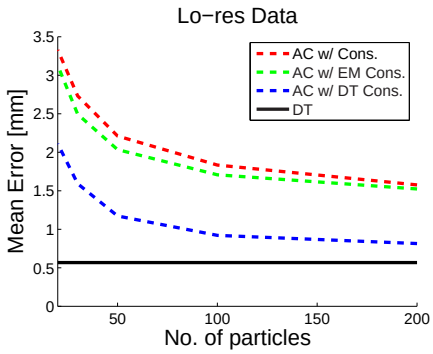


Figure 7: The error of the algorithms as a function of the number of particles for the low resolution data.

It can be seen that the proposed constraints on the active contour generally improves the accuracy of the fit. The refinement by the deformable template performs better than the EM method. The cost is an increased number of computations, which is resolution dependent. However, the deformable template method, initialized by double thresholding, is seen to outperform all active contour algorithms.

Hi-res	$E(x,y)$[mm]	$E(\theta)$	[frame/s]
AC	0.9	4.1	0.54
AC w/EM	0.8	3.7	0.49
AC w/DT	0.5	2.3	0.25
DT	0.3	1.4	2.2
Lo-res	$E(x,y)$[mm]	$E(\theta)$	[frame/s]
AC	1.5	7.3	0.57
AC w/EM	1.5	6.9	0.55
AC w/DT	0.8	3.7	0.49
DT	0.5	2.3	8.4

Table 1: Speed and precision comparison of the algorithms. The active contour uses 200 particles.

The table in figure 4 lists the mean error in accuracy in centimeters and degrees. Also listed is the computation time in frames per section of a Matlab implementation run on a 2.4Ghz PC. In general, the accuracy improves with high resolution as seen in table 4. However, the methods utilizing deformable template matching are less sensitive. The computation time for the basic active contour and EM refinement methods are independent of resolution. A significant increase in speed is noticed for the deformable template methods.

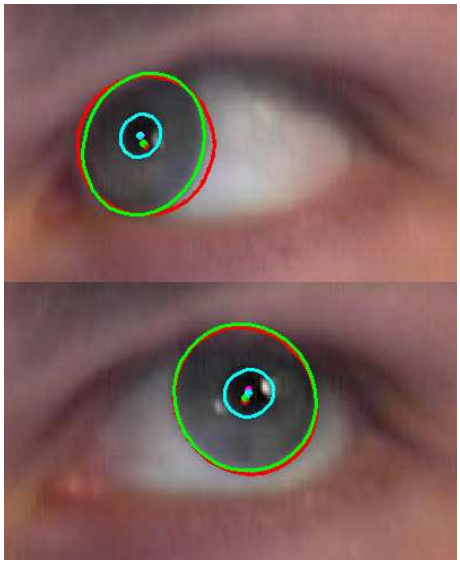


Figure 8: The resulting fit on two frames from a sequence - the red contour indicates the basic active contour, green indicates the EM refinement and the cyan indicates the deformable template initialized by the heuristic method. The top figure illustrates the benefit fitting to the pupil rather than the iris. Using robust statistic the influences from corneal reflections on the deformable template fit are ignored as depicted in the bottom image.

5 Conclusion

In this paper we have presented heuristics for improvement of the active contour method proposed by [4]. We have shown increased performance by using the prior knowledge that the iris is darker than its surroundings. This prevents the algorithm from fitting to the sclera as seen in figure 4.

Also presented is a novel approach to eye tracking based on a deformable template initialized by a simple heuristic. This enables the algorithm to overcome rapid eye movements. The active contour method handles these by broadening the state distribution and thus recovering the fit in a few frames. Furthermore, the accuracy is increased by fitting to the pupil rather than iris. This is particularly the case when a part of the iris is occluded as seen in figure 8.

It is shown that the deformable template model is accurate independent of resolution and it is very fast for low resolution images. This makes it useful for head pose independent eye tracking.

Acknowledgements

We wish to thank Hans Bruun Nielsen, Department of Informatics and Mathematical Modelling - Technical University of Denmark, for providing the optimization implementation. Additionally, we wish to thank Dan Witzner Hansen, IT-University of Copenhagen, for inspiring and insightful discussions. This research is supported by the IST Network of Excellence - Communication by Gaze Interaction (COGAIN).

References

[1] Jr. Adams, R.B. and R.E. Kleck. Perceived gaze direction and the processing of facial displays of emotion. *Psychological Science*, 2003.

[2] R.B. Jr. Adams, H.L. Gordon, A.A. Baird, N. Ambady, and R.E. Kleck. Effects of gaze on amygdala sensitivity to anger and fear faces. *Science*, 300:1536–1537, 2003.

[3] Jonathan Gratch and Stacy Marsella. Tears and fears: Modeling emotions and emotional behaviors in synthetic agents. *Proceedings of the 5th International Conference on Autonomous Agents, Montreal, Canada*, June 2001.

[4] Dan W. Hansen and Arthur E. C. Pece. Iris tracking with feature free contours. In *Proc. workshop on Analysis and Modelling of Faces and Gestures: AMFG 2003*, October 2003.

[5] Takahiro Ishikawa, Simon Baker, Iain Matthews, and Takeo Kanade. Passive driver gaze tracking with active appearance models. In *Proceedings of the 11th World Congress on Intelligent Transportation Systems*, October 2004.

[6] J. Pelz, R. Canosa, J. Babcock, D. Kucharczyk, A. Silver, and D. Konno. Portable eyetracking: A study of natural eye movements, 2000.

[7] Rainer Stiefelhagen, Jie Yang, and Alex Waibel. Estimating focus of attention based on gaze and sound. In *PUI '01: Proceedings of the 2001 workshop on Perceptive user interfaces*, pages 1–9. ACM Press, 2001.

[8] J.G. Wang and E. Sung. Study on eye gaze estimation. *IEEE Transactions on Systems, Man and Cybernetics - Part B: Cybernectics*, 32(3):332–350, June 2002.

Appendix D

Towards Emotion Modeling

Besides the presented paper in the previous chapter, an additional paper was submitted and accepted at the *HCI International 2005*, Las Vegas, 22-27 July 2005 (HCII2005). As documentation of the workload herein, the paper is presented below. The paper recapitulates much of the work documented in the thesis.

Towards emotion modeling based on gaze dynamics in generic interfaces

Martin Vester-Christensen, Denis Leimberg, Bjarne Kjær Ersbøll, and Lars Kai Hansen

Informatics and Mathematical Modelling, Technical University of Denmark, Building 321
DK-2800 Kgs. Lyngby, Denmark
Vester-Christensen@cogain.org, Denis.Leimberg@cogain.org,
Bjarne.Ersboell@cogain.org, Lars.Kai.Hansen@cogain.org

Abstract

Gaze detection can be a useful ingredient in generic human computer interfaces if current technical barriers are overcome. We discuss the feasibility of concurrent posture and eye-tracking in the context of single (low cost) camera imagery. The ingredients in the approach are posture and eye region extraction based on active appearance modeling and eye tracking using a new fast and robust heuristic. The eye tracker is shown to perform well for low resolution image segments, hence, making it feasible to estimate gaze using a single generic camera.

1 Introduction

We are interested in understanding human gaze dynamics and the possible applications of gaze dynamics models in human computer interfaces. We focus on gaze detection in the context of wide audience generic interfaces such as camera equipped multimedia PCs.

Gaze can play a role, e.g., in understanding the emotional state for humans (Adams & Kleck, 2003; Adams, Gordon, Baird, Ambady & Kleck, 2003), synthesizing emotions (Gratch & Marsella, 2001), and for estimation of attentional state (Stiefelhagen, Yang & Waibel, 2001). Gaze detection based interfaces may also be used for the disabled as a tool for generating emotional statements. The emotional state is a strong determinant for human behavior, hence, efficient estimators of emotion state are useful for many aspects of computing with humans. Emotion detection can be used to control adaptive interfaces and synthesized emotions may be used to transmit emotional context in an interface.

It has been noted that the high cost of state of the art gaze detection devices is a major road block for broader application of gaze technology, hence, there is a strong motivation for creating systems that are simple, inexpensive, and robust (Hansen & Pece, 2003). Relative low cost may be obtained using electro-oculography (EOG) (Kaufman, Bandopadhay & Shaviv, 1993), however, in many generic interfaces electrode based measures are infeasible, hence we will here focus on `non-invasive' measures obtained from visual data as in Figure 1.

Gaze detection consists of two related algorithmic steps, posture estimation and eye tracking. The posture is used to nail the head degrees of freedom and to locate the eye regions. In combination with eye tracking posture can be used to infer the gaze direction.

Detection of the human eye is a relatively complex task due to a weak contrast between the eye and the surrounding skin. As a consequence, many existing approaches use close-up cameras to obtain high-resolution images (Hansen & Pece, 2003). However, this imposes restrictions on head movements. Wang & Sung (2002) use a two camera setup to overcome the problem. We here focus on some of the image processing issues. In particular we discuss the posture estimation within the framework of active appearance models (AAM) and we discuss a recently proposed robust and swift eye tracking scheme for low-resolution video images (Leimberg, Vester-Christensen, Ersbøll & Hansen, 2005). We compare this algorithm with an existing method (Hansen & Pece, 2003) and relate the pixel-wise error to the precision of the gaze determination.

The authors all participate in the Network of Excellence: “Communication by Gaze Interaction” (COGAIN http://www.cogain.org) supported by the EU IST 6th framework program. Currently 20 partners participate in the network of excellence; who’s objective is to improve the quality of life for those impaired by motor-control disorders. The users should be able to use applications that help them to be in control of the environment, or achieve

Figure 1: We are interested in understanding gaze dynamics in the context of imagery from a single generic camera. The eye regions are obtained within a head tracking algorithm (Ishikawa, Baker, Matthews & Kanade, 2004) based on an active appearance model. Subimages are extracted and subsequently processed by eye tracking algorithms.

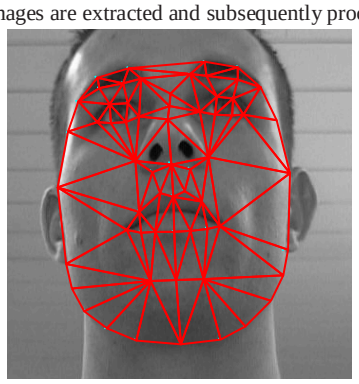


Figure 2: Face image of a face annotated with 58 landmarks for active appearance modeling.

a completely new level of convenience and speed in gaze-based communication. The goal is to have a solution based on standard PC technology. This will facilitate universal access and e-inclusion.

2 Head modeling using Active Appearance Modeling

Active appearance models combine information about shape and texture. In (Cootes, 2004) *shape* is defined as "... that quality of a configuration of points which is invariant under some transformation." Here a face shape consists of n 2D points, *landmarks*, spanning a 2D mesh over the object in question. The landmarks are either placed in the images automatically (Baker, Matthews & Schneider, 2004) or by hand. Figure 2 shows an image of a face (Stegmann, Ersbøll & Larsen, 2003) with the annotated shape shown as a red dots. Mathematically the shape $\mathbf{s}$ is defined as the $2n$-dimensional vector of coordinates of the n landmarks making up the mesh,

$$\mathbf{s} = [x_1, x_2, \ldots, x_n, y_1, y_2, \ldots, y_n]^T . \quad (1)$$

Given N annotated training examples, we have N such shapevectors $\mathbf{s}$, all subject to some transformation. In 2D the transformations considered are the similarity transformations (rotation, scaling and translation). We wish to obtain a model describing the inter-shape relations between the examples, and thus we must remove the variation

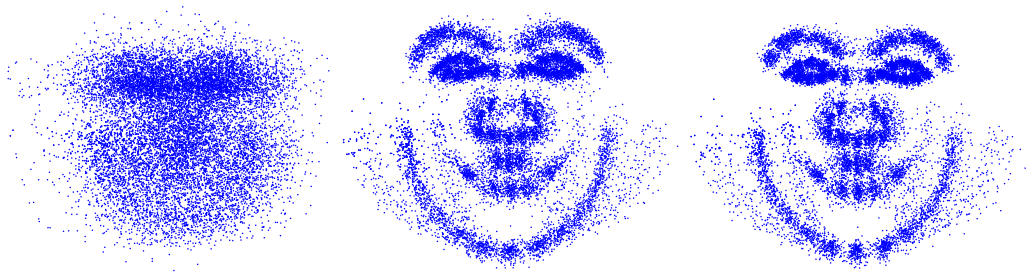


Figure 3: Procrustes analysis. The left figure shows all landmark points plotted on top of each other. The center figure shows the shapes after translation of their centers of mass, and normalization of the vector norm. The right figure is the result of the iterative Procrustes alignment algorithm.

given by this transformation. This is done by aligning the shapes in a common coordinate frame as described in the next section.

To remove the transformation, ie. the rotation, scaling and translation of the annotated shapes, they are aligned using iterative Procrustes analysis (Cootes, 2004). Figure 3 shows the steps of the iterative Procrustes analysis. The top figure shows all the landmarks of all the shapes plotted on top of each other. The lower left figure shows the initialization of the shape by the translation of their centers of mass and normalization of the norm of the shape vectors. The lower right figure is the result of the iterative Procrustes algorithm.

The normalization of the shapes and the following Procrustes alignment results in the shapes lying on a unit hypersphere. Thus the shape statistics will have to be calculated on the surface of this sphere. To overcome this problem the approximation that the shapes lie on the tangent plane to the hypersphere is made, and ordinary statistics can be used. The shape $\mathbf{s}$ can be projected onto the tangent plane using:

$$\mathbf{s}' = \frac{\mathbf{s}}{\mathbf{s}^T \mathbf{s}_0}, \tag{2}$$

where $\mathbf{s}$ is the estimated mean shape given from the Procrustes alignment.

With the shapes aligned in a common coordinate frame it is now possible to build a statistical model of the shape variation in the training set.

The result of the Procrustes alignment is a set of $2n$ dimensional shape vectors $\mathbf{s}_i$ forming a distribution in the space in which they live. In order to generate shapes, a parameterized model of this distribution is needed. Such a model is of the form $\mathbf{s} = M(\mathbf{b})$, where $\mathbf{b}$ is a vector of parameters of the model. If the distribution of parameters $p(\mathbf{b})$ can be modeled, constraints can be put on them such that the generated shapes $\mathbf{s}$ are similar to that of the training set. With a model it is also possible to calculate the probability $p(\mathbf{s})$ of a new shape.

To constitute a shape, neighboring landmark points must move together in some fashion. Thus some of the landmark points are correlated and the true dimensionality may be much less than $2n$. Principal Component Analysis(PCA) rotates the $2n$ dimensional data cloud that constitutes the training shapes. It maximizes the variance and gives the main axis of the data cloud.

The PCA is performed as an eigenanalysis of the covariance matrix, Σ, of the training data.

$$\Sigma = \frac{1}{N-1} \mathbf{S}\mathbf{S}^T, \tag{3}$$

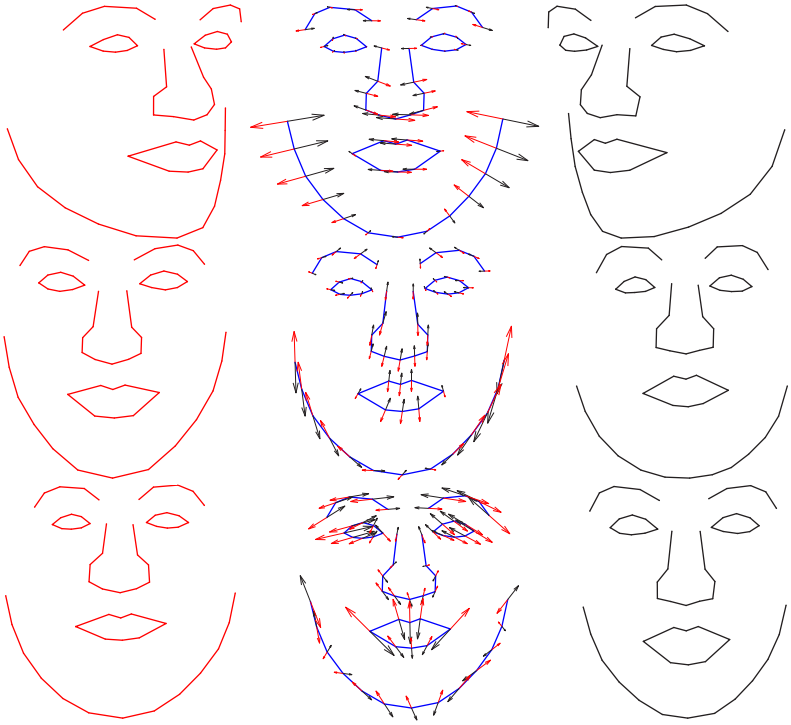


Figure 4: Mean shape deformation using first, second and third principal mode. The middle shape is the mean shape, the left column is minus two standard deviations corresponding to $b_{s_i} = -2\lambda$, the right is plus two standard deviations given by $b_{s_i} = 2\lambda$. The arrows overlain the mean shape indicates the direction and magnitude of the deformation corresponding to the parameter values.

where N is the number of training shapes, and $\mathbf{S}$ is the $n \times N$ matrix $\mathbf{D} = [\mathbf{s}_1 - \mathbf{s}_0, \mathbf{s}_2 - \mathbf{s}_0, \ldots, \mathbf{s}_N - \mathbf{s}_0]$. Σ is an $n \times n$ matrix. Eigenanalysis of the Σ matrix gives a diagonal matrix Λ_l of eigenvalues λ_i and a matrix Φ_l with eigenvectors ϕ_i as columns. The eigenvalues are equal to the variance in the eigenvector direction.

PCA can be used as a dimensionality reduction tool by projecting the data onto a subspace which fulfills certain requirements, for instance retaining 95% of the total variance or similar. Then only the eigenvectors corresponding to the t largest eigenvalues fulfilling the requirements are retained. This enables us to approximate a training shape instance $\mathbf{s}$ as a deformation of the mean shape by a linear combination of t shape eigenvectors,

$$\mathbf{s} \approx \mathbf{s}_0 + \Phi_\mathbf{s} \mathbf{b}_\mathbf{s}, \tag{4}$$

where $\mathbf{b}_\mathbf{s}$ is a vector of t *shape parameters* given by

$$\mathbf{b}_\mathbf{s} = \Phi_s^T (\mathbf{s} - \mathbf{s}_0), \tag{5}$$

and $\Phi_\mathbf{s}$ is the matrix with the t largest eigenvectors as columns.

A synthetic shape $\mathbf{s}$ is created as deformation of the mean shape $\mathbf{s}_0$ by a linear combination of the shape eigenvectors $\Phi_\mathbf{s}$,

$$\mathbf{s} = \mathbf{s}_0 + \Phi_s \mathbf{b}_s, \tag{6}$$

where $\mathbf{b}_s$ is the set of shape parameters. Specific facial expression may be learnt from examples in the AAM representation and re-synthesized by AAM simulation. Figure 4 shows three rows of shapes indicating the flexibility of the representation. The middle row is the mean shape. The left and right rows are synthesized shapes generated by deformation of the mean shape by $\pm 2\sqrt{\lambda_i}$.

In order to track moving faces, the AAM must be re-estimated for each frame. The objective is then to find the optimal set of parameters $\mathbf{b}_s$ and $\mathbf{b}_g$ such that the model instance $T(\mathbf{W}(\mathbf{x}, \mathbf{b}_s))$ is as similar as possible to the object in the image. An obvious way to measure the success of the fit is to calculate the error between the image and the model instance. An efficient way to calculate this error is to use the coordinate frame defined by the mean shape $\mathbf{s}_0$. Thus a pixel with coordinate $\mathbf{x}$ in $\mathbf{s}_0$ has a corresponding pixel in the image $\mathbf{I}$ with coordinate $\mathbf{W}(\mathbf{x}, \mathbf{b}_s)$ as described previously. The error of the fit can then be calculated as the difference in pixel values of the model instance and the image:

$$\mathbf{f}(\mathbf{b}_s, \mathbf{b}_g) = (\mathbf{g}_0 + \Phi_g \mathbf{b}_g) - \mathbf{I}(\mathbf{W}(\mathbf{x}, \mathbf{b}_s)), \tag{7}$$

This is a function in the texture parameters $\mathbf{b}_g$ and the shape parameters $\mathbf{b}_s$. A cost function can be defined as,

$$\mathbf{F}(\mathbf{b}_s, \mathbf{b}_g) = \left\| \mathbf{g}_0 + \Phi_g \mathbf{b}_g - \mathbf{I}(\mathbf{W}(\mathbf{x}, \mathbf{b}_s)) \right\|^2. \tag{8}$$

The optimal solution to (8) can be found as,

$$(\mathbf{b}_s^*, \mathbf{b}_g^*) = \arg\min_{\mathbf{b}_s, \mathbf{b}_g} \mathbf{F}. \tag{9}$$

Solving this, is in general a non-linear least squares problem, but fortunately there exist well-proven algorithms (Tingleff, Madsen & Nielsen 2004) for this step. The optimal shape can then be used for posture estimation and for locating the eye region, see Figure 5.

3 Eye tracking based on deformable template matching

In many existing approaches the shape of the iris is modeled as a circle. This assumption is well-motivated when the camera pose coincides with the optical axis of the eye. When the gaze is off the optical axis, the circular iris is rotated in 3D space, and appears as an ellipse in the image plane. Thus, the shape of the contour changes as a function of the gaze direction and the camera pose. The objective is then to fit an ellipse to the pupil contour, which is characterized by a darker color compared to the iris. The ellipse is parameterized,

$$\mathbf{x} = (c_x, c_y, \lambda_1, \lambda_2, \theta), \tag{10}$$

where (c_x, c_y) is the ellipse centroid, λ_1 and λ_2 are the lengths of the major and minor axis respectively. θ is the orientation of the ellipse.

The pupil region P is the part of the image I spanned by the ellipse parameterized by $\mathbf{x}$. The background region B is defined as the pixels inside an ellipse, surrounding but not included in P, as seen in Figure 6. When region P contains the entire object, B must be outside the object, and thus the difference in average pixel intensity is maximal. To ensure equal weighting of the two regions, they have the same area.

The pupil contour can now be estimated by minimizing the cost function,

$$\xi = \mathrm{Av}(P) - \mathrm{Av}(B), \tag{11}$$

where $\mathrm{Av}(B)$ and $\mathrm{Av}(P)$ are the average pixel intensities of the background - in this case the iris - and pupil region respectively.

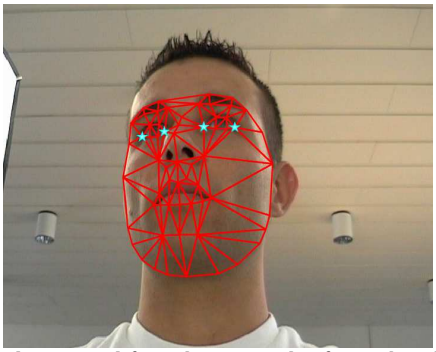


Figure 5: The eye images are easily extracted from the input video frames based on the fit of AAM. Each eye is modeled by six vertices. A bounding box containing the eye is easily extracted, by choosing a region slightly larger than the modeled eye.

The model is deformed by Newton optimization given an appropriate starting point. Due to rapid eye movements (Pelz et al., 2000}, the algorithm may break down if one uses the previous state as initial guess of the current state, since the starting point may be too far from the true state. As a consequence, we use a simple adaptive `double threshold' estimate (Sonka, M., Hlavac and Boyle, R., 1998) of the pupil region as starting point.

An example of the optimization of the deformable model is seen in Figure 7.

Although a deformable template model is capable of tracking changes in the pupil shape, there are also some major drawbacks. Corneal reflections, caused by illumination, may confuse the algorithm and cause it to deform unnaturally. In the worst case the shape may grow or shrink until the algorithm collapses. We propose to constrain the deformation of the model in the optimization step by adding a regularization term.

The iris is circular and is characterized by a large contrast to the sclera. Therefore, it seems obvious to use a contour based tracker. Hansen & Pece (2003) describe an algorithm for tracking using active contours and particle filtering. A generative model is formulated which combines a dynamic model of state propagation and an observation model relating the contours to the image data. The current state is then found recursively by taking the sample mean of the estimated posterior probability. The proposed method in this paper is based on Hansen & Pece (2003), but extended with constraints and robust statistics.

A dynamical model describes how the iris moves from frame to frame. Since the pupil movements are quite rapid at this time scale, the dynamics are modeled as Brownian motion (AR(1)),

$$\mathbf{x}_{t+1} = \mathbf{x}_t + \mathbf{v}_t, \qquad \mathbf{v}_t \sim \mathrm{N}(\mathbf{0}, \Sigma), \tag{12}$$

where $\mathbf{x}$ is the state from (10) and Σ is covariance matrix of the noise $\mathbf{v}_t$.

The observation model consists of two parts. A geometric component modeling the deformations of the iris by assuming a Gaussian distribution of all sample points along the contour. Secondly a texture component defining a pdf over pixel gray level differences given a contour location. Both components are joined and marginalized to produce a test of the hypothesis that there is a true contour present. The contour maximizing the combined hypotheses is chosen.

The tracking problem can be stated as a Bayesian inference problem by use of the recursive relation,

$$p(\mathbf{x}_{t+1}|M_{t+1}) \propto p(M_t|\mathbf{x}_t)\, p(\mathbf{x}_{t+1}|M_t), \tag{13}$$

$$p(\mathbf{x}_{t+1}|M_t) = \int p(\mathbf{x}_{t+1}|\mathbf{x}_t)\, p(\mathbf{x}_t|M_t)\mathrm{d}\mathbf{x}_t, \tag{14}$$

where M_t is the observations. Particle filtering is used to estimate the optimal state in a new frame.

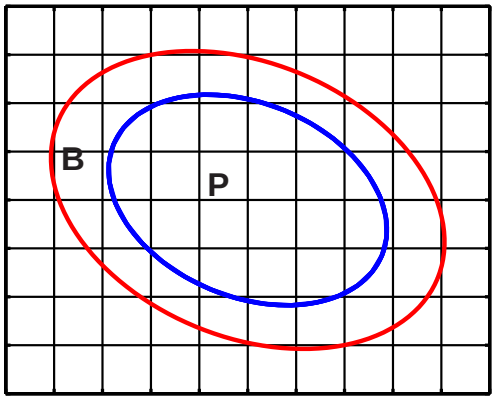


Figure 6: The deformable template model. Region P is the inner 'pupil' area, and region B is the outer 'background' area. These regions are deformed iteratively to maximize contrast between the regions.

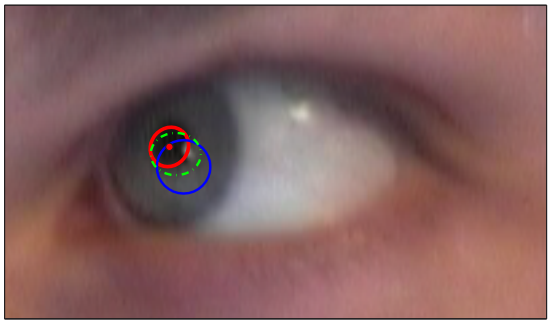


Figure 7: The blue ellipse indicates the starting point of the pupil contour. The template is iteratively deformed by an optimizer; one of the iterations is depicted in green. The red ellipse indicates the resulting estimate of the contour.

We propose to weigh the hypotheses through a sigmoid function. This has the effect of decreasing the evidence when the inner part of the ellipse is brighter than the surroundings. An example is depicted in Figure 8. In addition, this relaxes the importance of the hypotheses along the contour around the eyelids, which improves the fit.

By using robust statistics, hypotheses which obtain unreasonably high values compared to the others, are treated as outliers and therefore rejected, as seen in Figure 9.

3.1 Eye tracking results

A number of experiments have been performed with the proposed methods. We wish to investigate the importance of image resolution. Therefore the algorithms are evaluated on two datasets. One containing close up images, and one containing a down-sampled version hereof.

The algorithms estimate the center of the pupil. For each frame the error is recorded as the difference between a hand annotated ground truth and the output of the algorithms. This may lead to a biased result due to annotation error. However, this bias applies to all algorithms and a fair comparison can still be made.

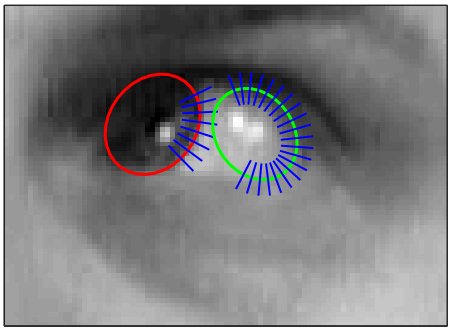


Figure 8: This figure illustrates the importance of the gray level constraint. Due to the general formulation of absolute gray level differences, the right contour has a greater likelihood, and the algorithm may thus fit to the sclera. Note the low contrast between iris and skin.

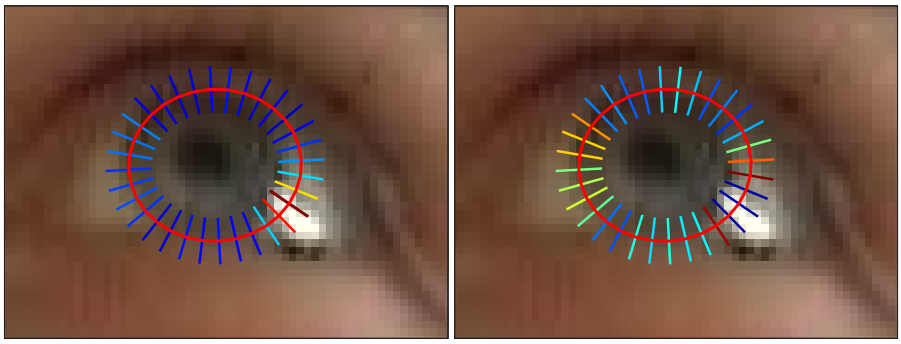


Figure 9: The relative normalized weighting of the hypotheses - Blue indicates low, while red indicates high scores. (*1*) Corneal reflections cause very distinct edges. Thus some hypotheses are weighted unreasonably high, which may confuse the algorithm. (*2*) This is solved by using robust statistics to remove outlying hypotheses.

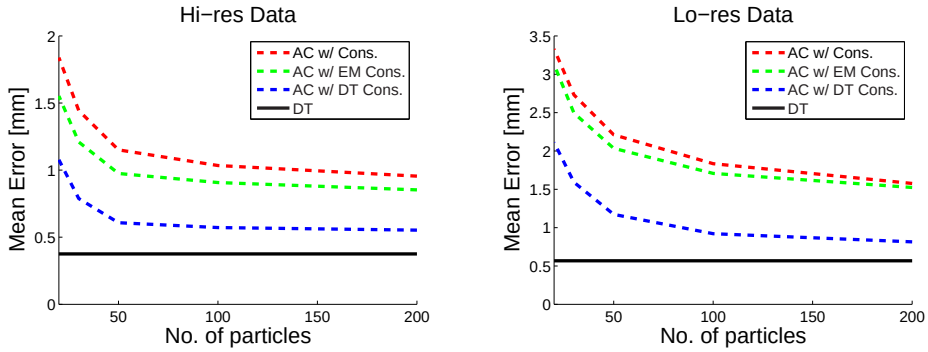


Figure 10: The error of the algorithms as a function of the number of particles for the high (left) and low (right) resolution data. The errors for three different active contour(AC) algorithms are shown; basic, with EM refinement, and with deformable template(DT) refinement of the mean; and for the the deformable template(DT) algorithm, initialized by double threshold.

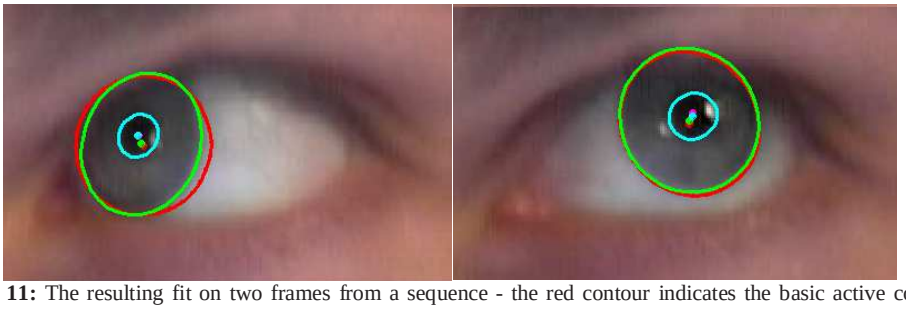


Figure 11: The resulting fit on two frames from a sequence - the red contour indicates the basic active contour, green indicates the EM refinement and the cyan indicates the deformable template initialized by the heuristic method. The left image illustrates the benefit fitting to the pupil rather than the iris. Using robust statistic the influences from corneal reflections on the deformable template fit are ignored as depicted in the right image.

Figure 10 depicts the error as a function of the number of particles used, for low resolution and high resolution images respectively. The errors for three different active contour (AC) algorithms are shown; basic, with EM refinement, and with deformable template (DT) refinement; and for the the deformable template (DT) algorithm, initialized by double threshold.It can be seen that the proposed constraints on the active contour generally improves the accuracy of the fit. The refinement by the deformable template performs better than the EM method. The cost is an increased number of computations, which is resolution dependent. However, the deformable template method, initialized by double thresholding, is seen to outperform all active contour algorithms.Table 1 lists the mean error in accuracy in centimetres and degrees. Also listed is the computation time in frames per section of a Matlab implementation run on a 2.4Ghz PC. In general, the accuracy improves with high resolution as seen in Table 1. However, the methods utilizing deformable template matching are less sensitive. The computation time for the basic active contour and EM refinement methods are independent of resolution. A significant increase in speed is noticed for the deformable template methods. Suppose, a geometric model of the eye is available (Ishikawa, Baker, Matthews, and Kanade , 2004), the gaze direction can be computed as a simple transformation of the 2D pupil center coordinates, to a 3D direction in space.

4 Conclusion

In this paper we have presented heuristics for improvement of the active contour method proposed by Hansen & Pece (2003). We have shown increased performance by using the prior knowledge that the iris is darker than its surroundings. This prevents the algorithm from fitting to the sclera as seen in Figure 8. Also presented is a novel approach to eye tracking based on a deformable template initialized by a simple heuristic. This enables the algorithm to overcome rapid eye movements. The active contour method handles these by broadening the state distribution and thus recovering the fit in a few frames. Furthermore, the accuracy is increased by fitting to the pupil rather than iris. This is particularly the case when a part of the iris is occluded as seen in Figure 11. It was shown that the deformable template model is accurate independent of resolution and it is very fast for low resolution images. In conclusion we have demonstrated that it is feasible to estimate gaze from a single generic camera. This opens for a multitude of new applications in human computer interfaces. By estimation of gaze dynamics we may detect emotional state. Emotion synthesis is feasible through the appearance model which can simulate faces with emotional expressions and given gaze.

5 Acknowledgments

We wish to thank Hans Bruun Nielsen, Department of Informatics and Mathematical Modelling - Technical University of Denmark, for providing the optimization implementation. Additionally, we wish to thank Dan Witzner Hansen, IT-University of Copenhagen and Mikkel B. Stegmann, Informatics and Mathematical Modelling for inspiring and insightful discussions.

The COGAIN Network of Excellence is supported by the EU IST 6th framework program. The authors gratefully acknowledge the support from the European Commission in order to be able to present this work.

Hi-res	$E(x,y)$ [mm]	$E(\theta)$	[frame/s]	Lo-res	$E(x,y)$ [mm]	$E(\theta)$	[frame/s]
AC	0.9	4.1	0.54	AC	1.5	7.3	0.57
AC w/EM	0.8	3.7	0.49	AC w/EM	1.5	6.9	0.55
AC w/DT	0.5	2.3	0.25	AC w/DT	0.8	3.7	0.49
DT	0.3	1.4	2.2	DT	0.5	2.3	8.4

Table 1: Speed and precision comparison of the algorithms. The active contour uses 200 particles. The errors for listed for three different active contour (AC) algorithms (basic, with EM refinement, and with deformable template (DT) refinement and for the the deformable template (DT) algorithm, initialized by double threshold.

References

Adams, R.B.Jr. & Kleck, R.E. Perceived gaze direction and the processing of facial displays of emotion. *Psychological Science*, 2003.

Adams, R.BJr., Gordon, H.L., Baird, A.A., Ambady, N. & and Kleck, R.E. Effects of gaze on amygdala sensitivity to anger and fear faces. *Science*, 300:1536–1537, 2003.

Baker, S., Matthews, I. & Schneider, J. Automatic construction of active appearance models as an image coding problem. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(10), October 2004.

Cootes, T. Statistical models of appearance for computer vision. A technical report available from http://www.isbe.man.ac.uk/ bim/Models/app_models.pdf, 2004.

Gratch, J. & Marsella, S. Tears and fears: Modeling emotions and emotional behaviors in synthetic agents. *Proceedings of the 5th International Conference on Autonomous Agents, Montreal, Canada*, June 2001.

Hansen, D.W. & Pece, A.E.C. Iris tracking with feature free contours. In *Proc. workshop on Analysis and Modelling of Faces and Gestures: AMFG 2003*, October 2003.

Ishikawa, T., Baker, S., Matthews, I. & Kanade, T. Passive driver gaze tracking with active appearance models. In *Proceedings of the 11th World Congress on Intelligent Transportation Systems*, October 2004.

Tingleff, O., Madsen, K. & Nielsen, H.B. Methods for Non-linear Least Squares Problems. Lecture Note in 02611 Optimization and Data Fitting, 2004.

Kaufman, A.E., Bandopadhay, A. & Shaviv, B.D. An eye tracking computer user interface. In *Proceedings of 1993 IEEE Research Properties in Virtual Reality Symposium*, pages 120–121, 1993.

Leimberg, D., Vester-Christensen, M., Ersbøll, B.K. & Hansen, L.K. Heuristics for speeding up gaze estimation. In *Proc. Svenska Symposium i Bildanalys, SSBA 2005, Malmø, Sweden, SSBA, 2005*, To appear, 2005.

Pelz, J., Canosa, R., Babcock, J., Kucharczyk, D., Silver, A. & Konno, D. Portable eyetracking: A study of natural eye movements, 2000.

Sonka, M., Hlavac & Boyle, R. Image Processing, Analysis and Machine Vision, 2.Edition. International Thomson Publishing, 1998

Stegmann, M.B., Ersbøll, B.K. & Larsen, R. FAME – a flexible appearance modelling environment. *IEEE Trans. on Medical Imaging*, 22(10):1319–1331, 2003.

Stiefelhagen, R., Yang, J. & Waibel, A. Estimating focus of attention based on gaze and sound. In *PUI '01: Proceedings of the 2001 workshop on Perceptive user interfaces*, pages 1–9. ACM Press, 2001.

Wang, J.G. & Sung, E. Study on eye gaze estimation. *IEEE Transactions on Systems, Man and Cybernetics - Part B: Cybernectics*, 32(3):332–350, June 2002.

Bibliography

[1] 100fps. http://www.100fps.com.

[2] Henrik Aanæs, Rune Fisker, Kalle Åström, and Jens Michael Carstensen. Robust factorization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(9):1215–1225, September 2002.

[3] Jr. Adams, R.B. and R.E. Kleck. Perceived gaze direction and the processing of facial displays of emotion. *Psychological Science*, 2003.

[4] R.B. Jr. Adams, H.L. Gordon, A.A. Baird, N. Ambady, and R.E. Kleck. Effects of gaze on amygdala sensitivity to anger and fear faces. *Science*, 300:1536–1537, 2003.

[5] Media Analyzer. http://www.mediaanalyzer.net/.

[6] S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp. A tutorial on particle filters for on-line non-linear/non-gaussian bayesian tracking. *IEEE Transactions on Signal Processing*, 50(2):174–188, February 2002.

[7] Sileye Ba and Jean-Marc Odobez. A Probabilistic Framework for Joint Head Tracking and Pose Estimation. IDIAP-RR 78, IDIAP, Martigny, Switzerland, 2003.

[8] A.M. Bagci, R. Ansari, A. Khokhar, and E. Cetin. Eye tracking using markov models. In *ICPR04*, pages III: 818–821, 2004.

[9] Simon Baker, Ralph Gross, and Iain Matthews. Lucas-kanade 20 years on: A unifying framework: Part 3. Technical Report CMU-RI-TR-03-35, Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, November 2003.

[10] Simon Baker, Ralph Gross, and Iain Matthews. Lucas-kanade 20 years on: A unifying framework: Part 4. Technical Report CMU-RI-TR-04-14, Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, February 2004.

[11] Simon Baker and Iain Matthews. Lucas-kanade 20 years on: A unifying framework. *International Journal of Computer Vision*, 56(3):221 – 255, March 2004.

[12] Simon Baker, Iain Matthews, and Jeff Schneider. Automatic construction of active appearance models as an image coding problem. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(10), October 2004.

[13] C. Bishop. *Neural Networks for Pattern Recognition*. Oxford University Press, 1995.

[14] BrainyDictionary. http://www.brainydictionary.com/.

[15] M. Brand. Morphable models from video. In *IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2001.

[16] Jens Michael Carstensen. *Image analysis, vision and computer graphics*. Technical University of Denmark, Kgs. Lyngby, 2 edition, 2002.

[17] Daniel Cheng and Roel Vertegaal. An eye for an eye: a performance evaluation comparison of the lc technologies and tobii eye trackers. In *Proceedings of the Eye tracking research & applications symposium on Eye tracking research & applications*, pages 61–61. ACM Press, 2004.

[18] K.N. Choi, M. Carcassoni, and E.R. Hancock. Estimating 3d facial pose using the em algorithm. In *BMVC98*, pages xx–yy, 1998.

[19] Knut Conradsen. *En Introduktion til Statistik. Bind 2*. IMM-DTU, 2003.

[20] Tim Cootes. Talking face, 2000. http://www.isbe.man.ac.uk/ bim/-data/talking_face/talking_face.html.

[21] Tim Cootes. Statistical models of appearance for computer vision. A technical report available from http://www.isbe.man.ac.uk/ bim/Models/app_models.pdf, 2004.

[22] Timothy F. Cootes, Gareth J. Edwards, and Christopher J. Taylor. Active appearance models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(6):681–685, June 2001.

[23] Teofilo Emidio de Campos, Rogerio Schmidt Feris, and Roberto Marcondes Cesar Junior. Eigenfaces versus eigeneyes: First steps toward performace assessment of representations for face recognition. In *MICAI*, pages 193–201, 2000.

[24] F. Dornaika and J. Ahlberg. Fast and reliable active appearance model search for 3-d face tracking. *Transactions on Systems, Man, and Cybernetics. Part B: Cybernetics*, 34(4):1838–1853, August 2004.

[25] F. Dornaika and J. Ahlberg. Model-based head and facial motion tracking. In *Computer Vision in Human-Computer Interaction: ECCV 2004 Workshop on HCI, Prague. Proceedings*, pages 221 – 232, 2004.

[26] Fadi Dornaika and Jörgen Ahlberg. Active appearance model search with motion compensation for 3d face tracking. In *Proceedings of the 4th European Workshop on Image Analysis for Multimedia Interactive Applications (WIAMIS)*, pages 359–364. IEEE, 2003.

[27] Teresa Farroni, Mark H. Johnson, and Gergely Csibra1. Mechanisms of eye gaze perception during infancy, 2004.

[28] P.E. Frandsen, K. Jonasson, H.B. Nielsen, and O. Tingleff. Unconstrained optimization, 2004.

[29] Zoubin Ghahramani. The em algorithm, 2002.

[30] Rafael C. Gonzales and Richard E. Woods. *Digital Image Processing.* Addison-Wesley, 1 edition, 1993.

[31] N.J. Gordon, D.J. Salmond, and A.F.M. Smith. Novel approach to nonlinear/non-gaussian bayesian state estimation. In *Radar and Signal Processing, IEE Proceedings F*, 1993.

[32] Jonathan Gratch and Stacy Marsella. Tears and fears: Modeling emotions and emotional behaviors in synthetic agents. *Proceedings of the 5th International Conference on Autonomous Agents, Montreal, Canada*, June 2001.

[33] Jane's Information Group. http://www.janes.com/defence/air_forces/news/jawa/jawa001013_1_n.shtml.

[34] D. X. Hammer, R. D. Ferguson, J. C. Magill, M. A. White, A. E. Elsner, and R. H. Webb. Image stabilization for scanning laser ophthalmoscopy, December 2002.

[35] D. W. Hansen, J. P. Hansen, M. Nielsen, A. S. Johansen, and M. B. Stegmann. Eye typing using markov and active appearance models. In *IEEE Workshop on Applications of Computer Vision - WACV*, pages 132–136, dec 2002.

[36] Dan W. Hansen and Arthur E. C. Pece. Iris tracking with feature free contours. In *Proc. workshop on Analysis and Modelling of Faces and Gestures: AMFG 2003*, October 2003.

[37] Dan W. Hansen and Arthur E.C. Pece. Eye typing off the shelf. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition 2004 (CVPR 2004)*, 2004. (to appear).

[38] Dan Witzner Hansen. *Committing Eye tracking.* PhD thesis, The IT University of Copenhagen, ITU, 2003.

[39] Dan Witzner Hansen and Arthur E.C. Pece. Iris tracking with feature free contours, 2003. http://brigade.umiacs.umd.edu/iccv2003/witznerPece_poster.pdf.

[40] J. Huang and D. Mumford. Statistics of natural images and models. In *CVPR99*, pages I: 541–547, 1999.

[41] National Eye Institute. http://www.nei.nih.gov/.

[42] SensoMotoric Instruments. http://www.smi.de/.

[43] Takahiro Ishikawa, Simon Baker, Iain Matthews, and Takeo Kanade. Passive driver gaze tracking with active appearance models. In *Proceedings of the 11th World Congress on Intelligent Transportation Systems*, October 2004.

[44] R. Jacob. Eye tracking in advanced interface design, 1995.

[45] Q. Ji and X. Yang. Real-time eye, gaze, and face pose tracking for monitoring driver vigilance. *RealTimeImg*, 8(5):357–377, October 2002.

[46] O. Tingleff K. Madsen, H.B. Nielsen. Methods for Non-linear Least Squares Problems. Lecture Note in 02611 Optimization and Data Fitting, 2004.

[47] T. Kawaguchi, D. Hidaka, and M. Rizon. Detection of eyes from human faces by hough transform and separability filter. In *ICIP00*, pages Vol I: 49–52, 2000.

[48] Kyung-Nam Kim and R.S. Ramakrishna. Vision-based eye-gaze tracking for human computer interface, 1999.

[49] M. La Cascia, S. Sclaroff, and V. Athitsos. Fast, reliable head tracking under varying illumination: An approach based on registration of textured-mapped 3d models. *PAMI*, 22(4):322–336, April 2000.

[50] Applied Science Laboratories. http://www.a-s-l.com/.

[51] The Software Usability Research Laboratory. http://psychology.wichita.edu/surl/usabilitynews/61/EZprint.htm.

[52] D. Leimberg, M. Vester-Christensen, B. K. Ersbøll, and L. K. Hansen. Heuristics for speeding up gaze estimation. In *Proc. Svenska Symposium i Bildanalys, SSBA 2005, Malmø, Sweden*. SSBA, mar 2005.

[53] SR Research Ltd. http://www.eyelinkinfo.com/.

[54] Bruce D. Lucas and Takeo Kanade. An iterative image registration technique with an application to stereo vision (darpa). In *Proceedings of the 1981 DARPA Image Understanding Workshop*, pages 121–130, April 1981.

[55] D. J. C. MacKay. *Information Theory, Inference and Learning Algorithms*. Cambrdige University Press, 2003.

[56] Y. Matsumoto and A. Zelinsky. An algorithm for real-time stereo vision implementation of head pose and gaze direction measurement. In *AFGR00*, pages 499–504, 2000.

[57] Iain Matthews and Simon Baker. Active appearance models revisited. *International Journal of Computer Vision*, 60(2):135 – 164, November 2004. In Press.

[58] Carlos Hitoshi Morimoto, Dave Koons, Arnon Amir, and Myron Flickner. Pupil detection and tracking using multiple light sources. *Image Vision Comput.*, 18(4):331–335, 2000.

[59] Tsuyoshi Moriyama, Jing Xiao, Jeffrey Cohn, and Takeo Kanade. Meticulously detailed eye model and its application to analysis of facial image. In *Proceedings of the IEEE Conference on Systems, Man, and Cybernetics*, pages 629 – 634, 2004.

[60] M. M. Nordstrøm, M. Larsen, J. Sierakowski, and M. B. Stegmann. The IMM face database - an annotated dataset of 240 face images. Technical report, Informatics and Mathematical Modelling, Technical University of Denmark, DTU, Richard Petersens Plads, Building 321, DK-2800 Kgs. Lyngby, may 2004.

[61] Takehiko Ohno and Naoki Mukawa. A free-head, simple calibration, gaze tracking system that enables gaze-based interaction. In *Proceedings of the Eye tracking research & applications symposium on Eye tracking research & applications*, pages 115–122. ACM Press, 2004.

[62] Takehiko Ohno, Naoki Mukawa, and Atsushi Yoshikawa. Freegaze: a gaze tracking system for everyday gaze interaction. In *ETRA*, pages 125–132, 2002.

[63] N. Otsu. A threshold selection method from grey-level histograms. *SMC*, 9(1):62–66, January 1979.

[64] Arthur E. C. Pece. The Kalman-EM contour tracker. In *Proc. 3rd workshop on Statistical and Computational Theories of Vision:SCTV 2003*, October 2003.

[65] Arthur E. C. Pece and Anthony D. Worrall. Tracking with the EM contour algorithm. In A. Heyden, G. Sparr, M. Nielsen, and P. Johansen, editors, *Proc. 7th European Conf. Comp. Vision: ECCV 2002*, LNCS 2350, pages 3–17. Springer, 2002.

[66] J. Pelz, R. Canosa, J. Babcock, D. Kucharczyk, A. Silver, and D. Konno. Portable eyetracking: A study of natural eye movements, 2000.

[67] A. Pérez, M. L. Córdoba, A. García, R. Méndez, M. L. Muñoz, J. L. Pedraza, and F. Sánchez. A precise eye-gaze detection and tracking system. In *WSCG*, 2003.

[68] Ravikrishna Ruddarraju, Antonio Haro, Kris Nagel, Quan T. Tran, Irfan A. Essa, Gregory Abowd, and Elizabeth D. Mynatt. Perceptual user interfaces using vision-based eye tracking. In *Proceedings of the 5th international conference on Multimodal interfaces*, pages 227–233. ACM Press, 2003.

[69] Stephen R. Schmitt. Analysis of conic sections. http://home.att.net/ srschmitt/conic_eqn_analysis.html.

[70] H. Schweitzer, J.W. Bell, and F. Wu. Very fast template matching. In *ECCV02*, page IV: 358 ff., 2002.

[71] Seeingmachines. http://www.seeingmachines.com/.

[72] J.R. Shewchuk. Triangle: engeneering a 2d quality mesh generator and delaunay triangulator. *Applied Computational Geometry, FCRC'96 Workshop*, pages 203–222, 1996.

[73] Frank Y. Shih and Chao-Fa Chuang. Automatic extraction of head and face boundaries and facial features. *Inf. Sci.*, 158:117–130, 2004.

[74] Smarteye. http://www.smarteye.se/.

[75] R. Smith, M. Shah, and N. da Vitoria Lobo. Determining driver visual attention with one camera. *ITS*, 4(4):205–218, December 2003.

[76] M. Sodhi, B. Reimer, J. L. Cohen, E. Vastenburg, R. Kaars, and Susan S. Kirschenbaum. On-road driver eye movement tracking using head-mounted devices. In *ETRA*, pages 61–68, 2002.

[77] M. Sonka, V. Hlavac, and R. Boyle. *Image Processing, Analysis and Machine Vision*. International Thomson Publishing, 2 edition, 1998.

[78] Kate Starbird and John Owens. Color edge detection theory. http://graphics.stanford.edu/ jowens/223b/theory.html.

[79] M. B. Stegmann. Active appearance models: Theory, extensions and cases. Master's thesis, Informatics and Mathematical Modelling, Technical University of Denmark, DTU, Richard Petersens Plads, Building 321, DK-2800 Kgs. Lyngby, aug 2000.

[80] M. B. Stegmann, B. K. Ersbøll, and R. Larsen. FAME – a flexible appearance modelling environment. *IEEE Trans. on Medical Imaging*, 22(10):1319–1331, 2003.

[81] Rainer Stiefelhagen, Jie Yang, and Alex Waibel. Tracking eyes and monitoring eye gaze. In *Workshop on Perceptual User Interfaces*, Banff, Canada, October 1997.

[82] Rainer Stiefelhagen, Jie Yang, and Alex Waibel. Estimating focus of attention based on gaze and sound. In *PUI '01: Proceedings of the 2001 workshop on Perceptive user interfaces*, pages 1–9. ACM Press, 2001.

[83] Eyetrack III study. http://poynterextra.org/EYETRACK2004/accuracy.htm.

[84] Eyetech Digital Systems. http://www.eyetechds.com/.

[85] K. Takemura, J. Ido, Y. Mastumoro, and T. Ogasawara. Development of non-contact drive monitoring system for advanced safety vehicle, 2003.

[86] K. Talmi and J. Liu. Eye and gaze tracking for visually controlled interactive stereoscopic displays, 1999.

[87] Eye Response Technologies. http://www.eyeresponse.com/.

[88] LC TECHNOLOGIES. http://www.eyegaze.com/.

[89] Toby Technology. http://www.tobii.se/.

[90] Ying-Li Tian, Takeo Kanade, and Jeffrey Cohn. Dual-state parametric eye tracking. In *Proceedings of the 4th IEEE International Conference on Automatic Face and Gesture Recognition (FG'00)*, pages 110 – 115, March 2000.

[91] Joseph van der Gracht, V. P. Pauca, Harsha Setty, Ramkumar Narayanswamy, Robert Plemmons, Sudhakar Prasad, and Todd Torgersen. Iris recognition with enhanced depth-of-field image acquistion. volume 5438, pages 120–129. SPIE, 2004.

[92] J.G. Wang and E. Sung. Study on eye gaze estimation. *IEEE Transactions on Systems, Man and Cybernetics - Part B: Cybernectics*, 32(3):332–350, June 2002.

[93] Jian-Gang Wang, Eric Sung, and Ronda Venkateswarlu. Eye gaze estimation from a single image of one eye. In *ICCV*, pages 136–143, 2003.

[94] Max Welling. *The Kalman Filter*. California Institue of Technology.

[95] Jing Xiao, Simon Baker, Iain Matthews, and Takeo Kanade. Real-time combined 2d+3d active appearance models. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, June 2004.

[96] Jing Xiao, Jinxiang Chai, and Takeo Kanade. A closed-form solution to non-rigid shape and motion recovery. In *The 8th European Conference on Computer Vision (ECCV 2004)*, May 2004.

[97] X. Xie, R. Sudhakar, and H. Zhuang. A cascaded scheme for eye tracking and head movement compensation. *T-SMC*, A28:487–490, 1998.

[98] X.D. Xie, R. Sudhakar, and H.Q. Zhuang. Real-time eye feature tracking from a video image sequence using kalman filter. *SMC*, 25(12):1568–1577, December 1995.

[99] S. Yoshimura and Takeo Kanade. Fast template matching based on the normalized correlation by using multiresolution eigenimages. In *Proceedings of the 1994 IEEE/RSJ/GI International Conference on Intelligent Robots and Systems, Advanced Robotic Systems and the Real World (IROS '94)*, volume 3, pages 2086 – 2093, September 1994.

[100] Y. Zhu and K. Fujimura. Head pose estimation for driver monitoring. In *IVS04*, pages 501–506, 2004.

[101] Zhiwei Zhu, Kikuo Fujimura, and Qiang Ji. Real-time eye detection and tracking under various light conditions. In *Proceedings of the symposium on Eye tracking research & applications*, pages 139–144. ACM Press, 2002.