

**UMA ABORDAGEM PARA ESTIMAÇÃO REMOTA
DO OLHAR SOB LUZ VISÍVEL EM TEMPO REAL**

SAMUEL FELIX DE SOUSA JUNIOR

**UMA ABORDAGEM PARA ESTIMAÇÃO REMOTA
DO OLHAR SOB LUZ VISÍVEL EM TEMPO REAL**

Dissertação apresentada ao Programa de Pós-Graduação em Ciência da Computação do Instituto de Ciências Exatas da Universidade Federal de Minas Gerais como requisito parcial para a obtenção do grau de Mestre em Ciência da Computação.

ORIENTADOR: MARIO FERNANDO MONTENEGRO CAMPOS

Belo Horizonte
16 de fevereiro de 2012

SAMUEL FELIX DE SOUSA JUNIOR

**AN APPROACH FOR REAL TIME REMOTE GAZE
ESTIMATION UNDER VISIBLE LIGHTING**

Thesis presented to the Graduate Program in Computer Science of the Universidade Federal de Minas Gerais in partial fulfillment of the requirements for the degree of Master in Computer Science.

ADVISOR: MARIO FERNANDO MONTENEGRO CAMPOS

Belo Horizonte
February 16, 2012

© 2012, Samuel Felix de Sousa Junior.
Todos os direitos reservados.

S725a Sousa Junior, Samuel Felix de
Uma Abordagem para Estimaco Remota do Olhar
sob Luz Visvel em Tempo Real / Samuel Felix de
Sousa Junior. — Belo Horizonte, 2012
xxviii, 73 f. : il. ; 29cm

Dissertao (mestrado) — Universidade Federal de
Minas Gerais — Departamento de Cincia da
Computao.

Orientador: Mario Fernando Montenegro Campos

1. Computao – Teses. 2. Viso Computacional –
Teses. I. Orientador. II. Ttulo.

519.6*84(043)



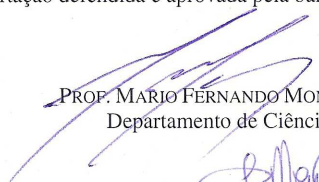
UNIVERSIDADE FEDERAL DE MINAS GERAIS
INSTITUTO DE CIÊNCIAS EXATAS
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

FOLHA DE APROVAÇÃO

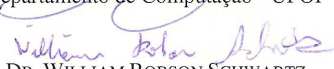
Uma abordagem para estimação remota do olhar sob luz visível em tempo real

SAMUEL FÉLIX DE SOUSA JÚNIOR

Dissertação defendida e aprovada pela banca examinadora constituída pelos Senhores:


PROF. MARIO FERNANDO MONTENEGRO CAMPOS - Orientador
Departamento de Ciência da Computação - UFMG


PROF. DAVID MENOTTI GOMES
Departamento de Computação - UFOP


DR. WILLIAM ROBSON SCHWARTZ
Instituto de Computação - UNICAMP

Belo Horizonte, 16 de fevereiro de 2012.

*to the sources of my strength and happiness:
my parents, Samuel and Silsa.*

Acknowledgments

I thank God for all the love that has been given to me and the strength to conclude my work. Also, it is a privilege to have great support from my family during this journey. My deepest gratitude goes to my parents Silsa Andrade and Samuel Felix, and my sister Samantha Vale.

I sincerely thank my supervisor Prof. Mario Campos, for kindly welcome me at Laboratório de Visão e Robótica (VeRLab), for introducing me to this research field and for his continuous support along the last two years. I also thank all the staff of PPGCC for helping me every time I needed. I thank all friends of my lab for the discussions, exchange of ideas and for the incredible help during experimental phase. Many thanks to Antônio Wilson, Erickson Nascimento, and Cláudio dos Santos.

My special thanks to Dafne Bastos, whose love, friendship and support guided me during this stage of my life. Special thanks to Marina Oikawa for always cheering me up. I also would like to thank Elizabeth Duane, Alberto Pimentel, and Yuri Tavares. Talking and spending time with these people has been one of the best parts of my study at UFMG, and I hope that some of them will remain friends for life.

Finally, I would like to thank the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES) for the scholarship that allowed this work to be finished. Portions of the research in this work use the FERET database of facial images collected under the FERET program, sponsored by the US Department of Defense Counterdrug Technology Development Program Office.

"We can dream the impossible and make it happen"
(Eugene Cernan)

Resumo

Esta dissertação propõe uma abordagem para o problema da estimação remota do olhar utilizando uma câmera localizada à frente do usuário. Uma solução também é proposta para o problema da estimação da pose facial, que permite melhorias na determinação do olhar. Dentre as inúmeras aplicações estão incluídas a interação humano computador, a monitoração de fadiga de motoristas e a recuperação de informação visual baseada em conteúdo.

Um dos desafios relacionados aos problemas da detecção do olhar, é a estimação da pose facial. Uma solução para esse problema pode permitir maior naturalidade na utilização do sistema e, a grosso modo, a possível orientação do olhar. Várias outras aplicações tem motivado o aumento crescente da investigação do rastreamento do olhar nos últimos anos. Diversas abordagens foram propostas na literatura, sendo que a grande maioria requer dispositivos específicos para o seu funcionamento correto. A solução proposta nesta dissertação baseia-se na análise de imagens adquiridas por câmeras comuns no espectro visível. Ou seja, nenhum dispositivo de hardware especializado ou mesmo requisitos rígidos sobre a iluminação são impostos.

A metodologia foi implementada e avaliada por meio de experimentos reais com diversos indivíduos. Os resultados obtidos demonstraram que o sistema opera adequadamente em tempo real, obtendo a localização dos olhos com exatidão. O sistema foi também aplicado no controle remoto de uma cabeça robótica e na produção de mapa de calor das regiões focadas pelo usuários.

Palavras-chave: Visão Computacional, Rastreamento de Olhar, Estimação da Pose Facial.

Abstract

This thesis proposes an approach for the remote gaze estimation problem using a single camera. A solution for the head pose estimation problem is also suggested, indicating the possible gaze location. It has applicability in Human Computer Interaction, in monitoring driver drowsiness for avoiding road accidents, and in Content Image Based Retrieval field.

Many applications have motivated the increasing investigation on gaze tracking on the past few years. Several approaches were proposed in the literature, many of those requiring specific hardware devices for working properly. The solution proposed in this thesis is based on the analysis of images acquired by ordinary cameras in visible lighting. In other words, no specific hardware device or even illumination constraints are imposed.

The methodology was implemented and evaluated by real experiments with different individuals. Results coming from those experiments demonstrated that the system works properly in real time, obtaining the eyes location accurately. Our system was also applied in the remote control of a robotic head. We built heatmaps of the regions focused by users in the screen.

Keywords: Computer Vision, Gaze Tracking, Head Pose Estimation.

List of Figures

1.1	Eye image acquired in visible light.	3
2.1	Anatomy of Human Eye	8
2.2	Sclera Search Coils	10
2.3	Electrooculography	10
2.4	EyeWriter Glasses	11
2.5	Active Infrared Illumination	12
2.6	Omega Group	15
2.7	K-Means Clustering	16
2.8	Yaw, Pitch, and Roll	19
2.9	Shape	22
2.10	ASM Start and Final shapes	23
3.1	Overview of the two main modules	25
3.2	Head Pose Methodology	28
3.3	Eye Tracking Methodology	29
3.4	LK Tracking	30
3.5	Mesh Projection	31
3.6	Face Proportion	32
3.7	ROI and Mask	34
3.8	Enhanced Iris and Binarized Image	35
3.9	Results of Iris Detection	38
3.10	Template Matching	38
3.11	Calibration Pattern	39
3.12	Calibration Procedure	40
3.13	Example of Good Calibration	40
4.1	Robotic Head	44
4.2	Tele-immersion robot used in experiments.	45

4.3	Robotic Head	45
4.4	Overview of Users Eyes	46
4.5	Calibration Results	48
4.6	Gaze results of Individual 1	49
4.7	Gaze results of Individual 2	50
4.8	Gaze results of Individual 3	51
4.9	Gaze results of Individual 4	52
4.10	Gaze results of Individual 5	53
4.11	Gaze results of Individual 6	54
4.12	Gaze results of Individual 7	55
4.13	Good iris exposure vs Bad iris exposure	57
A.1	Mathematical Morphology	67
A.2	Structuring Element	68
A.3	Erosion	69
A.4	Dilation	69
B.1	Conics	73

List of Tables

2.1	Dataset Use for Eyes and Face detection	17
4.1	Quantitative Results for Individual 1	47
4.2	Quantitative Results for Individual 2	47
4.3	Quantitative Results for Individual 3	49
4.4	Quantitative Results for Individual 4	50
4.5	Quantitative Results for Individual 5	51
4.6	Quantitative Results for Individual 6	52
4.7	Quantitative Results for Individual 7	53
B.1	Quadric Surfaces	72
B.2	Conic Sections	73

List of Acronyms

AAM	Active Appearance Model
ALS	Amyotrophic Lateral Sclerosis
ASEF	Average of Synthetic Exact Filters
ASM	Active Shape Model
FERET	The Facial Recognition Technology Database
EOG	Electrooculography
GSL	GNU Scientific Library
HCI	Human Computer Interaction
IC	Isocenters
IMU	Inertial Measurement Unit
LED	Light Emitting Diodes
LK	Lucas and Kanade [1981]
MM	Mathematical Morphology
IR	Infrared
PCA	Principal Component Analysis
PoR	Point of Regard
POS	Pose from Orthography and Scaling
POSIT	POS with Iterations
RANSAC	RANdom SAmples Consensus
RGB	Additive Color Space
ROI	Region of Interest
SSC	Scleral Search Coils
V4L2	Video for Linux 2
YUV	Luminance and Chrominance Color Space

List of Algorithms

1	Algorithm for Remote Gaze Estimation	26
2	Iris Detection and Tracking Algorithm.	33
3	Circle Detection Algorithm.	37

Contents

Acknowledgments	xi
Resumo	xv
Abstract	xvii
List of Figures	xix
List of Tables	xxi
List of Acronyms	xxiii
List of Algorithms	xxv
1 Introduction	1
1.1 Motivation	1
1.2 Overview of the Problem	2
1.3 Related Problems	3
1.4 Contributions	4
1.5 Roadmap of this Thesis	4
2 Gaze Tracking: A Literature Review	7
2.1 Eye Segmentation and Gaze Estimation	7
2.1.1 Anatomy of the Human Eye	8
2.1.2 Non Video-based Approaches	9
2.1.3 Video-based Approaches	9
2.1.4 Eye Segmentation Approaches	12
2.2 Head Pose Estimation	18
2.2.1 Human Head Behavior	19
2.2.2 Active Shape Model	21

3	Methodology	25
3.1	Methodology Overview	27
3.2	Head Pose Estimation Algorithm	29
3.2.1	Tracking of Facial Landmarks	29
3.2.2	Mesh Projection and Pose Estimation	30
3.3	Eye Segmentation Algorithm	32
3.3.1	Iris Compensation	33
3.3.2	Circle Detection	35
3.3.3	Consensus of Circle Fitting	35
3.3.4	Iris Tracking	38
3.4	Gaze Estimation Algorithm	39
4	Experimental Analysis	41
4.1	Experimental Setup	41
4.1.1	Software	41
4.1.2	Hardware	43
4.2	Qualitative Results	43
4.2.1	Heatmap of Gaze Location	43
4.2.2	Remote Control of a Robotic Head	44
4.3	Quantitative Analysis	44
4.3.1	Gaze Measurements	47
4.4	Limitations of Our Work	56
5	Conclusions and Future Work	59
	Bibliography	63
	Appendix A Mathematical Morphology	67
A.1	Structuring Element	68
A.2	Erosion	68
A.3	Dilation	69
	Appendix B Quadrics and Conics	71
B.1	Algebraic Curves and Surfaces	71
B.2	Quadrics and Conics	72

Chapter 1

Introduction

The advance of technology changes the way people interact with computers. Human Computer Interaction (HCI) has applied great effort on planning and designing a better interface for interaction with computational devices.

Recently, computer vision has brought many unusual and challenging opportunities for interaction with human beings using cameras. For instance, *Gaze Tracking* is the problem of estimating where a person is looking using head-mounted or remote devices. An effective gaze tracking system would definitely enhance HCI for most individuals. However, for those ones not capable of interacting with computational systems by using conventional devices such as mice or tablets, gaze tracking becomes more than just an innovative technique, but an essential device for allowing one to connect to the world.

1.1 Motivation

Many different problems can be solved using computer vision. Indeed, several different fields take advantage of cameras. Some include reconstruction problems (i.e. when a three-dimensional model of an object needs to be estimated), others have applied computer visions in surveillance, mobile robot localization, object detection and recognition, *etc.* There is a whole field called *Digital Image Processing* for dealing specifically with image related problems such as: Image Enhancing, Segmentation, and so forth.

When the cost for acquiring a sensor is reduced, it tends to become more popular. Therefore, digital cameras are inexpensive and useful sensors available for common users. This has provided a perfect scenario for applying computer vision to handle problems that were most unlikely to be solved in the past due to the diffi-

culties of obtaining a powerful and precise sensor.

Until the advent of tablets and mobile devices, the most popular way to interact with computers was using keyboards and mouses. Nevertheless, there are many users that due disability are incapable of using computers with their hands. For instance, individuals suffering from Amyotrophic Lateral Sclerosis (ALS) need *gaze trackers* to assist them in communication. Hence, this thesis is motivated by the need to handle the problem of estimating the Point of Regard (PoR) by proposing a simple, fast, and inexpensive remote gaze tracker. Point of Regard or Point of Gaze is a point in the scene imaged in the fovea of the eye [Guestrin and Eizenman, 2006].

1.2 Overview of the Problem

Roughly, the problem of estimating the PoR consists in determining where a person is looking. In our configuration, the user is looking towards a plane, which can be the computer screen and the *input* of our system is the data gathered by a single web cam standing frontal to the user. The system *output* is the two-dimensional location on the screen representing the gaze.

Many approaches address the gaze problem by using images acquired in Infrared (IR) spectrum from one or more cameras using active lighting. Active lighting is a technique that uses a light projected onto the eye surface generating a glint [Mulvey et al., 2008]. The difference between both pupil and glint is computed for estimating the gaze. Also, many commercial gaze trackers use chin rests and helmets to restrict head movements, making the task easier when compared with head free estimators. Although this head movement constraint helps the estimation, it is intrusive since it restricts the natural behavior of the user.

As aforementioned, several techniques use IR [Morimoto et al., 2000; Ji and Yang, 2002]. Nevertheless, a more interesting approach would perform it under visible wavelength lighting, since ordinary cameras can be used instead of specific hardware devices. Unfortunately, there are some drawbacks when using visible wavelength lighting instead of IR ones. For instance, lack of illumination jeopardizes the image quality which may lead to failure during the tracking process. Also, illumination produces specular reflections on the iris, corneal, and sclera surfaces. Pupil might not be detected, due to poor illumination and the low contrast on dark colored eyes. Hence, generally, tracking is based on the iris region. However, in some individuals, the iris might be blocked by the eyelids, turning the gaze estimation based on visible lighting into a hard and complex problem.



Figure 1.1. Human female eye image acquired in visible light. There are specular reflections and glints generated in eye's surface due to source lights in the scene.

Figure 1.1 shows the human eye under visible light. The iris is also known as the circular colored part of eye. Pupil is the aperture in the center of the iris. Sclera is the white region surrounding the iris. Section 2.1.1 provides a better explanation of the eye's anatomy.

Problem (remote non-intrusive gaze tracker) *Given a user with a single camera capable of capturing images under visible lighting, estimate and track the user's gaze in real time.*

This is the problem we address in this thesis and for handling it, we propose approaches for addressing two other problems: The *Head Pose Estimation Problem* and the *Iris Segmentation Problem*. In this thesis, we revise those three problems and propose solutions for them.

1.3 Related Problems

There are several problems related to gaze estimation, which are beyond the scope of this work. However, they are important to the context of this work and therefore we will just list them. The following problems are not considered in this thesis:

- *Face recognition* - the problem of distinguishing an individual by identifying his face against others. This is a very interesting problem and there are a vast number of promising works in the literature. Recently, Ni and Chellappa [2010] performed an overview on remote face recognition algorithms.
- *Multiple users* - In case of multiple faces detected, we track the largest one, assuming that it is the closest face to the camera.
- *Illumination invariance* - A poor illuminated face or abrupt light changes might cause problems to the system behaviour. Illumination invariance is not the

focus of this work.

1.4 Contributions

The main contributions sought for the development of this thesis are:

- An overview on *Head Pose, Eye Tracking, and Gaze Estimation* problems focusing special attention on remote techniques that use a single camera.
- The development of a simple *Head Pose Estimator* based on image features that allow a reliable pose estimation. Our contribution consists on the mesh projection based on some facial proportions.
- The development of a *Iris Segmentation Algorithm* for extracting the information needed for gaze estimation. Our contribution consists of a mask that decreases glints and reflections of eye surface for enhancing the limbus detection.
- The development of a *Gaze Tracker* and demonstration of its capabilities by heatmap computation and quantitative analysis of seven individuals. Our motivation for heatmap is based on the focus of attention. Regions that grabbed more attention of users are highlighted by using reddish colors.
- The application of the proposed approach on the remote control of a *Robotic Head*. Our motivation focused on the tele-immersion simulation, where the head movements of human controller are reproduced remotely by the robot's head.

1.5 Roadmap of this Thesis

Chapter 2 aims to provide an overview of recent results on head pose and gaze estimation, focusing on approaches using images acquired under visible wavelength. That chapter also describes relevant information regarding eye anatomy and head behavior. Furthermore, it introduces some techniques for iris and sclera detection based on different approaches: Isophotes, Correlation Filters, Clustering and so on. Finally, it presents some databases that are commonly used as ground-truth in eye and face detection.

Chapter 3 introduces the methodology applied in this work and the background to support our design decisions. It describes the two modules proposed in this thesis. The first module defines the steps applied for head detection and pose

estimation. The second one describes our approach on iris segmentation, tracking and gaze estimation. Also, it discusses the calibration procedure used. Considering that we do not have three-dimensional information, we need to establish some calibration for mapping the points in the image into points in the screen for estimating the user's gaze.

Chapter 4 explains how we have structured and conducted the experiments (both quantitative and qualitative). Some applications have been developed in robotics whose interaction is built on the head pose and gaze proposed in this thesis. All applications are introduced and discussed in that chapter.

Chapter 5 presents our conclusions related to those three problems, the constraints and limitations of the current approach, and it points out some directions for future work. It also discusses the importance of gaze trackers in computer related research in different areas.

Chapter 2

Gaze Tracking: A Literature Review

Due to their vast applicability in HCI and human behavior studies, several researchers have been investigating eye tracking techniques over the past years. Many different approaches have been proposed, some of them requiring specific hardware devices either to enhance the eye detection or to decrease the impact of head movement. Along with eye and gaze estimation, the head pose estimation problem has also played an important role in the eye tracking community, due to the more natural interaction when using such system, but also because head pose provides a coarse gaze direction when the eyes are not visible [Murphy-Chutorian and Trivedi, 2009].

A pose can be estimated by using, for instance, deformable models or nonlinear regression methods. In this chapter, we aim to provide a literature review on head pose estimation, eye, and gaze tracking solutions. Murphy-Chutorian and Trivedi [2009] have organized head pose estimation approaches into eight categories according to their operating domain. Hence, they proposed a *head taxonomy* that shall be discussed later.

2.1 Eye Segmentation and Gaze Estimation

The *eye taxonomy* described by Hansen and Ji [2010] categorizes eye detection into shape-based, appearance-based, and hybrid methods. Shape-based methods are the ones based on features, edges, and other structures that are useful for constructing a fixed or deformable model. Appearance-based, on the other hand, focuses on template matching or statistical approaches (holistic) for analysing the object appearance. An advantage of this latter method is the possibility to conduct a detection technique either in spatial or transformed domain. Hybrid methods combine those

approaches for improving the results.

In this chapter, we focus not on the explanation on many different approaches but on the theoretical foundation for the problems we address in this thesis and on recent results regarding those three problems. For general understanding, we first provide a simplistic model of head movement behavior and eye anatomy explaining both properties and constraints associated with relevant gaze topic and later in this chapter we discuss in a deeper analysis each technique.

2.1.1 Anatomy of the Human Eye

We now move into the eye tracking problem. However, in order to fully understand how a eye tracker system should work, it becomes essential to comprehend the anatomy of human eye. Figure 2.1 shows a simplistic view of human eyeball.

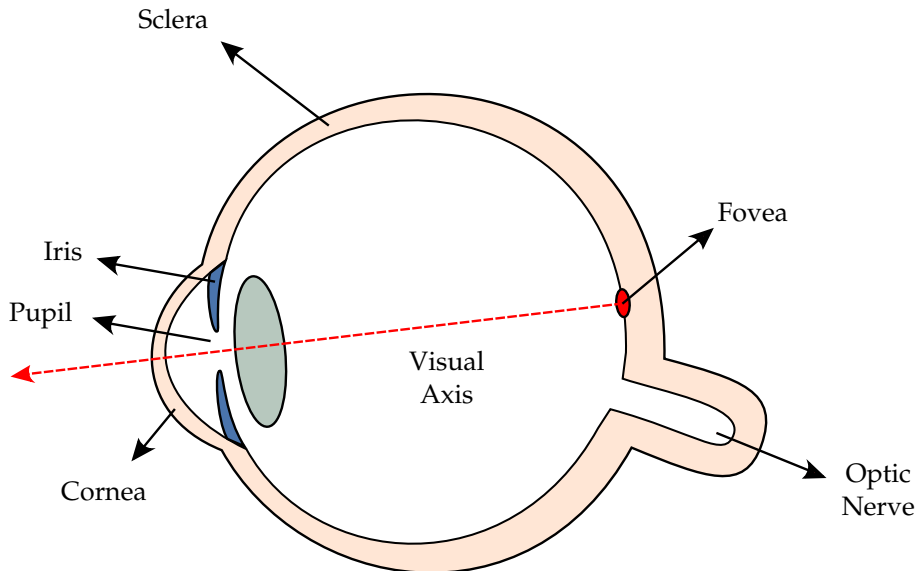


Figure 2.1. Anatomy of Human Eye

According to Hansen and Ji [2010], the human eyeball is relatively spherical with radius varying in the range 12-13 mm. The *sclera* (also known as the white part of the eye) is one of the easiest distinguishable properties. Iris is commonly defined as the colored part of the eye which contains an aperture called *pupil*. The pupil is responsible for controlling the amount of light which enters the eye by contracting and expanding itself. The *cornea* is a membrane that lies on the eye's surface. Finally, *fovea* is a region in the center of retina that contains many sensitive cells. The line which connects the fovea and the center of cornea is known as the Line of Gaze (LoG) or visual axis.

2.1.1.1 Eye Movements

The human eye has different kinds of movements. According to Duchowski [2007], some eye movements correspond to:

- **Saccades:** rapid eye movements (ranging from 10 to 100 ms) which are voluntary and reflexive. The goal of this movement is to reposition the fovea with respect to a new location in the environment. On the other hand, microsaccades are involuntary and correspond to small spacial random movements that generally occurs during fixation.
- **Smooth Pursuit:** this kind of movement is associated when an individual is tracking a moving object on the screen. The human eye is capable to synchronize with the velocity of the moving object.
- **Fixation:** this movement occurs when an individual is focusing a stationary object, so the image on the retina is stabilized. During fixation, other movements occur, such as microsaccades.

2.1.2 Non Video-based Approaches

Some techniques are not based on video processing, but on measuring signals obtained by different sensors.

Scleral Search Coils (SSC) is a technique that uses a contact lens with a coil of wire attached to the subject eye. It is based on the assumption that a magnetic field induces voltage while intersecting the coil. In this way, the position of the eye can be accurately retrieved, but this solution is intrusive due to use of contact lens and uncomfortable for users. Figure 2.2 displays the use of SSC technique.

Another way to measure the eye position is using the Electrooculography (EOG) method. The information is processed by placing electrodes around the eye and it infers the position also related to the head. Figure 2.3 shows an example of electro-oculography system.

2.1.3 Video-based Approaches

Video-based approaches (video-oculography) represent the most common way to track the gaze by acquiring images delivered by one or more cameras. There are several different implementations of video-oculography that may use either visible



Figure 2.2. Sclera Search Coils developed by Chronos Vision. Image extracted from: <http://www.chronos-vision.de/eye-tracking-produkte.html>



Figure 2.3. EOG system by PHYWE Biology. Image extracted from: <http://www.phywe.com/461/pid/26780>

wavelength light or infrared light. Moreover, the solution might require a wearable device, such as glasses or helmet, or it can determine the gaze using a remote approach, reducing intrusiveness.

This section provides information regarding those techniques and classifies the current work according to those differences.

2.1.3.1 Head-Mounted Gaze Trackers

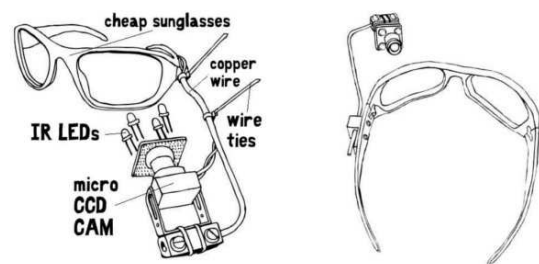
Free human head movements imply in a less intrusive and more natural system for the user but it is a hard task for tracking the gaze once head movements affect the accuracy of the estimation.

Among many HCI applications that could take advantage of a gaze tracker, the EyeWriter initiative [Lieberman et al., 2011] developed an open source software along cheap hardware for allowing disabled people to write and draw art by using only the eyes. The system is built upon head mounted device as shown in Figure 2.4. Potential users are those artists disabled by Amyotrophic Lateral Sclerosis (ALS) and others that are not capable of making movements with their hands or heads but eyes.

The EyeWriter group uses a Infrared Sensitive Camera attached to the glasses and the eye is illuminated by IR Light Emitting Diodes (LED) for creating a high contrast between the iris and pupil, thus enhancing the pupil region. Roughly, the software part of the system is divided in the eye tracking and drawing modules for performing the art activity. The eye tracking software has an ellipse detection algorithm based on the iris pixels and a calibration procedure that uses a method for interpolating the gaze location based on the segmented iris location.



(a) Artist using the EyeWriter System



(b) System Representation

Figure 2.4. EyeWriter: An initiative for artists and writers disabled by Amyotrophic lateral Sclerosis to produce art by using the movement of eyes. Images extracted from <http://www.eyewriter.org/>

2.1.3.2 Remote Gaze Trackers

Remote gaze estimation is usually obtained by using video cameras that do not require the usage of glasses, although some remote trackers use chin-rests for stabilizing the subject's head.

A common approach for commercial eye trackers is to use active near infrared light with wavelength in 780-880 nm as described by Hansen and Ji [2010]. Such

systems project an IR light onto the individual's eyes generating a glint. This kind of system is somehow calibrated to identify the pupil ellipse and the glint.

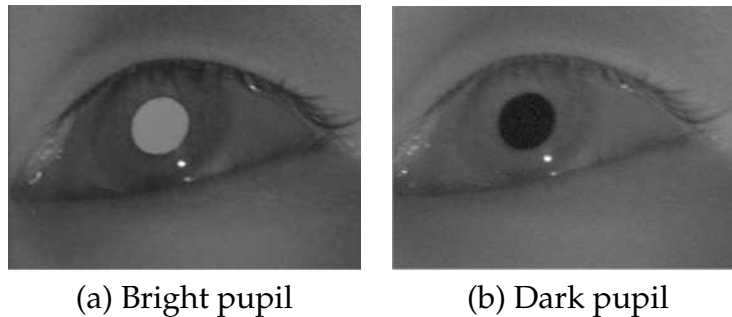


Figure 2.5. Active Infrared Illumination. Pupil reflection in infrared mode with one glint projection. Picture obtained in [Mulvey et al., 2008].

Despite Infrared illumination, there are many other remote gaze trackers based on visible light and they are carefully discussed in the next section.

2.1.4 Eye Segmentation Approaches

Parker and Duong [2009] propose a technique for gaze tracking based primarily on the sclera recognition. The color of sclera is approximately stable for different gender and ethnic individuals which becomes an easily detectable feature for the eye.

A two phase process is performed for detecting sclera region. First, all pixels inside the face bounding box are classified into four categories: $A = \{\text{skin}\}$, $B = \{\text{sclera}\}$, $C = \{\text{hair, iris, and eye shadow}\}$, $D = \{\text{noise}\}$. They remove pixels belonging to classes C and D based on thresholding and generate an histogram for the R channel. They say that this histogram is going to have a bi-modal behavior and the larger peak represents skin pixels while the smaller peak contains sclera pixels. For some people, this pattern might be more evident than for others. The second phase consists in using Mahalanobis distance in the U and V components to classify the first-phase successful pixels into either skin or sclera pixel (considering now the Luminance and Chrominance Color Space (YUV)). Finally, iris pixels are detected using thresholding and least squares approximation.

As aforementioned, many IR approaches use the point-reference technique. According to Proença [2011], IR wavelength can be dangerous, considering that the eye does not respond naturally to the illumination exposure (such as pupil contraction, blinking, etc). Hence, in order to compare the iris to a stable fixed point, they proposed a novel feature called *eye-region point*. So, the eye bounding box is esti-

mated using iris and sclera previously obtained and sclera shape is generalized as an ellipse. By using the eye region boundary and the major axis of the estimated ellipse, they compute the *eye-region point* by calculating the center of mass. They proposed the eye-iris vector $V = \{(e_{row}, e_{col}), (i_{row}, i_{col})\}$ which combines the row and columns of the stable reference point (*eye-region point*) and the iris center respectively. This vector is created during calibrating procedure while user focuses his gaze onto the corners of the screen.

Experiments were conducted with different subjects of different ethnicities, genders, and lighting conditions. A frame is only considered successfully detected if both eyes regions are identified and fully bound. Results coming from those experiments seem to support the reliability of proposed method.

Proença [2011] proposed a two phase technique for segmenting iris in degraded visible wavelength images. First, a deterministic linear-time algorithm is proposed for discriminating noise-free iris pixels from other pixels. After that, iris parametrization is performed using constrained polynomial fitting.

The proposed technique assumes that sclera is the most distinguishable property of the eye. They have empirically chosen the hue, blue and red chroma (hcbcr) color components and extracted a 20-dimensional feature set for the image pixels:

$$\{x, y, h_{0,3,7}^{\mu,\sigma}(x, y), cb_{0,3,7}^{\mu,\sigma}(x, y), cr_{0,3,7}^{\mu,\sigma}(x, y)\}, \quad (2.1)$$

x and y define the pixel location and h, cr, cb define the color region centered at the provided location. Superscripts denote average (μ) and standard deviation (σ) for each region defined by those radii values (subscripts).

They introduce a novel feature called *proportion of sclera* which aims to measure the proportion of sclera pixels in a direction $d(\uparrow, \downarrow, \leftarrow, \rightarrow)$ according to a reference point (x, y) . There are also some steps using neural network classification (back-propagation learning algorithm). This approach ends up by parametrizing the ellipse using constrained polynomial fitting.

They conducted several experiments with some well-known datasets. For instance, their algorithm was compared against three techniques: integrodifferential operator, active contour approaches, and the algorithm proposed by Tan et al. [2010]. They conclude that their results were usually similar to [Tan et al., 2010] for VW, but integrodifferential and active contours had some problems on degraded images in VW, although they are effective in IR datasets.

Valenti and Gevers [2008] proposed an accurate algorithm for eye location and tracking using *isophote curvature*. Isophotes are defined as curves connecting points

of same intensity and they can be used for estimating center of semicircular patterns. That center estimation is reached by a voting step based on isophotes. However, the authors point out that in real world there is no guarantee that the boundaries of an object will have the same intensity values, which means that it can produce meaningless results if all isophotes are allowed to vote for the curvature center. Hence, in order to overcome it, they state that only meaningful parts of the isophotes should be used: the ones that follow the edges of an object. They called Isocenters (IC) the high responses on isocentric isophotes patterns that are near edges. Finally, they can discriminate between dark and bright centers by checking the sign of the curvature. So, considering that sclera is brighter than iris they ignore the positive curvature. The estimated eye center is then represented by the maximum isocenter (MIC).

The approach proposed by Bolme et al. [2009] introduces a class of correlation filters called Average of Synthetic Exact Filters (ASEF). Authors endorse that basically eye detection can be performed when a prior knowledge of eye is known, e.g. when a face is successfully detected. However, an accurate eye localization without prior constraints would be a harder and challenging task. That is the category of localization that ASEF obtains better performance.

A relevant difference among ASEF and other correlation filters relies on the fact that ASEF filter is over constrained, which means that the training process of ASEF filters considers images which specify a desired response (generally it is a bright peak at the center of the desired object) at every location in the training images. For avoiding the over-fitting problem they average the filters of those images. Authors show the benefits of using ASEF such as: more freedom and flexibility in the training step, considering that training images are not required to be centered on the target due to the fact that the peak may be placed wherever the target appears.

In their experiments, they conducted a localization restricted to eye regions and a localization without constraint. The ASEF filters were compared with other correlation filters, to Cascade Classifier, and Gabor Wavelet algorithms. The results from the restricted experiment indicate that all algorithms have a good performance when searching for the eye in a restricted region, but ASEF and UMACE filters produce more interesting results if compared to those other two well known approaches. In the second experiment, the whole image was taken into account, but Gabor filter and Cascade Classifier were removed from the experiment due to some problems (such as high false alarm rate of the Cascade Classifier) that would not produce good results. Hence, ASEF showed better results than other methods. An interesting result is that ASEF produces high responses for a correct eye, but rarely it outputs a wrong eye.

Nguyen et al. [2010] noticed that Additive Color Space (RGB) images have information which can make the iris detection easier by creating a compensated red channel. They first define $\Psi_{R-G}(x, y)$ and $\Psi_{R-B}(x, y)$ as the difference between red ($\Psi_R(x, y)$), green ($\Psi_G(x, y)$), and blue ($\Psi_B(x, y)$) channels:

$$\begin{aligned}\Psi_{R-G}(x, y) &= \Psi_R(x, y) - \Psi_G(x, y), \\ \Psi_{R-B}(x, y) &= \Psi_R(x, y) - \Psi_B(x, y),\end{aligned}\tag{2.2}$$

Nguyen et al. [2010] also noticed that when the difference ($\Psi_{R-G}(x, y)$ and $\Psi_{R-B}(x, y)$) is computed between channels for green and blue irises, the region approximately over the irises tend to be near zero or smaller than zero. Hence, they defined the set Ω for those pixels that are smaller than zero in the $\Psi_{R-G}(x, y)$ channel, and those ones that are smaller than five in the $\Psi_{R-B}(x, y)$:

$$\Omega = \{(x, y) | (\Psi_{R-G}(x, y) < 0) \cup (\Psi_{R-B}(x, y) < 5)\}.\tag{2.3}$$

Figure 2.6 displays the Ω sets for three European eyes. The first row shows the original RGB images and the second row shows pixels inside the Ω set (white ones).

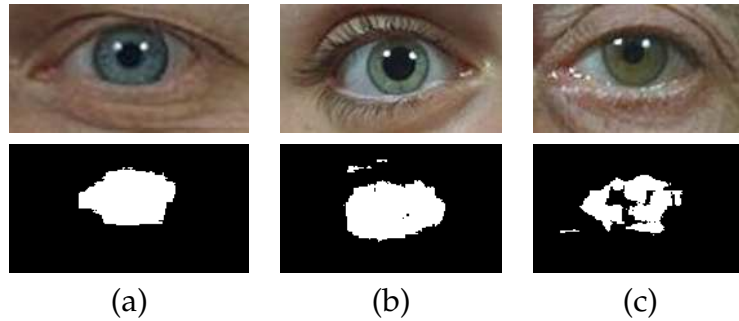


Figure 2.6. The first row shows European eyes and the second one shows the pixels inside the group Ω .

As it might be noticed, the group lies partially over the iris region giving us a good clue about where the iris is. Hence, they propose a compensation factor (λ) that takes into account information related to values of Ω pixels in the $\Psi_{R-G}(x, y)$ channel:

$$\lambda = \ln \left(\left| \frac{\sum_{(x,y) \in \Omega} \Psi_{R-G}(x, y)}{\|\Omega\|} \right| \right).\tag{2.4}$$

The compensated red channel ($\Psi_R^c(x, y)$) is obtained by convolving the

$\Psi_{R-G}(x, y)$ with a Gaussian kernel and the compensation factor (λ):

$$\Psi_R^c(x, y) = \Psi_R(x, y) + \lambda * G_\sigma * \Psi_{R-G}(x, y), \quad (2.5)$$

we will explain the impact of compensating an image during the clustering step.

Considering that the image has been compensated, pixels need to be separated into their classes. In other words, pixels belonging to the iris should be grouped into the same *cluster* in the same way that sclera pixels should remain together. For that task, they applied k-Means to find four clusters ($k = 4$), which is a non-supervised learning algorithm that has been applied for eye segmentation before [Li et al., 2010]. After the clustering process, there are four groups of pixels and one of them might represent the iris that the algorithm is trying to find.

We have implemented the algorithm of [Nguyen et al., 2010] and Figure 2.7 demonstrates the difference of applying the clustering algorithm on the original red channel of an image and applying it on an image compensated by Equation 2.5. The first row shows the result for the first case, where no compensation has been used. As one might notice, the iris is segmented into several classes. This undesired effect happens due to glints generated by source lights, but also due to the color difference of the iris and pupil region.

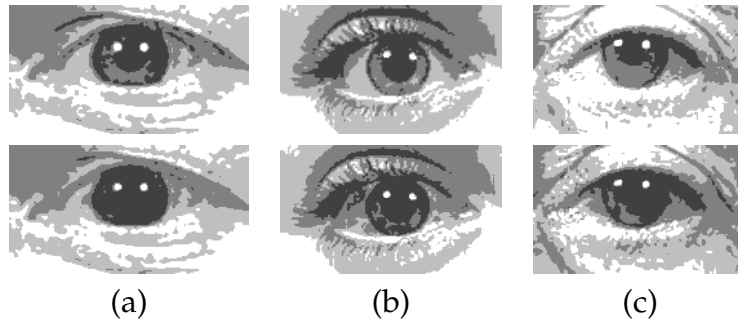


Figure 2.7. Clustering using k-Means. The first row shows the clustering without the compensation and the second one shows the groups in the compensated channel

The pixels of Ω set (generated by Equation 2.3) are mainly negative. So, by increasing their magnitudes and integrating them into the original image, the iris region will be darker. Hence, the difference generated by glints and pupil region is going to be reduced increasing the iris separability.

After applying k-Means, Nguyen et al. [2010] suggested that the class with the minimum mean in the $\Psi_R^c(x, y)$ should be selected as the iris one. They proposed the following equation for estimating the iris class:

$$\Omega_{iris} = \arg \min \frac{\sum_{(x,y) \in \Omega_k^I} \Psi_R^C(x,y)}{\|\Omega_k^I\|}, k = \{1, 2, 3, 4\}. \quad (2.6)$$

In Equation 2.6, Ω_k^I is the set of pixels grouped into the cluster k . So, after taking the values (intensity) of group k from the compensated channel and dividing it by the size of the this group, we have a number that represents the mean of that group in the $\Psi_R^C(x,y)$. Assuming that the iris has a small mean, the class which minimizes that equation is chosen as the iris class (Ω_{iris}).

The techniques presented in this section use different datasets. Those datasets contain images acquired in visible wavelength. Table 2.1 summarizes those previous methods¹, the evaluated dataset, and main strategy applied.

Table 2.1. Dataset Use for Eyes and Face detection

Method	Datasets	Strategies
Valenti and Gevers [2008]	BioID, YALE B	Isophotes Curvature
Bolme et al. [2009]	FERET	Correlation Filter (ASEF)
Parker and Duong [2009]	(None)	Thresholding + Mahalanobis
Proença [2011]	FERET, ICE FRGC, UBIRIS.v2	Feature Extraction + NN + Constrained Polynomial Fitting
Milborrow [2007]	XM2VTS, BioID, AR	Active Shape Model (ASM)
Nguyen et al. [2010]	FERET	K-Means + Hough

BioID [2001] database offers 1521 grayscale images in *pgm* format with resolution of 384×286 pixels. Images were taken with different illumination conditions of 23 people and for each image, there is a file describing the eye location. Owners of the dataset proposed the *relative eye distance* as measure for comparing the quality of different detection algorithms. There is also several additional points that were manually labeled for allowing facial analysis and gesture recognition studies.

The Facial Recognition Technology Database (FERET) [Phillips et al., 2000] was a program sponsored by the Department of Defense through the DARPA whose mission was to develop automatic facial recognition technologies for security, intelligence and law enforcement personnel in the performance of their duties [Phillips et al., 2000]. It contains 3,368 images of 1,204 people.

The University of Surrey XM2VTS database [Messer et al., 1999] is a multi-modal face database available for purchase containing different information from

¹The work of Milborrow [2007] has not been introduced yet. However it will be explained later.

295 people acquired during four months. A great advantage of this database is that it has been manually labelled with 68 points across images. This is the dataset chosen by Milborrow [2007] for training his face detector based on deformable templates.

Many eye segmentation approaches have been proposed with interesting and promising ideas. For instance, the colored iris compensation proposed by [Nguyen et al., 2010] helps to reduce glints in the image. We will expand this compensation idea for any color of iris due to the fact it reduces reflections and enhances the segmentation approach. Valenti and Gevers [2008] proposed a voting procedure for detecting the idea based on isophotes. However, Hansen and Ji [2010] points out that the isophote voting method might produce incorrect detection by finding eye corners and eyebrows instead of eyes due to the fact they rely on maxima in feature space. For avoiding such problems in our algorithm, we create an image ROI based on a prior face model and we apply an algorithm to to both constraint the search space and to remove non relevant pixels in the image.

2.2 Head Pose Estimation

Human expressions and intentions are easily recognisable in the face. It is straightforward for a human being to perceive and distinguish the orientation and displacements of someone's head. In the computer vision sense, head pose estimation might be understood as the determination of a person's head relative to a coordinate system (e.g. camera).

Despite all the meaningful roles of head orientation in human behavior such as intention or gesture, we are interested in the relationship between head orientation and gaze location. Murphy-Chutorian and Trivedi [2009] points out that the head pose problem is intrinsically related to the gaze problem, discussing that head's pose presents itself as a coarse indication of the gaze. Moreover, they emphasize that for an accurate eye tracking, it is required to know the head pose, and they argue that using only a passive camera without knowledge of light conditions, it is not possible to accurately estimate the human gaze.

Before estimating the pose of a given object, it is required to determine whether such object exists in the image or not, and if so, where it is located in the image.

Problem 2.1 (head detection and localization). *Let I be an image, determine if there is a head H in I . If yes, retrieve the tuple $\{x, y, width, height\}$ associated to H where x, y stand for the left upper corner (in pixel coordinate) of the Region of Interest (ROI) that contains H and width and height stand for the dimension of this ROI.*

For solving Problem 2.1, two of the most relevant solutions are the Viola and Jones [2004] and Rowley et al. [1998] detectors. In the study of Active Shape Model conducted by Milborrow [2007], he applied both algorithms and he has shown pros and cons for both of them. He concluded that although for some of his experiments Rowley et al. [1998] presented better results, it is slower than Viola and Jones [2004]. Hence, in this work, we have decided to choose Viola and Jones [2004] as our face detector.

2.2.1 Human Head Behavior

Figure 2.8 displays a human head along its three angles. Now, we state a related problem that we also address in this thesis.

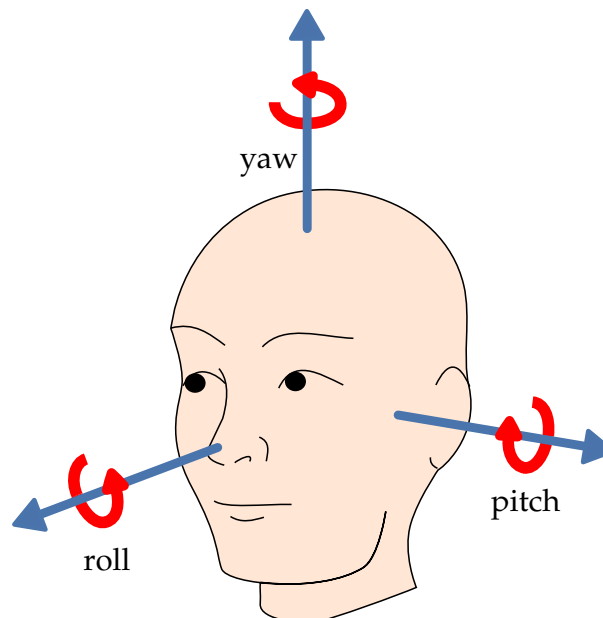


Figure 2.8. The three degrees of freedom: yaw, pitch, and roll.

Problem 2.2 (head pose estimation). *Let I be an image of a head H , determine the three angles (yaw, pitch, and roll) that compose the orientation of H with respect to the camera that acquired I ².*

Problem 2.2 is closely related to Problem 2.1, which consists on the face detection. A common solution for the Head Pose Estimation Problem starts with the face

²We found it convenient to represent orientation using Euler angles, however one might choose quaternions or other representations. Any object orientation format is applicable.

detection, although some approaches do not need this detection step. Many solutions have been created for solving Problem 2.2 and they were carefully categorized by Murphy-Chutorian and Trivedi [2009] into the following taxonomy:

- (i) **appearance template:** An image containing a head is compared against a set of previously labeled images with known head poses. The goal is to find the most similar pose of the queried image. This approach is suitable for both high or low resolution images.
- (ii) **detector array:** An image is analyzed by different face detectors that were trained for specific poses. An advantage of this approach is that a step for face localization is not required. Once trained, the algorithm is capable of distinguishing whether or not a face is present in the image.
- (iii) **nonlinear regression:** The solution is based on learning a nonlinear functional that maps the image space and the desired poses. An example would be Multilayer Perceptron and Locally Linear Mapping.
- (iv) **manifold embedding:** It is a technique that applies a dimensionality reduction algorithm (e.g. Principal Component Analysis (PCA)) for creating low-dimensional manifolds for modeling different face poses.
- (v) **flexible models:** This method tries to adjust a generic model to the input face by deforming the features of model. Pose is estimated using the adjusted features.
- (vi) **geometric methods:** These methods use a set of features, such as nose, mouth to infer the head pose. The estimation is based on the geometric configuration of those features.
- (vii) **tracking methods:** Methods that estimate the head pose based on the movement between frames.
- (viii) **hybrid methods:** Methods that combine those previous approaches.

Murphy-Chutorian and Trivedi [2009] provide a great discussion about advantages and drawbacks of those methods. For instance, they point out that detector array methods do not need an additional step for detection and localization due to the fact that the trained detectors are capable of distinguishing between face and non face. However, one clear disadvantage is the difficult training step that needs

to handle each discrete pose. Geometric models exploit the relations between facial features such as angles, distances, etc. Our idea for estimating the pose tries to explore facial features, but using deformable models in order to estimate a good model for each specific user.

Deformable models have been used before for estimating the head pose. Martins and Batista [2008] propose the use of Active Appearance Model (AAM) and POS with Iterations (POSIT). They use a three-dimensional scanner to obtain the points of a face model. First, they first adjust the AAM to the users face and using the model created with the scan, they estimate the pose using POSIT. Deja [2010] propose a head pose and gaze estimator. For solving the head estimation problem, he also applies AAM and POSIT.

2.2.2 Active Shape Model

In this work, we take advantage of deformable models for finding human facial markers in the image (*e.g.* eyebrows, nose, *etc.*). We briefly describe what general models are and the specific behaviour of Active Shape Model (ASM) as defined by Milborrow [2007].

Definition 2.1 (Model or Shape). *A shape is a set of (x,y) points (landmarks) located in an image that defines or represents a geometric figure.*

Figure 2.9 shows two different representations of a simple shape. In this shape, there are six points whose Delaunay triangulation generates a pentagon or five triangles. Those edges do not exist in the original shape, just the vertices. However, edges (or even triangles) help humans visualize shapes (Figure 2.9.a). The shape in Figure 2.9.b shows a typical representation for computer-related tasks, a $n \times 2$ matrix.

Similarly to the shape presented before, there are much more complex shapes describing facial features. For instance, the database *BioID* presented in Chapter 2 has been manually marked with 20 points, while the XM2VTS has been labelled with 68 points.

In his thesis, Milborrow [2007] first explains the concept of *template alignment*: an iterative method for producing the minimum distance between shapes. It is a set of transformations (scaling, rotation, and linear translation) applied to a template for matching a reference shape. For instance, the following equation transforms $P(x, y)$ into $P'(x, y)$ by applying a rotation (α) and a translation (δ_x, δ_y):

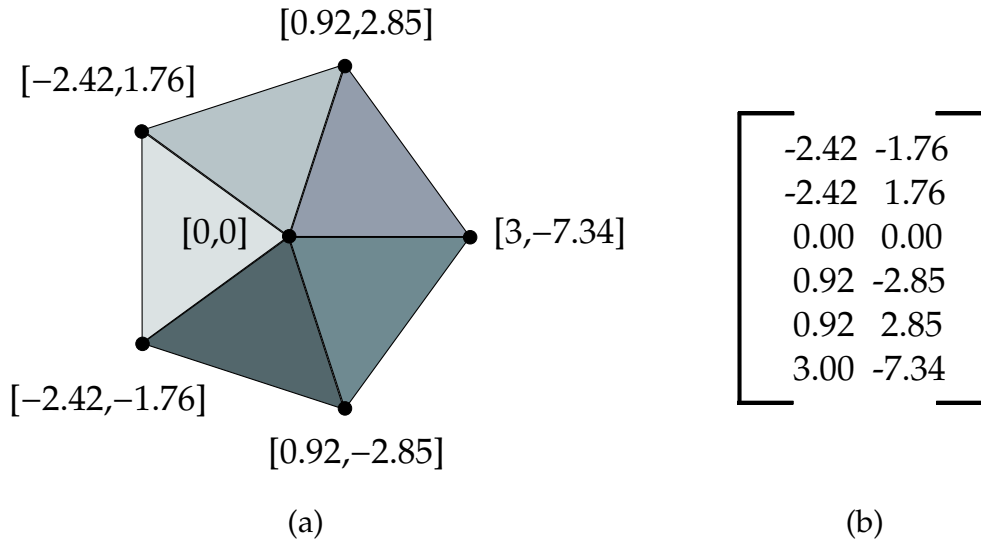


Figure 2.9. Six points shape. Left image (a) shows the Delaunay triangulation of the two-dimensional vertices and right image (b) shows a common matrix representation.

$$P \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} \delta_x \\ \delta_y \end{bmatrix} + \begin{bmatrix} \cos(\alpha) & -\sin(\alpha) \\ \sin(\alpha) & \cos(\alpha) \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

As many other deformable models, ASM requires an initialization close the real face. Hence, considering that ASM is initialized over a ROI face. Milborrow [2007] created two submodels:

- (i) **Profile Model:** It acts *locally* trying to find the best location for positioning a feature.
- (ii) **Shape Model:** It acts *globally* trying to correct the feature locations suggested by the profile model in order to adjust a possible face model.

A profile is based on training and searching. The training phase consists in averaging the area around each landmark for building a profile (specific profile for each landmark in all training images). The algorithm of Milborrow [2007] samples the area around the landmark (± 3 pixels) and tries to find the best match using Mahalanobis distance (d) between *profile* g and the *model mean profile* \bar{g} :

$$d = (g - \bar{g})^T S_g^{-1} (g - \bar{g}), \quad (2.7)$$

where S_g^{-1} is a covariance matrix.

The search phase consists in positioning the landmark in the best place that matches the profile built. The model tries to adjust the proposed model for conformity with allowable face shapes. The shape model (\hat{x}) consists of averaging the face with distortions:

$$\hat{x} = \bar{x} + \Phi b, \quad (2.8)$$

in which \bar{x} stands for the average of aligned trained shapes, Φ is the eigenvectors of the covariance matrix of training shape points.

Milborrow [2007] trains his algorithm using the XM2VTS database [Messer et al., 1999], he evaluates in the AR dataset [Martinez and Benavente, 1998] and he tests on BioID. He also extends the original 68 landmarks of XM2VTS to 76 and 84 dataset points, providing better accuracy on detection. Figure 2.10 illustrates the result of an ASM adjustment using Stasm [Milborrow, 2007]. In the left, the initial model applied over the region containing the face. This is really important for a successful match, in the right, the final adjusted face.

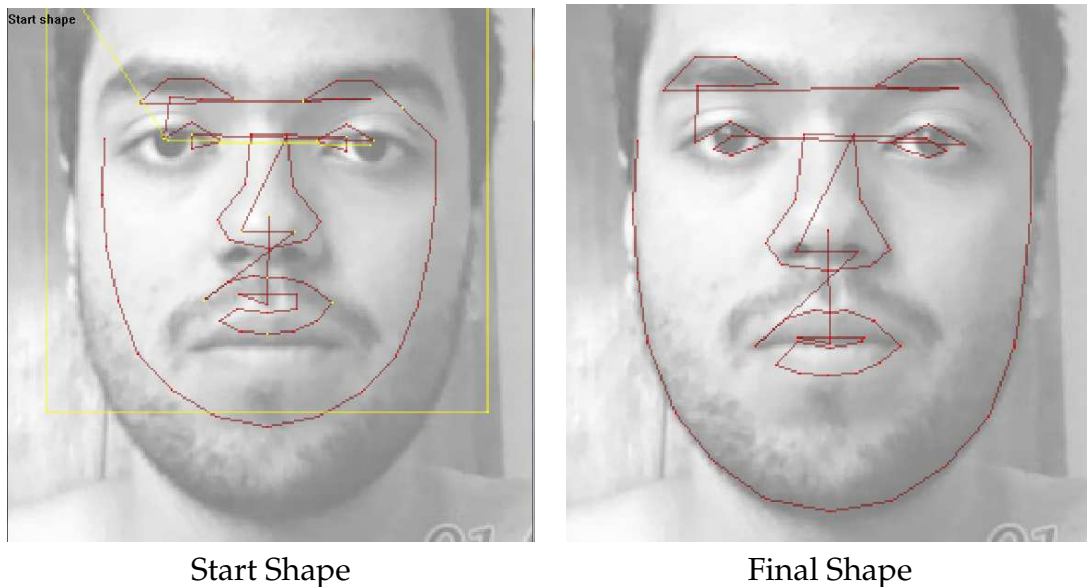


Figure 2.10. Results of applying ASM for adjusting the user's face with Stasm software [Milborrow, 2007]. On the left, the initial model and on the right the final model adjusted.

Chapter 3

Methodology

The gaze tracking problem is usually based on several subproblems. Our methodology for this problem is based on two main modules: (i) the head pose estimation module, and (ii) the iris segmentation module that will lead us to gaze estimation. Those modules are described in Figure 3.1.

Usually, before one targets its gaze to a certain direction, the face moves towards that direction for better accommodation although it is possible to move the eyes with fixed head. However, pose estimation becomes an important step in gaze tracking task.

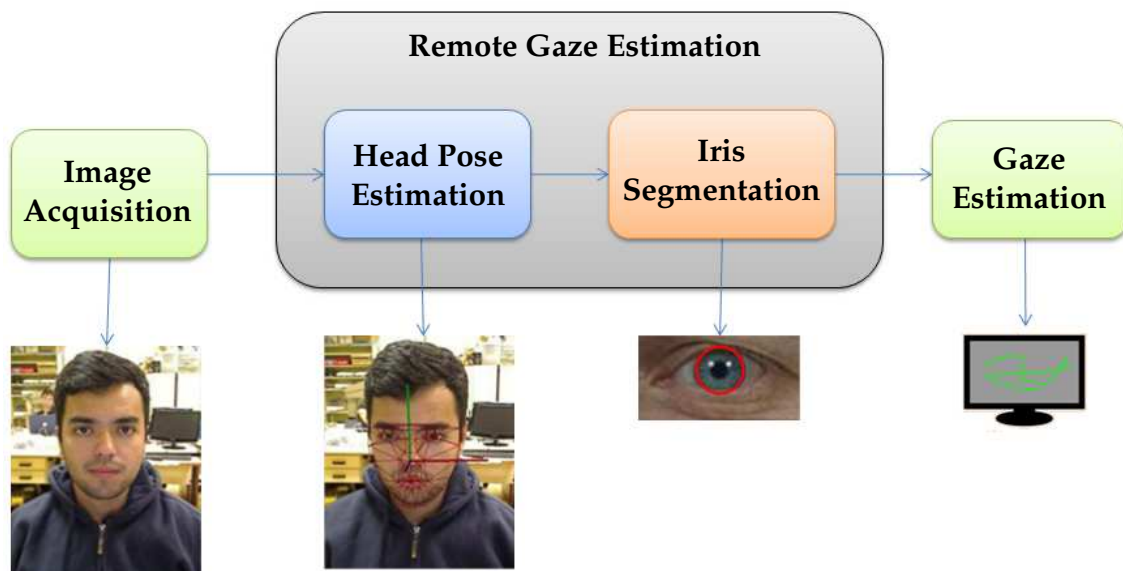


Figure 3.1. Overview of our methodology. The input is an image acquired by a camera. After pose estimation, we perform the iris segmentation and finally the system outputs the gaze location.

The input is the image acquired by a camera. After capturing the image, we start the first module for estimating the head pose and then, the second module detects the user iris. Finally, the system outputs the gaze location. It is also important to mention that before estimating the gaze, there is a calibration step for mapping the points in the image to points in the screen. Those modules are closely related, a failure in first module will affect the second one. For instance, the gaze is estimated based on the position of the iris which is, by definition, inside a ROI that was determined by the head pose. Thus, a failure in the head pose module will cause failure in the other modules in cascade.

Algorithm 1 describes all steps applied for the gaze estimation in a higher level. In this chapter, we explore each one of those methods explaining both the ideas and procedures to accomplish each task.

```

1 begin
2   Points model2D  $\leftarrow$   $\emptyset$ , model3D  $\leftarrow$   $\emptyset$ ;
3   Template eyeTplt  $\leftarrow$   $\emptyset$ ;
4   Point2D iris  $\leftarrow$   $\emptyset$ , gaze  $\leftarrow$   $\emptyset$ ;
5   while true do
6     Image I  $\leftarrow$  grabFrame();
7     if model is  $\emptyset$  then
8       model2D  $\leftarrow$  ASM(I);
9       model3D  $\leftarrow$  adjust3DModel(model2D);
10    else
11      model2D  $\leftarrow$  trackModel(I, model2D);
12    end
13    pose  $\leftarrow$  POSIT(model2D, model3D);
14    ROI  $\leftarrow$  findEyeROI(I, pose);
15    if eyeTplt is  $\emptyset$  then
16      iris  $\leftarrow$  conicFitting(ROI);
17      eyeTplt  $\leftarrow$  createTemplate(iris);
18    else
19      iris  $\leftarrow$  templateMatching(ROI, eyeTplt);
20    end
21    gaze  $\leftarrow$  estimateGaze(iris);
22  end
23 end

```

Algorithm 1: Algorithm for Remote Gaze Estimation

From lines 2 to 4, we initialize some sets. During the main loop, we process each frame by first estimating the pose (Lines 7 to 13). After that, we define the ROI of the eye (Line 14) and we estimate the eye location by either detecting or tracking

over the frames (Lines 15 to 20). Finally, we estimate the gaze (Line 21). However, for the gaze estimation step, it is required a calibration procedure that is discussed later.

3.1 Methodology Overview

Figure 3.2 introduces the stages that we apply for the first problem: the head pose estimation. First, it is required to detect a face in the image as stated in Problem 2.2. For the detection step, we are assuming a single frontal face looking towards the camera. In case of multiple faces, we will consider only the biggest face in the image, which is also the closest one to the camera. The algorithm of Viola and Jones [2004] was chosen for the detection stage.

After detecting the face, we crop the ROI and apply the ASM [Milborrow, 2007] for adjusting a generic face model into the specific user face. Now, considering we have specific features detected (mouth, eyebrows, chin), we no longer detect the face in the consecutive frame, but we *track* those features over the next images. This decision has been taken considering that face detection and ASM adjustment are slow activities to be performed every new frame. If we track those features, we obtain a faster solution.

ASM fits a two-dimensional model, however for estimating the pose, we need a three-dimensional model. Hence, we performed an offline stage (Section 3.2.2) called *Mesh Projection* for estimating the depth of the two-dimensional model adjusted by ASM.

Considering that so far we have obtained the points of the head model (deformed by some transformations), we can restore the pose of that object by using POSIT [Dementhon and Davis, 1995]. We use POSIT for restoring the *rotation matrix* and *translation vector* of the user head. The head pose can be estimated as long as there is a successful tracking. By successful, we mean a *correct consensus* in the movement of the image features, e.g: Two neighbor features in the image t should remain neighbors in the image $t + 1$.

Our second module contains the eye segmentation step for estimating the gaze. Figure 3.3 shows an overview of the module. The input for the eye segmentation step is the estimated face pose. For every new image acquired, we track the head features whose pose provides the location of both eyes.

The first step of this module consists in applying a compensation in the iris for decreasing the brightness of the eye in a similar way as the one proposed by

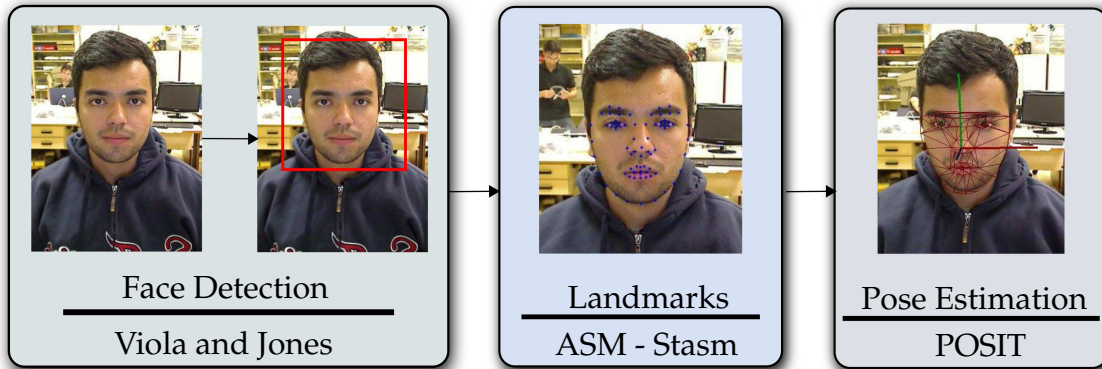


Figure 3.2. Methodology for head pose estimation is divided in three main stages: (i) the face is detected using Viola and Jones [2004], the landmarks representing the face features are obtained using ASM [Milborrow, 2007] and (iii) the pose is restored using POSIT [Dementhon and Davis, 1995].

Nguyen et al. [2010]. After being processed by compensation and equalization, the image is binarized. An implicit conic fitting is applied along with RANdom Sample Consensus (RANSAC) for estimating the iris. The whole process for iris detection is carefully discussed in Section 3.3.

The eye detection process occurs only once per eye due to real time requirements. After detecting the user iris, a template is created based on the specific properties of the eye. Hence, the eyes displayed in the subsequent images do not need to be compensated considering we know how the appearance of the iris is. The only required step is to find the best match for the template. This process is called *template matching*.

Taking into account that so far we know the current head pose and we have successfully obtained the iris location, we are ready to proceed to the final part of the problem: *gaze estimation*. The surface of the eye is not plane but curved. However, in this work, we can approximate the eye surface as a plane based on weak perspective. Thus, to estimate the gaze, we perform a calibration that maps points in the image into points on the screen (gaze locations). This calibration procedure pairs two point sets $P_{eye} = \{^1P_{eye}, ^2P_{eye}, \dots, ^n P_{eye}\}$ and $P_{screen} = \{^1P_{screen}, ^2P_{screen}, \dots, ^n P_{screen}\}$ that represent the location of the iris in the image and location of the gaze on the screen respectively.

Considering that we are trying to estimate the gaze based on the iris location, we calculate the *homography* between those two point sets.

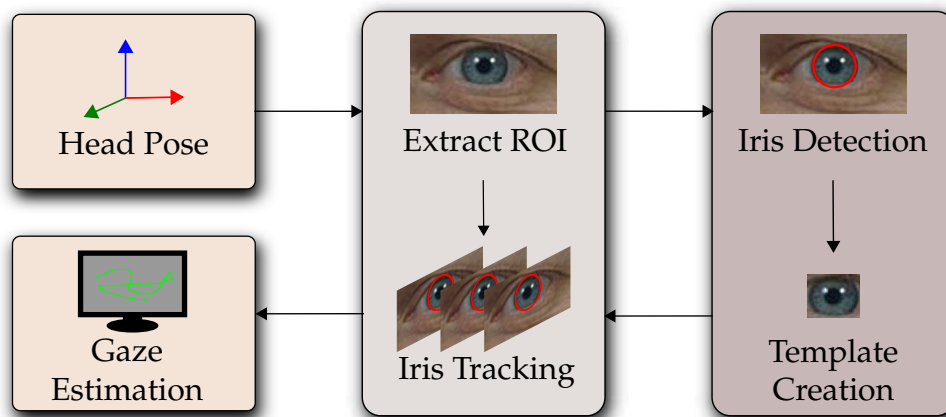


Figure 3.3. Methodology for gaze estimation: the ROI that contains the eye is obtained based on the estimated head pose. The iris is detected using implicit fitting and a template is created for the specific eye. After the template creation, the subsequent frames are tracked using template matching. Finally, the gaze is estimated using homography.

3.2 Head Pose Estimation Algorithm

Our algorithm estimates the face pose based on facial landmarks. The landmarks localization is performed with ASM introduced by Cootes et al. [1995]. We apply the ASM developed by Milborrow and Nicolls [2008] to find those points. After the successful initialization of the model, the points are tracked using Lucas and Kanade [1981] (LK) pyramidal optical flow.

A three-dimensional model is required for the head pose estimation. Hence, the two-dimensional points adjusted by ASM were manually projected to a generic three-dimensional mesh (Section 3.2.2).

Considering that we have a model for the possible head, we apply an iterative method called POSIT [Dementhon and Davis, 1995] for estimating the pose of the points related to the model we built. POSIT gives us the transformation of the pose (rotation and translation matrices).

3.2.1 Tracking of Facial Landmarks

Due to real time constraints, it is expensive to perform another ASM fitting for every new frame in order to find the landmarks. Hence, we decided to track those points over the sequence using Optical Flow.

The tracking task is based on both *spatial* (x, y) and *temporal* (t) information. Thus, considering that the brightness (B) is a function of (x, y, t) , the optical flow

assumes that B is constant and differentiable over time. Hence:

$$\frac{dB(x(t), y(t), t)}{dt} = \frac{\partial B}{\partial x} \frac{dx}{dt} + \frac{\partial B}{\partial y} \frac{dy}{dt} + \frac{\partial B}{\partial t} = 0, \quad (3.1)$$

where the partial derivative components of this equation ($\frac{\partial B}{\partial x}, \frac{\partial B}{\partial y}$) correspond to the spatial gradient ∇B and the total derivative ones ($\frac{dx}{dt}, \frac{dy}{dt}$) correspond to the vector field v . Then, the *Image Brightness Constancy Equation* as shown by Trucco and Verri [1998] is:

$$(\nabla B)^T v + B_t = 0, \quad (3.2)$$

which relates the image brightness $B(x, y, t)$ with the motion field v . We apply the pyramidal algorithm (LK) of Lucas and Kanade [1981] for tracking the 76 markers presented in the previous section.

Roughly, LK relies on three important assumptions: The brightness of the tracked pixels should remain constant and the points moving slowly from frame to frame. Finally, it is assumed a spatial relationship between points, considering that they belong to the same surface. Figure 3.4 shows two different frames whose landmarks were tracked using LK algorithm.

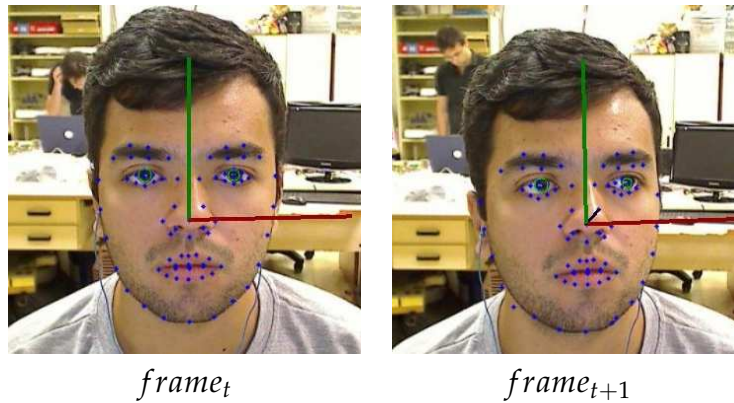


Figure 3.4. Lucas and Kanade [1981] algorithm for tracking the blue dots that are used for pose estimation.

3.2.2 Mesh Projection and Pose Estimation

Before explaining the pose estimation, we discuss the *mesh projection*, which is an offline stage where we try to obtain a coarse depth estimation of those 76 features adjusted by the ASM model.

We apply POSIT [Dementhon and Davis, 1995] to retrieve the transformation of the head. This algorithm requires a set of points describing an object under some perspective (the points we adjust with the deformable model). Moreover, we need to provide the model of the object whose pose we are trying to estimate. To build this model, we obtained a three-dimensional head using the makehuman¹ software and we adjusted the ASM model to the frontal projection of the face. After the adjustment, we manually extracted the depth for each one of those points to create the template in Figure 3.5.

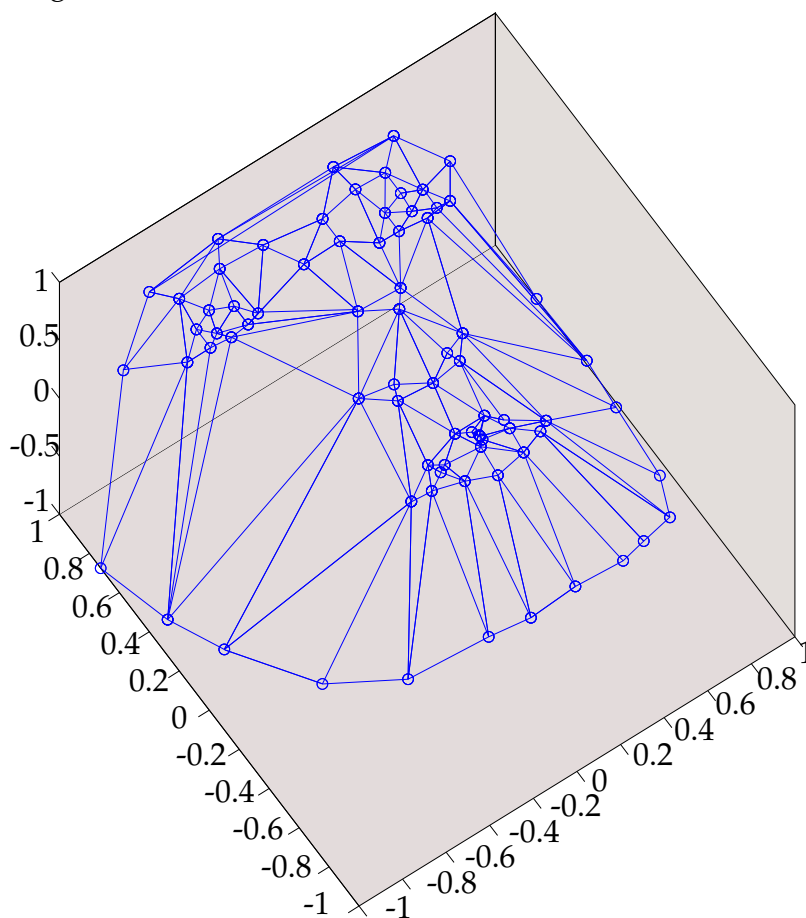


Figure 3.5. Mesh Projection: Merging three-dimensional information.

When ASM fits the generic model into the user face, we no longer have a generic model, but an adjusted model which is unique for each user. In order to extend this specific model to the three-dimensional model, we assume that the maximum depth of the face (nose to ear distance in profile view) is proportional to the distance from the chin to eyebrow (in frontal view) as described in Figure 3.6. Hence, we interpolate the depth value of the mesh based on this proportion. This idea is

¹<http://www.makehuman.org/>

based on the study of divine proportion in human anatomy described in [Jefferson, 2004]. Our experiments seem to support that this chin-eyebrow proportion provides good results on pose estimation.

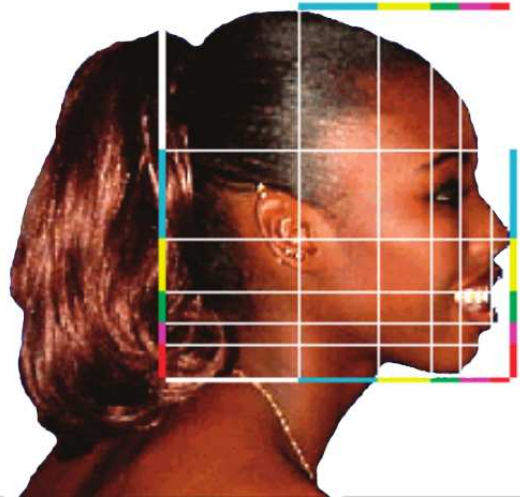


Figure 3.6. We consider that the chin to eyebrow distance is proportional to nose to ear distance. Image extracted from [Jefferson, 2004].

POSIT is divided into two algorithms:

1. **Pose from Orthography and Scaling (POS):** the goal of this algorithm is to approximate the perspective projection using a scaled orthographic projection. It also estimates the rotation matrix and the translation vector.
2. **POS with Iterations (POSIT):** the goal of this algorithm is to compute iteratively a better scaled orthographic projection using the proposed pose found by POS instead of the original image.

Finally, after initial adjustment, we can track the points and estimate the pose for every new frame acquired by the camera. As soon as the pose is obtained, we set a ROI for the eyes and start the eye segmentation algorithm.

3.3 Eye Segmentation Algorithm

The face pose allows us to crop a Region of Interest (ROI) around each eye. For the initial iris detection, we first start by compensating the iris region, which aims to enhance some properties of the eye. The image is then binarized and sent to a circle detector algorithm. After that, we track the known iris pattern.

Algorithm 2 describes how we segment the iris.

```

input : Eye ROI
output: Iris  $x,y,r$ 
1 begin
2    $Template \leftarrow \emptyset$ ;
3   if not  $Template$  then
4      $ROI_b \leftarrow \text{binarize}(ROI)$ ;
5      $C \leftarrow \text{compensate}(ROI_b)$ ;
6      $x,y,r \leftarrow \text{conic\_ransac}(C)$ ;
7      $Template \leftarrow \text{obtain\_template}(ROI, x, y, r)$ 
8   else
9      $x,y,r = \text{track}(Template, ROI)$ ;
10  end
11 end

```

Algorithm 2: Iris Detection and Tracking Algorithm.

3.3.1 Iris Compensation

We are dealing with images acquired in visible light. The eye's surface reflects lights that come from many different sources. Specifically, the lights in the scene generate glints on the iris and the sclera surfaces. Those reflections make reliable eye segmentation difficult. Figure 1.1 shows an example of an eye with specular reflections.

Problem 3.1 (Iris Compensation). *Let I be an image that contains an eye. Find the region $w \in I$ which most likely represents the iris. Reduce the brightness of w by decreasing the light reflections on the cornea surface and increasing the contrast between iris and sclera (limbus).*

When we apply ASM for detecting the head features, we obtain several features delimiting the eye region (such as eye corners, etc.). So, based on the location of those features, we define ROI and we create a mask that will be applied to the image for enhancing the iris region. The first amelioration that we perform is the *histogram equalization*. We choose to work with the red channel in the RGB space, because the sclera is better visualized in this channel [Parker and Duong, 2009].

Figure 3.7 shows the original image, the histogram equalization of the red channel (E), and the mask (\mathfrak{R}) that is going to be applied in the image. Our algorithm works as follows, we should erase pixels that do not belong to the sclera (S),



Figure 3.7. Each row shows the original image ROI, the image after histogram equalization, and the mask applied for decreasing the brightness of the iris. In this example, we see the results for a black and a blue eye.

iris (I), and pupil (P) since they have no relevance to the problem. Pixels belonging to I and P should have their brightness reduced, so:

$$E(x, y) = E(x, y) - \mathfrak{R}(x, y).$$

Considering that each channel is composed of unsigned 8 bits, the subtraction will set pixels outside sclera into zero due to the fact that $\mathfrak{R}(x, y) = \{255 \mid \forall(x, y) \notin S(x, y) \cup I(x, y) \cup P(x, y)\}$. As one might notice, the region $\mathfrak{R}(x, y) = \{0 \mid \forall(x, y) \in S(x, y)\}$. Thus, the difference presented in the previous equation will not interfere in the original region of the sclera. Finally, the region $\mathfrak{R}(x, y) = \{\tau \mid \forall(x, y) \in I(x, y) \cup P(x, y)\}$ has a value τ that is responsible for decreasing the brightness.

Figure 3.8 shows three images describing the enhancing process. The first image shows the subtraction of the equalized image by the mask introduced before. By doing that, the brightness of the iris region is decreased. After that, the whole image is binarized. However, there are still holes generated by glints and reflections. We have then applied an algorithm for filling those holes and we prune the irises using Mathematical Morphology (MM) that is discussed in the Appendix A .

The idea behind this is that dilation might connect noisy pixels to the iris. Then, we first apply the erosion operation in the image. However, if there are many holes generated by glints, the iris will break apart destroying the circular shapes. Thus, we first fill all possible holes and then we start applying erosion for removing noisy pixels without losing the circular shape of the iris.



Figure 3.8. Each row shows the image ROI for the black iris and the blue iris showed in Figure 3.7. The first column shows the ROI after the mask subtraction, the second column shows the result the binarization result, and the last column shows the final result after hole filling and morphological operations.

3.3.2 Circle Detection

Once we obtain the binary image, we need to estimate the center and radius of the iris. For reaching that goal, we apply implicit conic fitting.

Appendix B provides the background for understanding conics and quadrics. In this section, we demonstrate how to perform a conic fitting using Singular Value Decomposition (SVD). The technique for estimating quadrics is very similar, however, we focus on the implicit estimation of a conic (circle).

Given the equation of circle with center (a, b) and radius r :

$$r^2 = (x - a)^2 + (y - b)^2,$$

we can expand it:

$$\underbrace{1}_A x^2 + \underbrace{1}_B y^2 - \underbrace{2a}_C x - \underbrace{2b}_D y + \underbrace{a^2 + b^2 - r^2}_E = 0$$

if we take advantage of the following notation ($P = Q^t \cdot S$), we have:

$$Q^t = [x^2, y^2, x, y, 1], \text{ and}$$

$$S^t = [A, B, C, D, E].$$

Thus, our problem is now the estimation of S . We solve this problem using Singular Value Decomposition. Once we obtain S , we can retrieve a , b , and r .

3.3.3 Consensus of Circle Fitting

In order to provide robustness to our approach, the implicit conic fitting is applied along with RANSAC [Fischler and Bolles, 1981]. RANSAC is a technique for fitting (experimental) data to a model when there are outliers. We have made that decision due to the fact that even after morphological operations, we still have some noisy

pixels that might disturb the circle estimation process.

Roughly, instead of using all data points to fit the model, RANSAC starts with a small random subset. Algorithm 3 demonstrates the usage of RANSAC along SVD for estimating the best circle in the image.

The input of the algorithm contains: *(i)* the points that we want to fit (P), *(ii)* the number of iterations we want to perform (k), *(iii)* the minimum points (m) inside the inliers group to consider this model as a good one, and finally, *(iv)* the threshold (t) for allowing a point to enter the inliers group. Line 2 initializes model, inliers and error sets. Lines 4 to 6, we randomly pick a subset of the original points and we estimate the circle (M_C) for those points (lines 7 to 10).

When the initial model is created, we test the other points that are not in our initial set. If those other points are closer enough to the circle (i.e. distance is smaller than t), we add this point to the inliers group (I_C). If the number of inliers are greater than m , it means we found a good model, and finally, if our new model has a smaller error than the current model, we reassign the previous data (model, inliers,

and error).

```

Data:  $P = \{p_1, \dots, p_n\}$ ,  $p_i$  is the  $i^{\text{th}}$  point in  $P$ ;
Result: the best model (center  $(a, b)$  and radius  $r$ ) for the circle;
input : iterations  $k$ , threshold  $t$ , min points  $m$ 
output: best model  $M$ , inliers  $I$ , and best error  $\epsilon$ 
1 begin
2    $I \leftarrow \emptyset; M \leftarrow \emptyset; \epsilon \leftarrow \infty;$ 
3   for  $i \leftarrow 1$  to  $k$  do
4     for  $z \leftarrow 1$  to  $5$  do
5        $I_c \leftarrow I_c \cup \{P_r\};$  //  $r$  is random number
6     end
7      $Q \leftarrow I_c \times I_c^T;$ 
8      $U\Sigma D^T \leftarrow \text{SVD}(Q);$ 
9      $\epsilon_c \leftarrow \Sigma_{5,5};$ 
10     $M_c \leftarrow U_{1:5,5};$ 
11    for  $p_j \in P$  and  $p_j \notin I_c$  do
12      if  $\text{distance}(p_j, M_c) < t$  then
13         $I_c \leftarrow I_c \cup \{p_j\};$ 
14      end
15    end
16    if  $\|I_c\| \geq m$  then
17      if  $\epsilon_c < \epsilon$  then
18         $I \leftarrow I_c;$ 
19         $M \leftarrow M_c;$ 
20         $\epsilon \leftarrow \epsilon_c;$ 
21      end
22    end
23  end
24  return  $I, M, \epsilon;$ 
25 end

```

Algorithm 3: Circle Detection Algorithm.

Figure 3.9 shows the results of applying RANSAC along implicit fitting for estimating the best iris in the image. The result represents the circle (green) describing the best iris found.

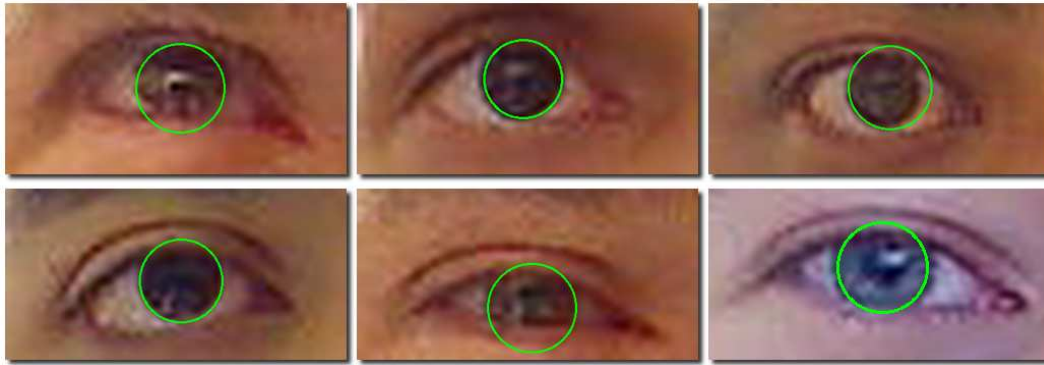


Figure 3.9. Results of Iris Detection

3.3.4 Iris Tracking

Once we have found the user iris, we track it for avoid the computational cost of compensating the image again, running RANSAC and all those previous steps. Our assumption is based on the fact that the user iris does not change too much during the process, the iris in frame $t + 1$ has a strong relation with the iris in frame t . Therefore, it is suitable to create an image patch with discovered iris for tracking it along frames.

The technique known as *template matching* computes the difference between an image (patch) against another image (the one that we are trying to find that patch) by sliding the patch over it. Figure 3.10 shows how the matching is performed.

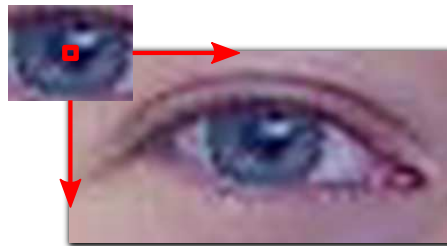


Figure 3.10. Template Matching: the patch is slid over the image searching for the best match.

There are many different ways for calculating the difference between the two images: Squared Difference, Correlation Difference, Correlation Coefficient Difference, and so on. We decide to use the squared difference. Let F be the final image, P the iris patch used and I the eye ROI. The squared difference is computed as follows:

$$F(x, y) = [P(x', y') - I(x + x', y + y')]^2 \quad (3.3)$$

Hence, when the iris is detected, a template is created using the region over the iris. Thus, in the subsequent images, instead of performing all those steps, we compute the squared difference between the patch and the image to find the best match for the user iris.

3.4 Gaze Estimation Algorithm

We apply homography technique to estimate the gaze by mapping points in the image into points in the screen. Deja [2010] has also applied homography for gaze estimation. It requires a calibration procedure to pair up those two planes (image and screen). This calibration contains a pattern (Figure 3.11) that guides the user during the procedure. This idea is similar to the EyeWriter software described in Chapter 2.

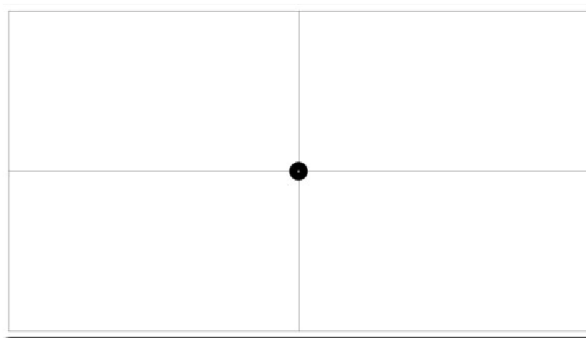


Figure 3.11. Calibration Pattern: while calibrating, the user has to look at corner of this grid. A black circle with changeable radius appears showing the next location for fixation and when the red circle appears, the location of iris in the image is grabbed.

At each grid intersection, a black circle with a changeable radius size appears indicating the next calibration location. When the black circle becomes a red circle with fixed size, the two points are being paired up and the user cannot blink, otherwise it will pair wrong iris values. The grabbing time is about three seconds, which means that the user cannot blink during three seconds for each one of those nine points. This process is really important for the estimation, considering that a poor calibration will lead to a poor gaze estimation.

The homography procedure becomes easier to be understood when looking at the system configuration displayed in Figure 3.12. The user stands in the front of the screen and camera. The camera can be placed either at the top of the screen or at

the bottom. During our experimental step, we found that when the camera placed at the bottom, it presents better results because there less occlusion by eyelids.

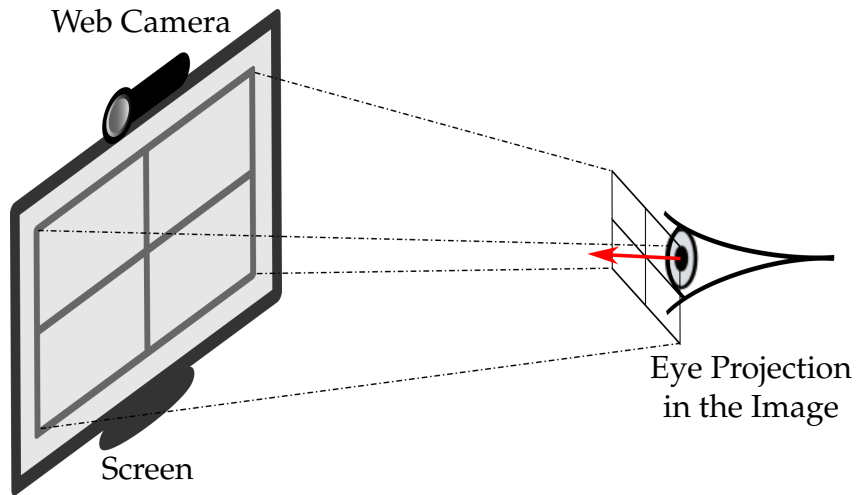


Figure 3.12. Calibration Procedure: the user stands in front of the camera and the screen that contains the grid. A homography is created mapping the eye location in the image and the gaze in the screen.

Figure 3.13 shows a good calibration example. All nine points were successfully obtained.

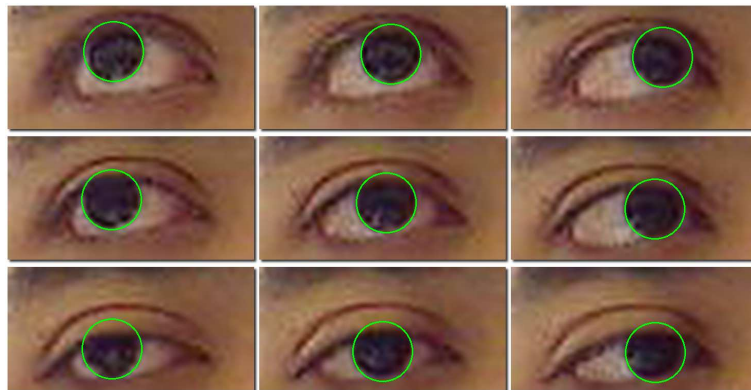


Figure 3.13. Example of a good calibration. All nine points were successfully detected. As one can notice, the displacement between those points is really subtle.

Chapter 4

Experimental Analysis

In this chapter, we describe the results obtained by applying the methodology discussed in this work: the head pose estimation and the gaze tracker.

This chapter is organized as follows: we first present the qualitative results, where we focus on the development of different applications that could take advantage of each technique. After that, we explain how we have conducted our quantitative results and the uncertainty associated with the gaze estimation. Finally, we discuss the limitations of our approach.

4.1 Experimental Setup

This section clarifies the setup of our system. Hence, we describe the libraries (software) and devices (hardware) that were used during the development of the main module and additional hardware devices (such as robots and servomotors) that were used during the experimental phase of this work.

4.1.1 Software

Our software was developed under the *Linux* Operating System (but it is not restricted to it) using the C++ language with the help of the following libraries:

- **OpenCV (<http://opencv.willowgarage.com>):** A widely used computer vision library which contains many algorithms for handling image processing and vision problems.
- **OpenGL (www.opengl.org):** A library for rendering three-dimensional models, such as the axes of head direction and the face mesh. It allows one to

define the location of a virtual camera pointing towards a specific direction where many models (e.g. triangles, meshes, points, *etc.*) can be visualized.

- **Boost (www.boost.org):** A set of C++ libraries (e.g. Thread, System, Filesystem) containing many helpers classes and functions for speeding up the software development process. For instance, our eye tracker was working with threads available in Boost.
- **Golld¹:** OpenCV retrieves images from a camera. However, currently it is not possible to change specific hardware controls, such as enabling or disabling white balance, gain, exposure, and so on. Hence, Golld is a library developed in VeRLab at UFMG that allows one to set a specific hardware configuration.
- **Video for Linux 2 (V4L2):** A a video capture programming interface for the linux operating system. Golld is based on V4L2.
- **Stasm [Milborrow and Nicolls, 2008]:** A the Active Shape Model (ASM) library for shrinking the generic model into a specific user face.
- **GSL (www.gnu.org/s/gsl):** stands for GNU Scientific Library (GSL) and it is a numerical library that was used in this work for handling linear algebra problems.
- **QT (qt.nokia.com):** A the graphical user interface applied for rendering windows, buttons and so on. For instance, the calibration procedure required to drawing of lines, circles and animation in the screen. Those ones were totally rendering in QT windows.
- **QWT (qwt.sourceforge.net):** A QT library extension for rendering graphics (line, histogram plots). In this work, QWT was used to provide the scatter plot after completing the calibration procedure in order to know the quality of results obtained.
- **ImageMagick (www.imagemagick.org) :** A library that contains many algorithms for image manipulation and drawing that was used for drawing information inside images, overlapping images, and so on.

¹www.verlab.dcc.ufmg.br

4.1.2 Hardware

The approach presented in this dissertation used only one camera. The chosen camera was the Logitech 9000 with resolution of 800×600 pixels standing about 50 cm from the user. We have disabled most of automatic control, as gain and auto white balance. This camera is presented in Figure 4.1 along with the robotic head developed. The computer processor was an Intel Core 2 Duo 2.1 Ghz.

Later in this chapter, we introduce the remote control of a robotic head. For this robotic head, we have used two Logitech 9000 cameras and two servomotors. The images grabbed by the camera were delivered to the user through UDP network protocol. During the experiments, the camera was standing over a pionner robot.

The calibration procedure requires the user to look to a computer screen. In our experiments, it was a 17 inches screen with resolution 1600×900 .

4.2 Qualitative Results

In this section, we present two applications of the techniques developed in this work. One application uses gaze tracking while the other is advantageous only for pose estimation.

4.2.1 Heatmap of Gaze Location

The first application we have developed for validating the proposed approach was the generation of a *heatmap*. A heatmap is a visualization technique in which some regions are highlighted by overlapping color intensity maps. In our case, we have tracked the user's gaze and highlighted the regions where he focused its attention.

Our algorithm displays the gaze locations computed when the user was observing the image content. Regions that the user spent more time are more likely to be red. On the other hand, regions that the user spent less time will be green.

The main idea behind this application is the fact that for some studies, it might be extremely relevant to understand what is attracting the human attention, either to study the basis of image understanding, but also, it may be an indication of which properties of an image affect the interaction with a computational system.

Hence, heatmap is an adequate visualization technique to combine both information: spatial locations and fixation duration. The spatial location is represented by image locations where the user focused his gaze. The fixation duration is associated to the color of those locations in the map.

4.2.2 Remote Control of a Robotic Head

Tele-immersion is one of the several research fields that could take advantage from the results of the work presented here. There has been a lot of effort on remote control of robots, e.g. remote surgeries, rescue robotics, surveillance. For instance, Fernandes et al. [2011] have controlled a robotic head using an Inertial Measurement Unit (IMU) as displayed in Figure 4.1.



Figure 4.1. Robotic Head

Our robotic head is composed of two servomotors and two Logitech 9000 cameras and it is mounted on a Pioneer robot (Figure 4.2). It has two degrees of freedom (pan and tilt) and we can be remotely controlled using our head pose estimator. When the head pose of the user is obtained, it is transmitted to the robot which will reproduce the behaviour using the robotic head. Figure 4.3 shows the user moving the robotic head using his pose.

Another improvement would be to calibrate that stereo system for allowing reconstruction of a remote environment prioritizing the region the user is looking. Besides that, by combining both images, one could use a three-dimensional stereo vision to simulate the environment that the robot is inserted.

4.3 Quantitative Analysis

Our quantitative analysis approach is based on the results obtained by an application where some points are displayed on the screen and the gaze is estimated during three seconds for each point. By using this approach, we can obtain the accuracy



Figure 4.2. Tele-immersion robot used in experiments.

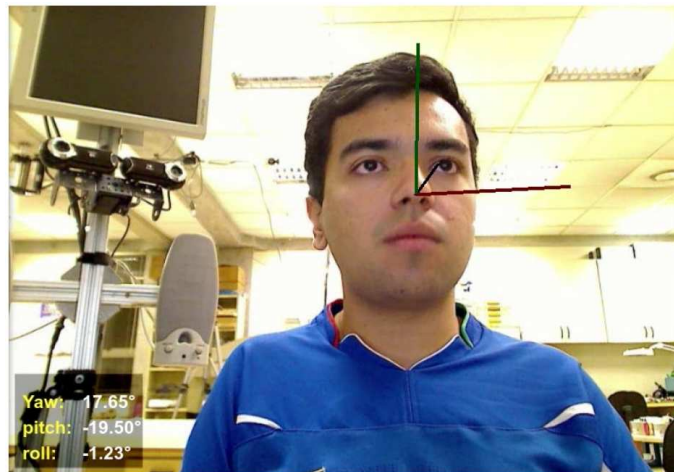


Figure 4.3. Robotic Head: two cameras were attached to two servomotors for providing the yaw and pitch degrees of freedom.

and precision of the system. We present the results we have obtained as well as the uncertainty associated with each user in both axes for all points.

The process occurs as follows: users use a chin rest for stabilizing the head position. The system is initiated obtaining the face pose. When the user feels comfortable with the system, the calibration procedure starts. The user looks at ten points on the screen for three seconds. Also, it was given three seconds for resting the eye. During this resting time, the user is allowed to blink, however during the capturing time, the user cannot blink neither move his head. We have tested the system with seven users whose skin colors vary from Caucasian to a darker skin (Latin) with light brown and dark black eyes.

Although we obtain the head pose, we were unable to estimate the gaze and allow free head motion at the same time. Thus, it is really important to mention that the results were obtained using a chin rest for holding the users head, otherwise the results presented here would not consider the uncertainty associated with the algorithm, but the uncertainty associated with how long the user can stay still without moving his head.



Figure 4.4. Overview of Individuals. Seven individuals were chosen and they are using a chin-rest during the experiments.

Figure 4.4 shows the users during experimental evaluation. Users are looking at the top of the screen. Most individuals have dark colored eyes, which turns the pupil detection in visible light a really hard task. Hence, using visible light we

Id	Point	Mean	SD Row	SD Col	Err Row	Err Col
1	(200,200)	(131.614,216.055)	22.062	30.872	68.386	16.055
2	(200,400)	(169.977,372.211)	10.290	12.998	30.023	27.789
3	(200,600)	(180.032,595.909)	22.423	18.855	19.968	4.091
4	(600,200)	(606.913,224.814)	12.075	33.563	6.913	24.814
5	(600,400)	(599.360,395.442)	21.097	14.242	0.640	4.558
6	(600,600)	(606.452,584.591)	17.548	33.829	6.452	15.409
7	(1000,200)	(993.268,194.902)	36.651	25.197	6.732	5.098
8	(1000,400)	(973.932,398.385)	18.739	17.815	26.068	1.615
9	(1000,600)	(966.821,595.219)	27.150	22.523	33.179	4.781
10	(1400,200)	(1425.927,209.254)	35.029	23.189	25.927	9.254
11	(1400,400)	(1389.441,411.427)	39.486	23.797	10.559	11.427
12	(1400,600)	(1380.071,591.811)	30.406	26.405	19.929	8.189

Table 4.1. Quantitative Results for Individual 1

Id	Point	Mean	SD Row	SD Col	Err Row	Err Col
1	(200,200)	(-60.178,144.838)	10.993	10.535	260.178	55.162
2	(200,400)	(78.533,402.470)	26.996	17.809	121.467	2.470
3	(200,600)	(-129.165,642.806)	63.276	22.047	329.165	42.806
4	(600,200)	(512.367,172.542)	19.985	54.221	87.633	27.458
5	(600,400)	(550.538,319.736)	22.002	29.502	49.462	80.264
6	(600,600)	(603.151,587.753)	52.059	10.839	3.151	12.247
7	(1000,200)	(965.478,202.038)	18.628	11.336	34.522	2.038
8	(1000,400)	(953.926,377.338)	18.653	18.514	46.074	22.662
9	(1000,600)	(1094.450,623.324)	32.929	35.501	94.450	23.324
10	(1400,200)	(1486.091,132.511)	40.263	16.690	86.091	67.489
11	(1400,400)	(1510.403,385.137)	60.059	20.260	110.403	14.863
12	(1400,600)	(1560.118,508.448)	68.624	37.147	160.118	91.552

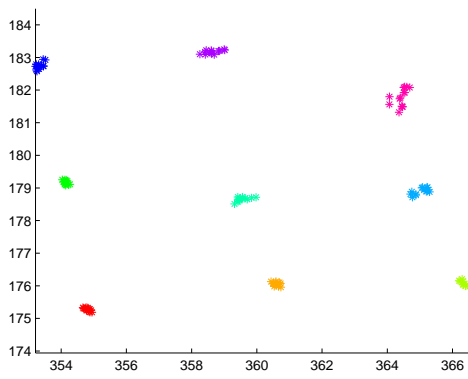
Table 4.2. Quantitative Results for Individual 2

need to identify the boundaries of the irises. However, some users have their irises slightly blocked by eyelids leading to more complexity.

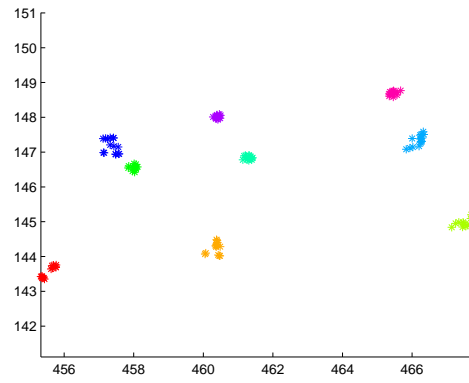
Calibration is one of the most important steps during the gaze estimation. Based on the calibration, we can estimate the user's gaze either with poor or good accuracy. In Figure 4.5 we show the calibration results for the seven users introduced here.

4.3.1 Gaze Measurements

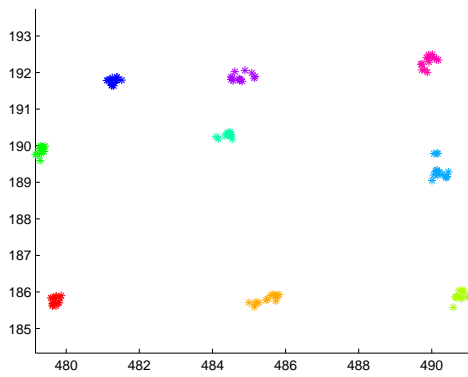
Figure 4.5 shows the center of the iris in the image coordinate system when users were looking at the calibration points in the screen. The horizontal and vertical axes



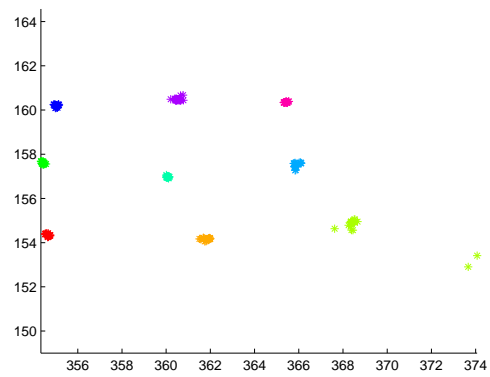
Individual 1



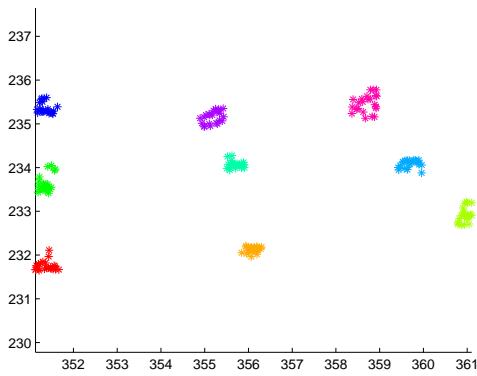
Individual 2



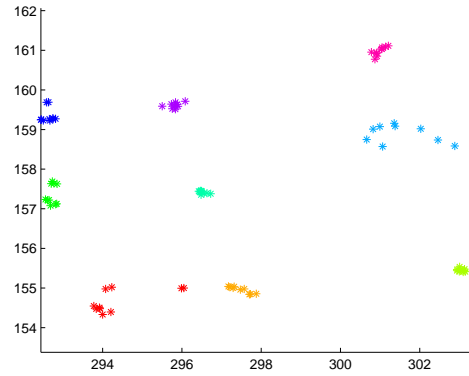
Individual 3



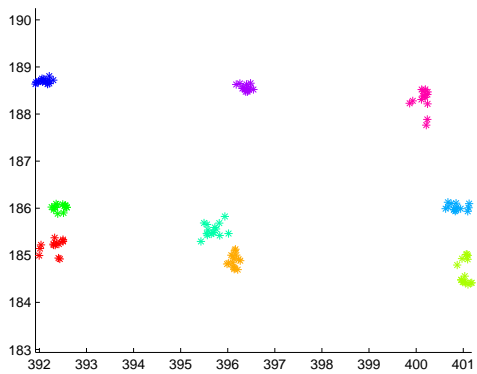
Individual 4



Individual 5



Individual 6



Individual 7

Figure 4.5. Calibration Results. All results use pixels as units of measurement.

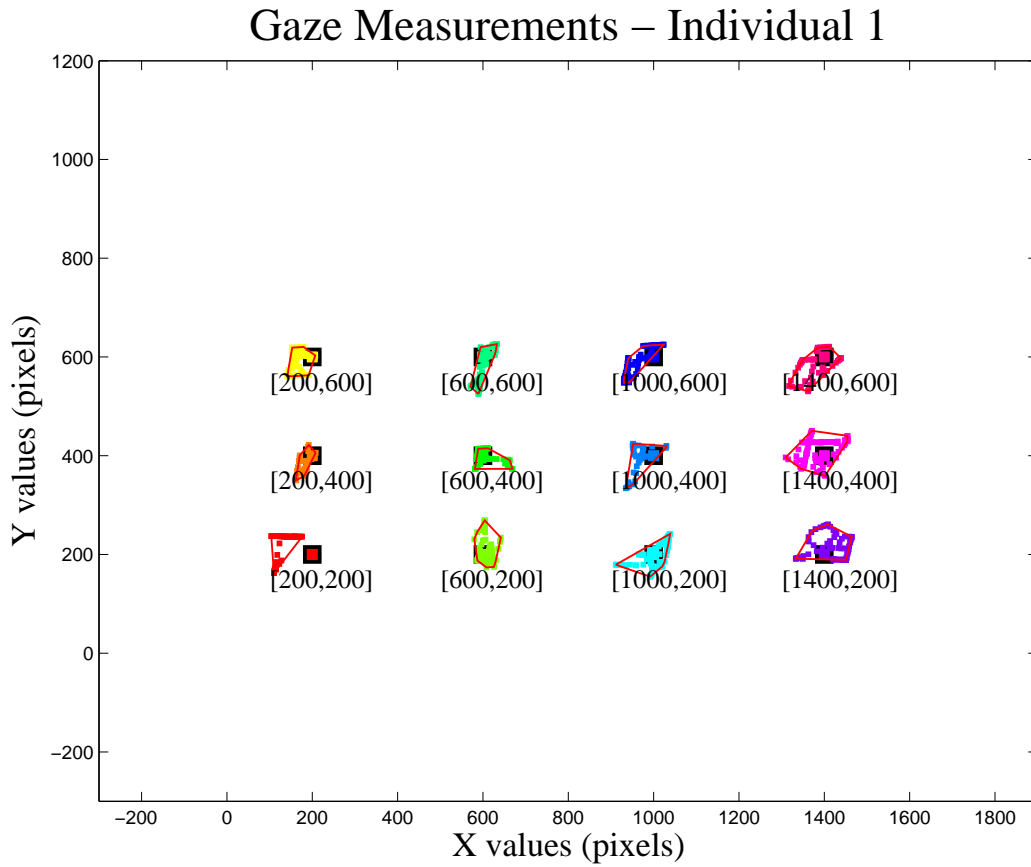


Figure 4.6. Gaze results of Individual 1

Id	Point	Mean	SD Row	SD Col	Err Row	Err Col
1	(200,200)	(142.239,190.214)	22.989	6.201	57.761	9.786
2	(200,400)	(-3.164,416.743)	17.680	10.152	203.164	16.743
3	(200,600)	(129.689,549.464)	28.927	29.837	70.311	50.536
4	(600,200)	(611.784,165.926)	47.375	17.082	11.784	34.074
5	(600,400)	(408.927,466.807)	0.983	9.338	191.073	66.807
6	(600,600)	(610.500,594.624)	28.238	19.509	10.500	5.376
7	(1000,200)	(1019.238,115.202)	27.787	23.119	19.238	84.798
8	(1000,400)	(866.498,480.524)	29.135	18.960	133.502	80.524
9	(1000,600)	(1031.223,594.839)	21.916	9.569	31.223	5.161
10	(1400,200)	(1530.578,114.990)	41.904	17.720	130.578	85.010
11	(1400,400)	(1485.639,334.262)	34.329	11.091	85.639	65.738
12	(1400,600)	(1676.977,617.480)	43.176	17.053	276.977	17.480

Table 4.3. Quantitative Results for Individual 3

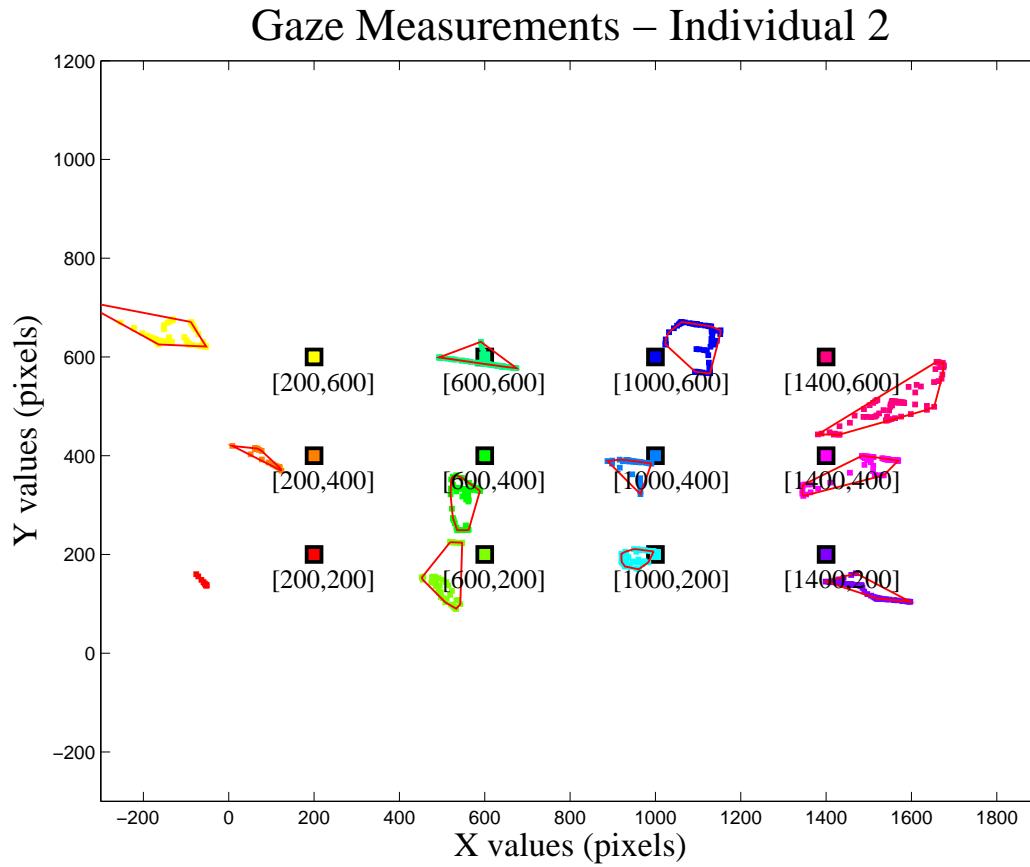


Figure 4.7. Gaze results of Individual 2

Id	Point	Mean	SD Row	SD Col	Err Row	Err Col
1	(200,200)	(110.113,147.994)	0.387	0.019	89.887	52.006
2	(200,400)	(101.208,383.863)	21.568	3.416	98.792	16.137
3	(200,600)	(0.000,683.429)	0.000	16.716	200.000	83.429
4	(600,200)	(598.112,154.785)	8.735	9.676	1.888	45.215
5	(600,400)	(525.159,340.260)	28.122	10.105	74.841	59.740
6	(600,600)	(404.204,671.818)	17.925	37.546	195.796	71.818
7	(1000,200)	(912.656,174.348)	22.232	9.530	87.344	25.652
8	(1000,400)	(838.786,361.253)	20.475	17.337	161.214	38.747
9	(1000,600)	(896.948,674.977)	8.222	6.548	103.052	74.977
10	(1400,200)	(1376.000,175.000)	0.000	0.000	24.000	25.000
11	(1400,400)	(1290.507,370.913)	6.195	17.383	109.493	29.087
12	(1400,600)	(1287.062,598.537)	13.105	18.880	112.938	1.463

Table 4.4. Quantitative Results for Individual 4

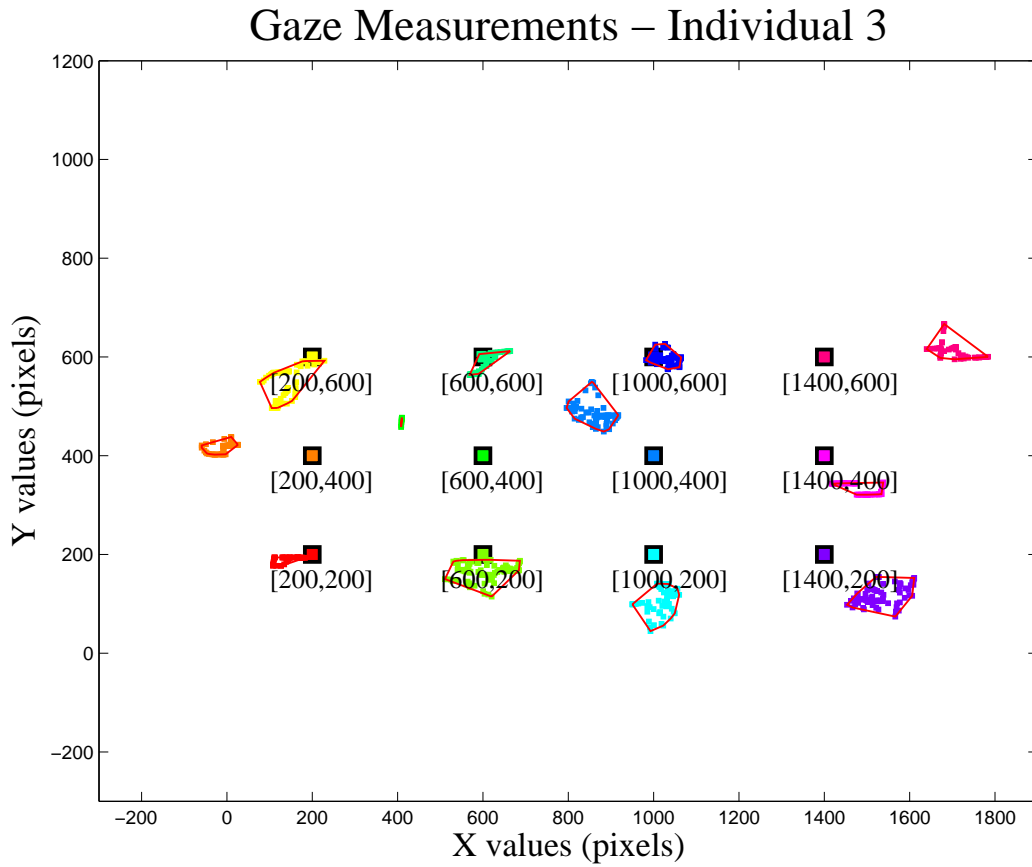


Figure 4.8. Gaze results of Individual 3

Id	Point	Mean	SD Row	SD Col	Err Row	Err Col
1	(200,200)	(16.826,89.022)	10.885	1.360	183.174	110.978
2	(200,400)	(27.627,582.831)	30.057	22.094	172.373	182.831
3	(200,600)	(135.945,1019.860)	30.503	26.658	64.055	419.860
4	(600,200)	(410.257,129.236)	28.026	31.917	189.743	70.764
5	(600,400)	(578.579,648.087)	44.898	33.452	21.421	248.087
6	(600,600)	(653.728,873.690)	50.316	10.974	53.728	273.690
7	(1000,200)	(813.115,131.465)	19.835	28.711	186.885	68.535
8	(1000,400)	(1050.247,652.203)	34.890	38.219	50.247	252.203
9	(1000,600)	(1032.386,852.798)	50.434	38.659	32.386	252.798
10	(1400,200)	(1372.943,233.429)	65.999	39.433	27.057	33.429
11	(1400,400)	(1573.102,659.920)	71.647	45.054	173.102	259.920
12	(1400,600)	(1629.571,904.635)	36.556	44.835	229.571	304.635

Table 4.5. Quantitative Results for Individual 5

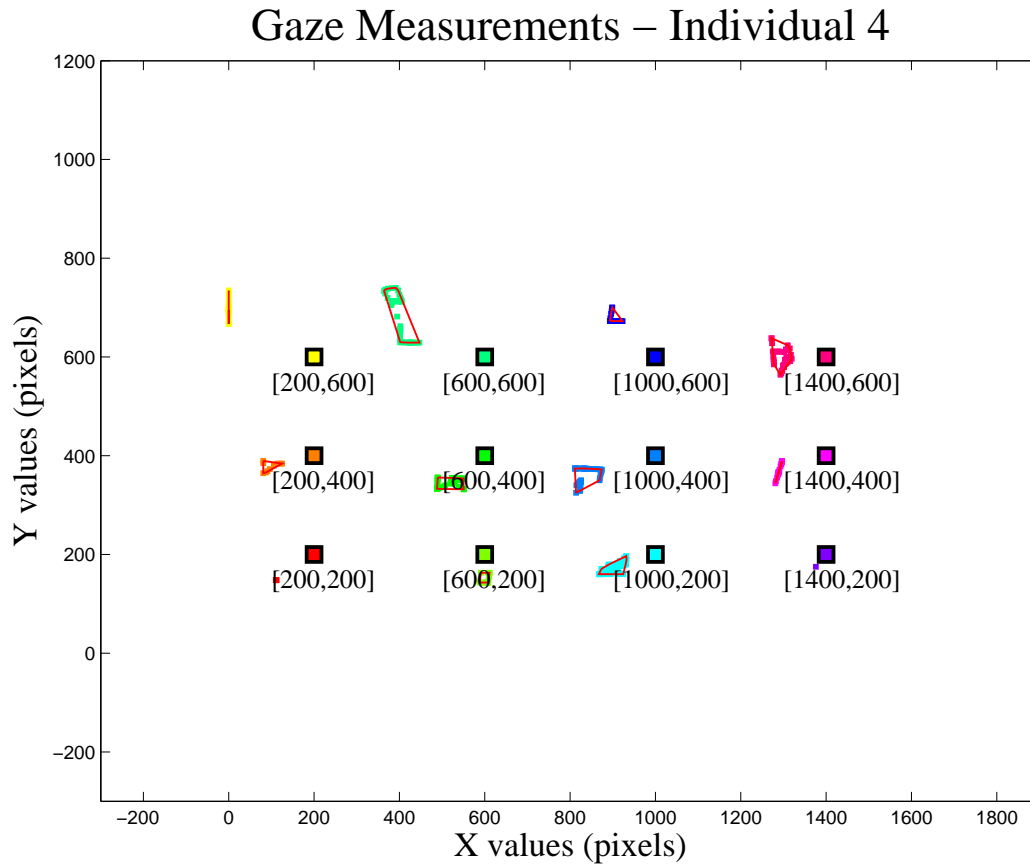


Figure 4.9. Gaze results of Individual 4

Id	Point	Mean	SD Row	SD Col	Err Row	Err Col
1	(200,200)	(138.022,503.411)	17.224	13.138	61.978	303.411
2	(200,400)	(181.784,454.205)	19.318	41.067	18.216	54.205
3	(200,600)	(251.834,670.298)	40.114	62.801	51.834	70.298
4	(600,200)	(395.697,457.423)	97.269	22.257	204.303	257.423
5	(600,400)	(597.841,413.933)	24.290	28.840	2.159	13.933
6	(600,600)	(551.275,584.272)	32.964	15.885	48.725	15.728
7	(1000,200)	(801.184,272.191)	51.068	20.063	198.816	72.191
8	(1000,400)	(973.739,427.979)	32.502	31.729	26.261	27.979
9	(1000,600)	(973.356,530.904)	57.845	28.093	26.644	69.096
10	(1400,200)	(1538.885,93.665)	61.040	25.307	138.885	106.335
11	(1400,400)	(1565.189,428.568)	114.562	46.047	165.189	28.568
12	(1400,600)	(1490.541,631.526)	87.952	50.073	90.541	31.526

Table 4.6. Quantitative Results for Individual 6

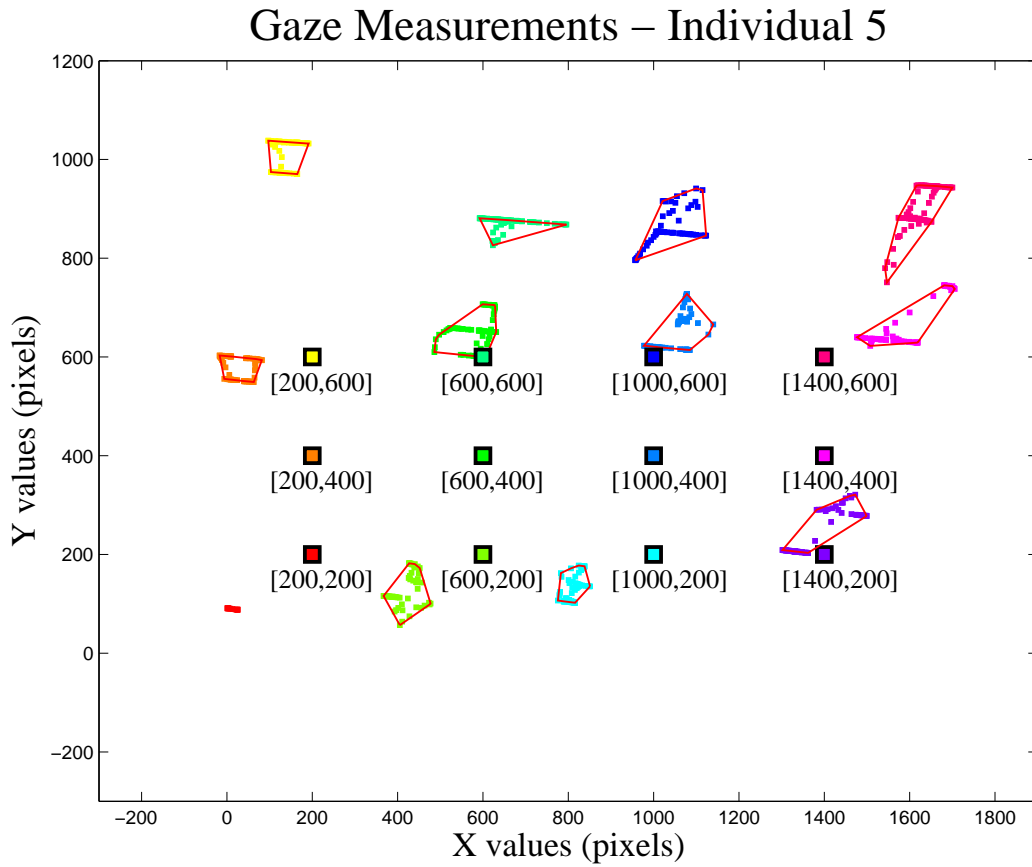


Figure 4.10. Gaze results of Individual 5

Id	Point	Mean	SD Row	SD Col	Err Row	Err Col
1	(200,200)	(160.632,-504.749)	1418.648	921.570	39.368	704.749
2	(200,400)	(-864.896,308.313)	439.424	139.571	1064.896	91.687
3	(200,600)	(-420.925,619.291)	87.518	77.279	620.925	19.291
4	(600,200)	(573.726,-57.811)	122.444	53.830	26.274	257.811
5	(600,400)	(497.753,400.380)	79.186	57.727	102.247	0.380
6	(600,600)	(556.243,607.184)	72.797	45.754	43.757	7.184
7	(1000,200)	(1230.198,191.090)	40.206	26.339	230.198	8.910
8	(1000,400)	(1116.814,398.094)	35.781	32.805	116.814	1.906
9	(1000,600)	(1319.060,851.367)	9.325	33.174	319.060	251.367
10	(1400,200)	(1538.532,287.852)	10.449	31.342	138.532	87.852
11	(1400,400)	(1532.482,497.620)	6.929	15.730	132.482	97.620
12	(1400,600)	(1570.934,768.745)	4.528	10.951	170.934	168.745

Table 4.7. Quantitative Results for Individual 7

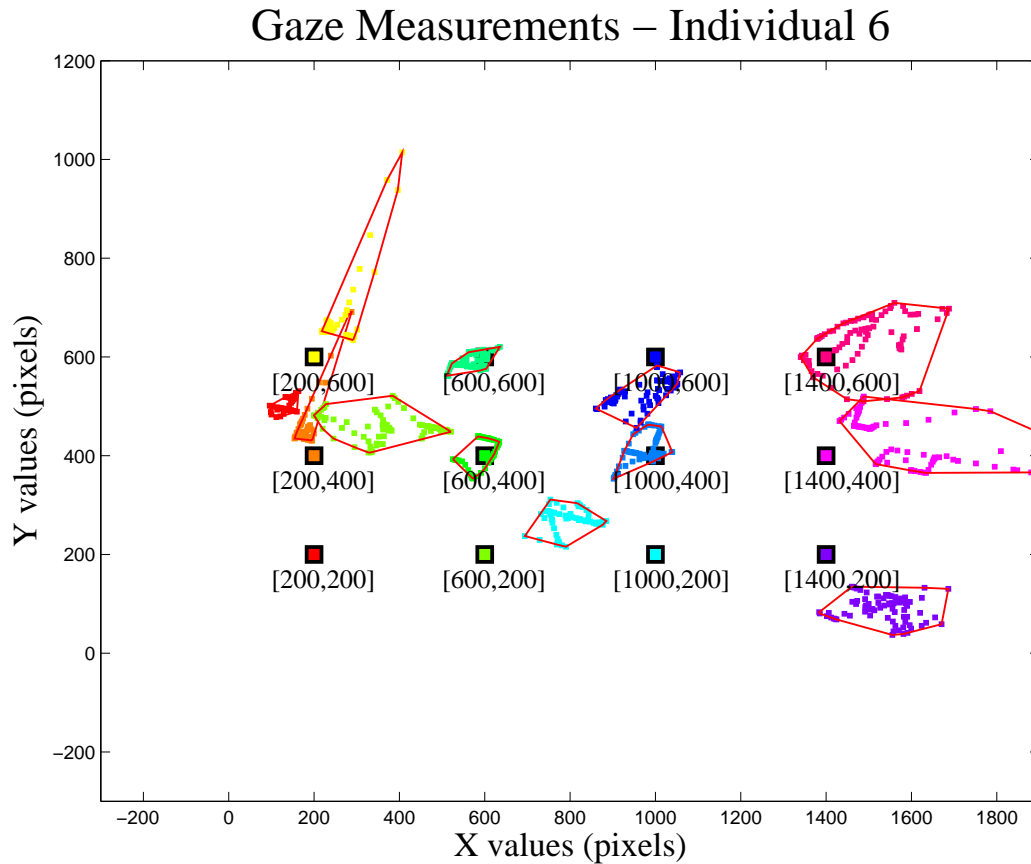


Figure 4.11. Gaze results of Individual 6

correspond to both rows and columns of the image in pixels. The calibration chart shows, for instance, the difference between the highest and lowest values in y axis, the accuracy and precision of the calibration. We have computed those values and they are presented in Tables 4.1 to 4.7. The first column (*Id*) shows the index of the point (in left-right, top-down). The second column (*Point*) shows the real location of the point in the screen. The third column (*Mean*) shows the estimated point location. Columns *SD Row* and *SD Col* show the precision of the estimation and columns *Err Row* and *Err Col* describe how accurate the estimation was.

The iris of Individual 1 is mainly visible. The uncertainty associated with the iris detection during the calibration procedure was really low, which indicates that the gaze estimation might present good results as well. Indeed, the iris tracking was really reliable for that user as shown in Figure 4.6. It contains the plot of 12 points in the screen (colored square with black border) and the estimation locations for the Individual 1. As can be seen, the dispersion of the estimated gaze was low showing

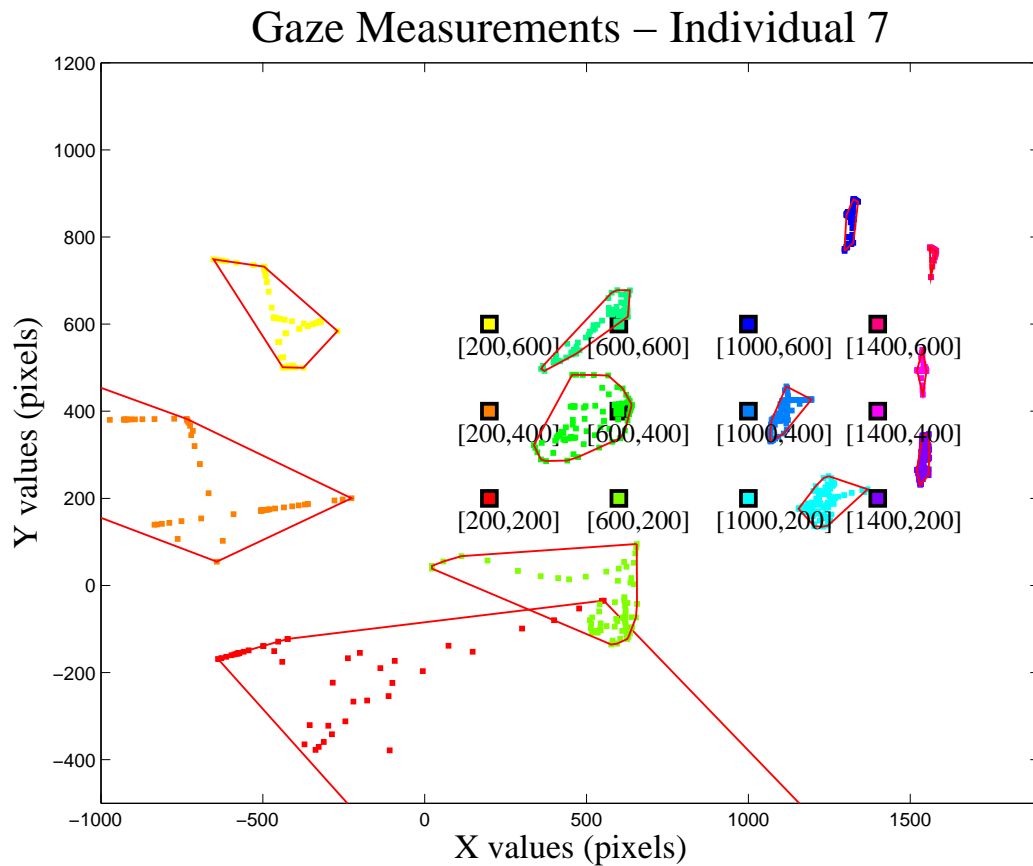


Figure 4.12. Gaze results of Individual 7

good precision and the distance between the the estimation and real location shows good accuracy.

Before discussing about Individual 2, we need to observe the calibration results in Figure 4.5. The dispersion of the calibration for this user was also really small. It is also possible to see that this user was not as close to the camera as Individual 1. For Individual 1, the max difference of highest and lowest location in axis y is about 8 pixels. However, for Individual 2, it is only around 4 pixels. Results for individual 2 indicate that the hard cases for estimation are in the borders as shown in Figure 4.7.

The calibration procedure Individual 3 also shows good precision, which means that when the user looks in a certain direction, the estimation of the eye tend to produce small deviation even though it might not be accurate. Figure 4.8 shows good precision in the gaze estimation, but not so good accuracy. Individual 4 has really good precision in almost all mapped locations as displayed by the cali-

bration chart. However, there was an outlier in the corner, which indicates that for this location, the estimation might produced wrong results.

Individuals 5, 6, and 7 have really poor calibration results. Individual 5 does not have a good precision and results displayed in Figure 4.10 show that the poor estimation for this individual is more related to the y-axis. Calibration for individual 6 also lacks precision, meaning that for each point, the tracking produced many different values. This high dispersion around the mean produced bad estimation as shown in Figure 4.11.

One behaviour that can be noticed for all individuals is that central points have less uncertainty. This gives us a clue that it is easier to track the iris when it is standing towards the center of the screen, but when individuals look towards the corners, the uncertainty may increase.

4.4 Limitations of Our Work

As many works, our approach contain limitations that will be investigated. There are some small issues that are discussed in the conclusions, for instance, one limitation of our approach is to perform gaze estimation along with head movements. We, now, state those limitations from the most serious issues to the most simplistic ones.

Although we have successfully obtained the head pose, the gaze estimation is only reliable when the head is static. It is not straight forward to merge both information. For instance, a simple translation of the user in the X axis would invalidate the mapping obtained by the homography. Currently, the user can move his head freely and the pose and iris location can be estimated, but as soon as it calibrates for gaze estimation, the user cannot move his head, otherwise the gaze will not be established. We intend to solve this problem in the future considering that we have both the rotation matrix and the translation vector of the head.

The current approach works well for eyes that show a good iris exposition. For many eyes, this is not true, as shown in Figure 4.13. As can be seen, the major part of the iris in the left eye is visible. On the other hand, the right eye has the iris partly blocked by eyelids. Although the system might eventually find the iris, it is very likely that it will fail to accurately track it. The accuracy for this latter eye is really low compared with the one in the left image. Hence, a limitation of our approach is the low accuracy in the eye location when the iris is partially occluded by eyelids. A straight-forward solution for users with this characteristic is the use of infrared

light for detecting the pupil instead of iris.



Figure 4.13. The left eye demonstrates good iris exposure, (i.e. a high percentage of iris is visible), on the other hand, the right eye has the iris blocked by the eyelids

The last problem is the robustness related to the head tracking. If the user is being tracked by the system and the face region is blocked by the hand, or a moving object crosses the background in such a way that it would disturb the features, the pose estimation fails. Hence, our system needs to obtain the feature movement consensus in order to correct wrong tracked values for increasing stability.

Chapter 5

Conclusions and Future Work

The advance of technology changes the way people interact with computers. *Computer Vision* has brought many unusual and challenging opportunities to interact with human beings. For instance, gaze tracking is the problem of estimating where a person is looking. To be more specific, the problem discussed over this thesis is *the remote estimation of human gaze using a single camera in visible lighting*. This problem can be addressed in very different ways. Our solution was to split this problem into two others: (i) *the head pose estimation* and (ii) *eye segmentation*.

A single camera system consists in a scenario where there is no explicit three dimensional information when observing the image acquired, i.e. we can not know, without prior information, if the distance between a certain object to the camera is larger than the distance from another object to the same camera when this camera is the only one responsible for acquiring images. This is our scenario, although, we found a way to estimate the missing information: By using ASM and the three-dimensional face model, we are able to notice, for instance, that the nose is closer to camera when compared to the eye. Hence, the first challenge adressed in this thesis was to overcome the lack of depth information in the image by solving the *Head Pose Estimation Problem*.

Our experiments showed that the algorithm implemented in this thesis is powerful and reliable when user performs smooth movements. However, when fast movements occur, or the face is blocked by the user's hand, or even when the user turns his head completely in such a way that most features are not visible, the pose can not be estimated and tracking is lost. Our tracking system, thus, lacks robustness.

A possible solution for the lack of robustness could be implemented by obtaining the *movement consensus*, i.e. when one feature is lost during tracking, we could

infer where it should be supporting our assumption by using the consensus of the others. Afterwards, we could lock this wrong feature into its correct location.

If this proposed consensus method can be successfully built, then, partial occlusion of the face would not disturb the tracking activity, providing, in this way, a powerful tracking. However, it would not be able to solve the *complete facial occlusion problem* when all features are lost and therefore there is no consensus. Hence, an automatic restabilization procedure could be developed to restore the face tracking when all features are lost. This might happen when the user changes his location in such a way that he is not visible by the camera anymore, e.g. user leaves the room. This could also happen when an abrupt illumination change occurs, e.g. turning the lights off.

A possible direction for solving this latter problem could consist on pairing up both images and poses to create an online learning approach that would feed an algorithm with both face poses and images. The correspondence would be created using the output of the estimator. Therefore, when the system loses completely its tracking, the learning technique could restore the system to the best match between the detected face (with wrong or no pose) to its dictionary of own correct poses. The challenge would now be to perform this learning task in a non-supervised way, otherwise this solution would be cumbersome to most users.

Considering that our estimation has identified the pose correctly, we move now to the second phase of our problem: *eye segmentation*. Segmentation means that we are trying to decompose the image into its different parts. Our case consists in information acquired on visible lighting only. So, we face many problems such as glints and specular reflections generated on the user's eye as shown in Figure 1.1. However, the biggest problem is that we cannot separate the user's pupil from his iris when the subject eye is dark, the boundaries between pupil and iris are not easily distinguishable, even for human beings. This is the first big difference when working in NIR spectrum and visible lighting. The iris-pupil boundary is easily found in NIR images. Thus, our information relies solely on the limbus, i.e. the boundary between iris and sclera which shows great contrast for most eyes.

The eye segmentation algorithm proposed in this work assumes that the eye is centered in the image (ROI) and for achieving that, we use information of the head pose estimated before. Thus, we are able to see that the pose estimation has an important impact in the rest of the system.

There are many different ways of segmenting an eye. For some studies, it might be relevant to identify not only the iris, but also pupil, eye corners, eyelids. For our purposes, we need only the successful identification of the iris. We first have

presented a way of reducing the impact of specular reflections on the eye surface by decreasing the brightness of the region which is more likely to contain the iris. Hence, a mask was proposed for enhancing the iris region and eliminate undesired content, such as skin pixels.

Experiments showed that our iris compensation have presented a good impact to reduce the noise generated by glints. Without compensation, the iris might be divided into several parts due to the high intensity of reflections. However, in a poor illuminated scene, sometimes the segmentation process may mingle skin pixels with iris pixels leading to a failure in the iris estimation. This specially happens in a poor illuminated scene or when the individual has a small sclera exposure.

Considering that we proposed a solution for iris segmentation problems, it is important to clarify our decision about tracking the iris instead of detecting it again. Particularly, our decision for using template matching was based on the strong relationship between the iris pixels. Our algorithm relies on the assumption that irises from different individuals vary strongly. On the other hand, the iris of one individual does not vary too much when compared on different frames in the same illumination condition. Hence, template matching acts like a snapshot of the iris identified whose tracking corresponds to a very lightweight solution if compared to the initial iris detection algorithm.

For the gaze estimation problem, our decision consisted on applying homography to map the points in the image to gaze locations in the screen. This technique provides a simple solution for finding the relationship between image and gaze plane. Results indicate that the quality of the estimation using homography is related to how reliable and accurate was the iris detection and calibration procedure. One drawback of our approach is once the head starts moving, the relationship between plane and eye changes. We need to integrate the head pose obtained previously with the angles in order to turn this solution into a head invariant gaze tracker. So far, we have solved those problems separately and applied it into different applications, such as tele-immersion with remote control of robotic head and content analysis with the heatmap.

Bibliography

- BioID (2001). The bioid face database available at <https://support.bioid.com/downloads/>. Last accessed february 2012.
- Bolme, D., Draper, B., and Beveridge, J. (2009). Average of synthetic exact filters. In *Proceedings of Computer Vision and Pattern Recognition (CVPR)*, pages 2105–2112.
- Cootes, T. F., Taylor, C. J., Cooper, D. H., and Graham, J. (1995). Active shape models – their training and application. *Computer Vision and Image Understanding*, 61(1):38–59.
- Deja, S. (2010). System rozpoznawania i aktywnego śledzenia oczu użytkownika komputera za pośrednictwem kamery w czasie rzeczywistym. Master’s thesis, Akademia Wydział Elektrotechniki, Automatyki, Informatyki i Elektroniki, Poland.
- Dementhon, D. F. and Davis, L. S. (1995). Model-based object pose in 25 lines of code. *International Journal of Computer Vision*, 15:123–141.
- Duchowski, A. T. (2007). *Eye Tracking Methodology: Theory and Practice*. Springer-Verlag New York, Inc., Secaucus, NJ, USA.
- Fernandes, C., Alves Neto, A., and Campos, M. F. M. (2011). Um sistema de rastreamento de pose para aplicações em teleimersão. In *X Simpósio Brasileiro de Automação Inteligente (SBAI’11)*, SBAI’11, São João del-Rey, Brazil.
- Fischler, M. A. and Bolles, R. C. (1981). Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM*, 24(6):381–395.
- Gonzalez, R. C. and Woods, R. E. (2007). *Digital Image Processing (3rd Edition)*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA.

- Guestrin, E. D. and Eizenman, M. (2006). General theory of remote gaze estimation using the pupil center and corneal reflections. *IEEE Transactions on Biomedical Engineering*, 53(6):1124–33.
- Hansen, D. W. and Ji, Q. (2010). In the eye of the beholder: A survey of models for eyes and gaze. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 99(1).
- Jefferson, Y. (2004). Facial beauty - establishing a universal standard. *International Journal of Orthodontics - IJO*, 15(1):9–22.
- Ji, Q. and Yang, X. (2002). Real-time eye, gaze, and face pose tracking for monitoring driver vigilance. *Real-Time Imaging*, 8:357–377.
- Li, P., Liu, X., Xiao, L., and Song, Q. (2010). Robust and accurate iris segmentation in very noisy iris images. *Image Vision Computing*, 28:246–253.
- Lieberman, Z., Powderly, J., Roth, E., Sugrue, C., Tempt1, and Watson, T. (2011). Eyewriter, <http://www.eyewriter.org>.
- Lucas, B. D. and Kanade, T. (1981). An iterative image registration technique with an application to stereo vision. In *IJCAI'81: Proceedings of the 7th international joint conference on Artificial intelligence*, pages 674–679, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.
- Martinez, A. and Benavente, R. (1998). The AR face database. Technical report, CVC. Technical Report #24. <http://www2.ece.ohio-state.edu/~aleix/ARdatabase.html>.
- Martins, P. and Batista, J. (2008). Monocular head pose estimation. In *Proceedings of the 5th international conference on Image Analysis and Recognition, ICIAR '08*, pages 357–368, Berlin, Heidelberg. Springer-Verlag.
- Messer, K., Matas, J., Kittler, J., Luettin, J., and Verlag, G. M. X. (1999). M2VTS: The extended M2VTS database. In *Proceedings 2nd Conference on Audio and Video-base Biometric Personal Verification, AVBPA '99*. Springer.
- Milborrow, S. (2007). Locating facial features with active shape models. Master's thesis, University of Cape Town.
- Milborrow, S. and Nicolls, F. (2008). Locating facial features with an extended active shape model. *European Conference on Computer Vision*. <http://www.milbo.users.sonic.net/stasm>.

- Morimoto, C., Koons, D., Amir, A., and Flickner, M. (2000). Pupil detection and tracking using multiple light sources. *Image and Vision Computing*, 18(4):331 – 335.
- Mulvey, F., Villanueva, A., Sliney, D., Lange, R., Cotmore, S., and Donegan, M. (2008). Exploration of safety issues in eyetracking. d5.4. Technical report, Communication by Gaze Interaction (COGAIN). IST-2003-511598.
- Murphy-Chutorian, E. and Trivedi, M. M. (2009). Head pose estimation in computer vision: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(4):607–626.
- Nguyen, V. H., Nguyen, T. H. B., and Kim, H. (2010). Eye feature extraction using k-means clustering for low illumination and iris color variety. In *International Conference on Control, Automation, Robotics and Vision, ICARCV '10*, pages 633–637. IEEE.
- Ni, J. and Chellappa, R. (2010). Evaluation of state-of-the-art algorithms for remote face recognition. In *International Conference on Image Processing, ICIP '10*, pages 1581–1584. IEEE.
- Parker, J. R. and Duong, A. (2009). Gaze tracking - a sclera recognition approach. In *IEEE International Conference on Systems, Man and Cybernetics, San Antonio, TX, USA, 11-14 October 2009*, pages 3836–3841.
- Phillips, J. P., Moon, H., Rizvi, S. A., and Rauss, P. J. (2000). The FERET Evaluation Methodology for Face-Recognition Algorithms. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(10):1090–1104.
- Proença, H. (2011). Quality assessment of degraded iris images acquired in the visible wavelength. *IEEE Transactions on Information Forensics and Security*, 6(1):82–95.
- Rowley, H. A., Baluja, S., and Kanade, T. (1998). Neural network-based face detection. *IEEE Transactions On Pattern Analysis and Machine intelligence*, 20:23–38.
- Tan, T., He, Z., and Sun, Z. (2010). Efficient and robust segmentation of noisy iris images for non-cooperative iris recognition. *Image Vision Computing*, 28:223–230.
- Tasdizen, T., Tarel, J.-P., and Cooper, D. (1999). Algebraic curves that work better. In *Computer Vision and Pattern Recognition, 1999. IEEE Computer Society Conference on.*, volume 2, pages 2 vol. (xxiii+637+663).

- Trucco, E. and Verri, A. (1998). *Introductory Techniques for 3-D Computer Vision*. Prentice Hall PTR, Upper Saddle River, NJ, USA.
- Valenti, R. and Gevers, T. (2008). Accurate eye center location and tracking using isophote curvature. In *Proceedings of Computer Vision and Pattern Recognition (CVPR)*.
- Viola, P. and Jones, M. J. (2004). Robust real-time face detection. *International Journal of Computer Vision*, 57(2):137–154–154.
- Wang, J. Z., Li, J., and Wiederhold, G. (2000). Simplicity: Semantics-sensitive integrated matching for picture libraries. In *Proceedings of the 4th International Conference on Advances in Visual Information Systems, VISUAL '00*, pages 360–371, London, UK, UK. Springer-Verlag.

Appendix A

Mathematical Morphology

Mathematical Morphology (MM) can be used to extract components of the image that are meaningful to represent and describe a shape [Gonzalez and Woods, 2007]. So, we apply the hole filling algorithm followed by morphological operators for removing the outside ridges. The key idea is to approximate the original form of the iris.

In this work, we apply *Erosion* and *Dilation* to eliminate holes and ridges. However, before defining those operations, we first present the concept of *Structuring Element* and finally we introduce the basis and demonstrate the usage of the MM operations in the iris detection problem.

Figure A.1 shows results of applying morphological operations on binary images. As one might notice, the holes were filled and “bad” pixels were removed by the application of morphological operations (erosion and dilation) using a circular structuring element.

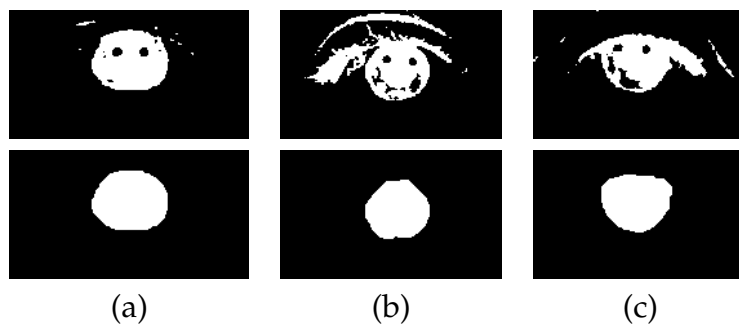


Figure A.1. Morphological Operations: the first row shows some binary images and the second row shows the same images after hole filling and MM operations.

A.1 Structuring Element

A *Structuring Element* is a shape used in MM whose goal is to interact with an image. For instance, in this work, we choose a circular shape (disk) structuring element to *dilate* and to *erode* the image that is displayed in Figure A.2.

Gonzalez and Woods [2007] introduce MM as operations in set theory. For instance, given an set I (binary image) and sliding the set E (structuring element) over it, one could take into account only those pixels that lie on the intersection of both sets.

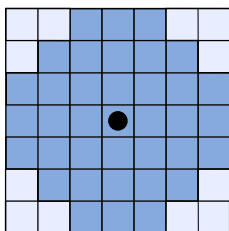


Figure A.2. Structuring Element: a 7×7 convolution idea representing a circular shape.

We move now onto a formal definition of those two morphological operations applied in this work: *erosion* and *dilation*.

A.2 Erosion

Given a binary image I and a structuring element E , Gonzalez and Woods [2007] defined the erosion ($I \ominus E$) of the I by the element E is:

$$I \ominus E = \{z | (E)_z \subseteq I\}. \quad (\text{A.1})$$

In other words, when sliding the structuring element E over the image I , we retain those pixels of I only if E is completely contained by I . Figure A.3 demonstrates the application of five erosion operations of a square using the structuring element already mentioned in Figure A.1. After applying an erosion, the original image is shrunked.

In this case, the result is a smaller square, however, the shape of the image after erosion depends mainly on the shape of structuring element (including its size) and the number of times that this operation has been applied.

The impact of the erosion in an image is really significant. Elements of the image that are smaller than the structuring element are removed (filtered). In this

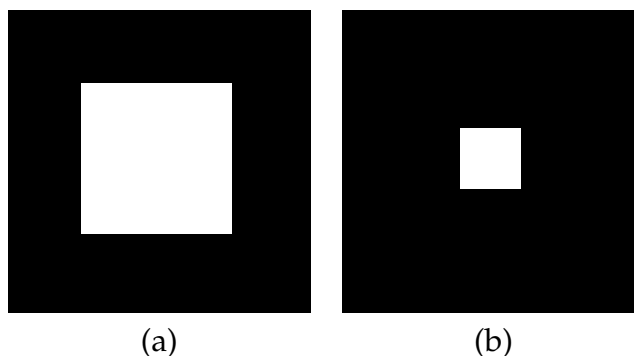


Figure A.3. Erosion Operation: (a) binary image and (b) image eroded five times by the structuring element of Figure A.2

way, it is possible to reduce and hopefully eliminate ridges and isolated pixels.

A.3 Dilation

Differently from erosion, dilation focuses on expanding the image using a structuring element. Again, Gonzalez and Woods [2007] define dilation as:

$$I \oplus S = \{z | [(S)_z \cup I] \subseteq I\} \quad (\text{A.2})$$

If I and S overlap by at least one element, we retain those pixels. Figure A.4 demonstrates the results of applying dilation on the same square presented before. Due to the circular nature of the structuring element, the result is now a rounded squared.

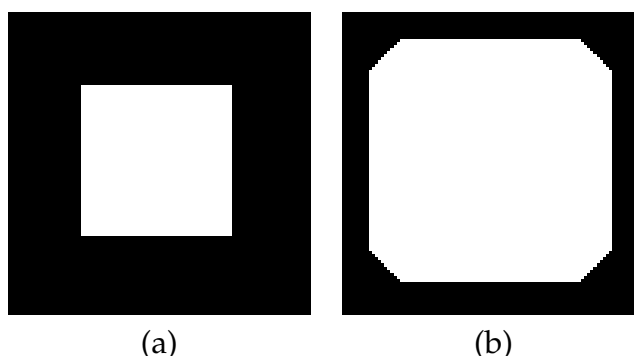


Figure A.4. Dilation Operation: (a) binary image and (b) image dilated five times by the structuring element A.1

According to Gonzalez and Woods [2007], erosion and dilation have a dual relationship related to the set operations (complementation and reflection). The fol-

lowing equation (duality) shows that the erosion of the image S by I can be obtained by dilating the image background (if the structuring element is the same).

$$(I \ominus S)^c = I^c \oplus \hat{S}. \quad (\text{A.3})$$

Considering we explain the important behavior of mathematical morphology, one might wonder why it is necessary to fill holes once morphological operations may fill them. *Dilation* operation might be able to fill holes but it also might connect outliers to the iris. Hence, we first apply *erosion* for erasing most outliers, but the decision of applying erosion may cause a problem even bigger: The iris might be splitted turning into two or more parts. Hence, filling the holes inside iris before eroding showed good results in order to restore the iris shape.

Appendix B

Quadrics and Conics

This chapter provides the background for understanding quadrics and conics.

B.1 Algebraic Curves and Surfaces

An *algebraic curve* is represented as the zero value of a polynomial in two variables [Tasdizen et al., 1999]. The polynomial P with degree n is a function $P_n : \mathbb{R}^2 \rightarrow \mathbb{R}$ given by the following expression:

$$P_n(x, y) = \sum_{0 \leq i+j \leq n} a_{i,j} x^i y^j = 0. \quad (\text{B.1})$$

An *algebraic surface* has the same properties and definition as well. However, the polynomial now has three variables taking into account the third dimension:

$$P_n(x, y, z) = \sum_{0 \leq i+j+k \leq n} a_{i,j,k} x^i y^j z^k = 0. \quad (\text{B.2})$$

Let S be the coefficients of the polynomial, and Q its monomials, it is convenient to use the following notation for the polynomial:

$$P_n(x, y, z) = Q^t \cdot S. \quad (\text{B.3})$$

For instance, the following polynomial

$$P_2(x, y, z) = 3x^2 + 2y^2 - 4z^2 - xy + 2yz + 4x - 5y + 8,$$

can be described as the dot product of Q^t and S :

$$Q^t = [x^2, y^2, z^2, xy, xz, yz, x, y, z, 1], \text{ and}$$

$$S^t = [3, 2, -4, -1, 0, 2, 4, -5, 0, 8].$$

B.2 Quadrics and Conics

A quadric surface is a special case of an algebraic surface in which

$$P_2(x, y, z) = Ax^2 + By^2 + Cz^2 + Dxy + Exz + Fyz + Gx + Hy + Iz + J, \quad (\text{B.4})$$

and it can be rewritten as $P = Q^t \cdot S$:

- $Q^t = [A, B, C, D, E, F, G, H, I, J]$, and
- $S^t = [x^2, y^2, z^2, xy, xz, yz, x, y, z, 1]$.

By using quadrics, many surfaces can take advantage of implicit formulation $L = \{(x, y, z) \in \mathbb{R}^3; P(x, y, z) = Q^t \cdot S = 0\}$. Table B.2 describes some of those quadric surfaces and their proper equations. It also classifies the quadrics as degenerate or non-degenerate.

Table B.1. Quadric surfaces.

Quadric	Equation	Degenerate
Ellipsoid	$\frac{x^2}{a^2} + \frac{y^2}{b^2} + \frac{z^2}{c^2} = 1$	No
Elliptic Paraboloid	$\frac{x^2}{a^2} + \frac{y^2}{b^2} - z = 0$	No
Hyperbolic Paraboloid	$\frac{x^2}{a^2} - \frac{y^2}{b^2} - z = 0$	No
Hyperboloid	$\frac{x^2}{a^2} - \frac{y^2}{b^2} - \frac{z^2}{c^2} = 1$	No
Cone	$\frac{x^2}{a^2} - \frac{y^2}{b^2} - \frac{z^2}{c^2} = 0$	Yes

In Euclidean space, quadrics have dimension $d = 2$. However, in the Euclidean plane they have dimension $d = 1$, also known as *conics*. Let c be a cone, and l a hyperplane. Figure B.2 shows some conics generated by the intersection of l and c .

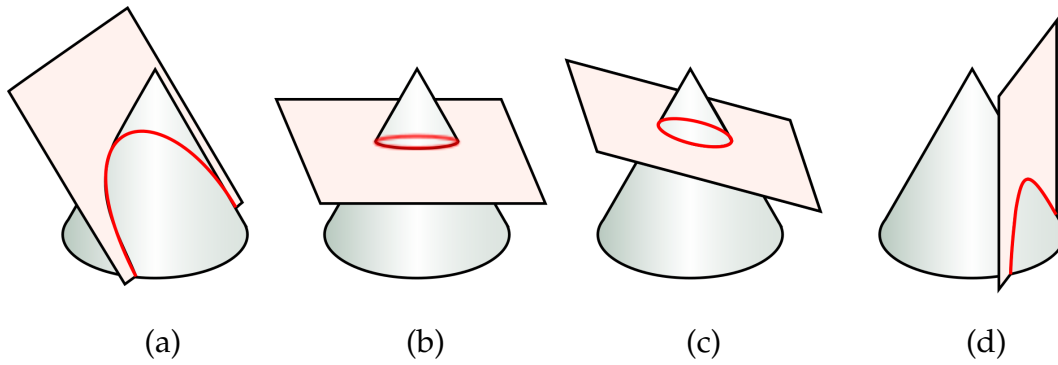


Figure B.1. Conics Sections generated by the intersection of a hyperplane through a cone: (a) parabola, (b) circle, (c) ellipse, and (d) hyperbola.

Table B.2. Conic Sections.

Conic	Equation
Circle	$x^2 + y^2 = r^2$
Ellipse	$\frac{x^2}{a^2} + \frac{y^2}{b^2} = 1$
Hyperbola	$\frac{x^2}{a^2} - \frac{y^2}{b^2} = 1$
Parabola	$y^2 = 4ax$