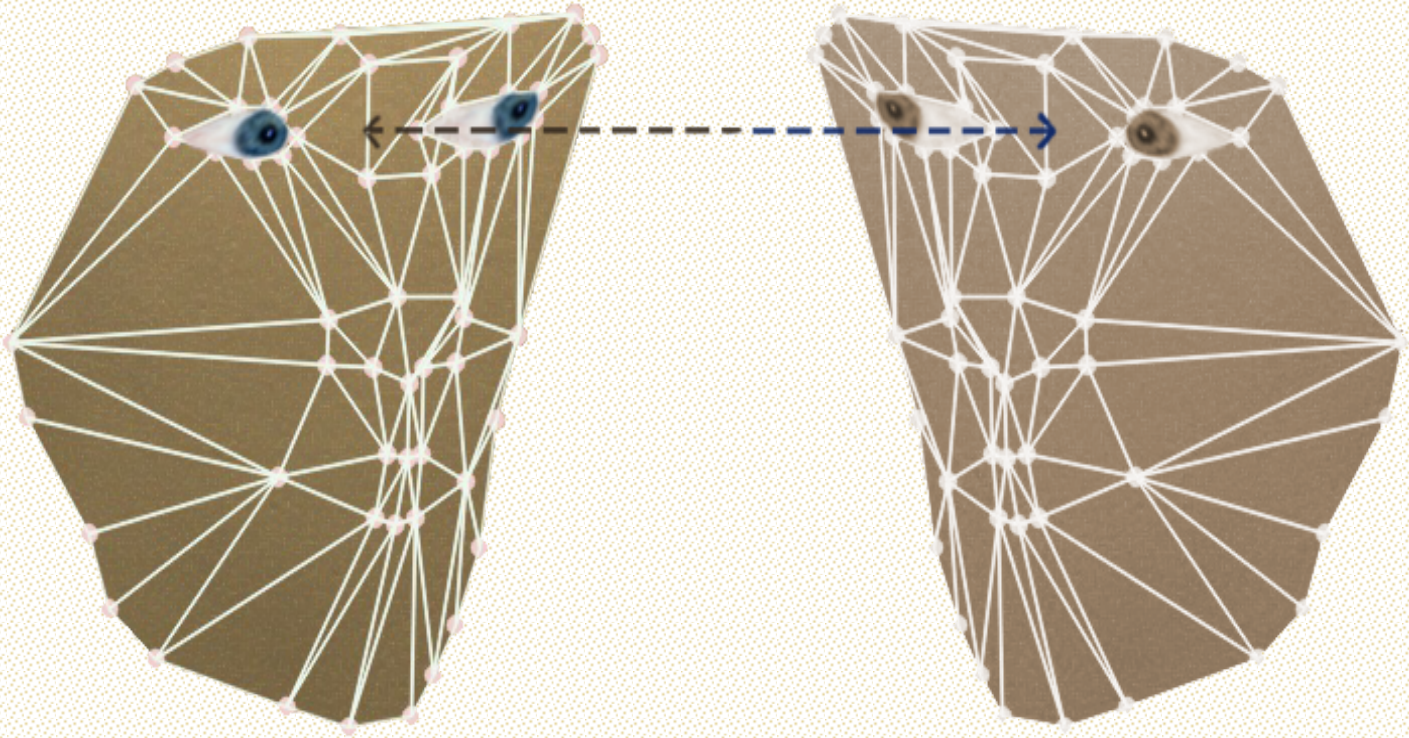


Cristóvão Cordeiro

“Are You Looking at Me?”

An Eye2Eye Human Interaction Evaluation

Thesis submitted in partial fulfillment
of the requirements for the Master's Degree
July 2012

· U C ·

Universidade de Coimbra

FACULTY OF SCIENCES AND TECHNOLOGY
UNIVERSITY OF COIMBRA
Masters in Electrical and Computer Engineering

"Are you looking at me?"

An Eye2Eye Human Interaction Evaluation

by

Cristóvão José Domingues Cordeiro

Thesis submitted in partial fulfillment of the
requirements for the Master's Degree in the
Department of Electrical and Computer Engineering

July 2012

This thesis was done under the supervision of Dr. Jorge Manuel Pereira Batista

Department of Electrical and Computer Engineering, University of Coimbra

Resumo

Os sistemas de "eye tracking" (seguimento do olho) são cada vez mais usados nos dias que correm. Em diversas áreas, tais como a seguranạ, neurologia, ergonomia, etc...o estudo do olho humano através da tecnologia é uma grande vantagem. Os sistemas comerciais existentes requerem um processo inicial de calibração exaustivo, ou então são baseados em métodos intrusivos, usando fontes de iluminação infravermelha. Além disso, a maior parte destes assume que a cabeça do utilizador está fixa, para que assim se possam ignorar os 6 graus de liberdade que o movimento desta acrescenta ao problema. Nesta tese é apresentado um sistema de avaliação de contacto visual para atuar numa conversação entre duas pessoas. Este sistema funciona com base na direção do olhar das pessoas, com liberdade de movimento da cabeça, usando um par de câmaras calibradas. A estimação da pose da cabeça é feita com base em modelos rígidos da cabeça combinados com Modelos de Aparências Ativos. A direção do olhar é estimada a partir de um modelo geométrico 3D do olho. Finalmente, considera-se que existe contacto visual quando ambas as pessoas se olham nos olhos. Os resultados experimentais mostram que com o método proposto é possível obter um sistema de avaliação de contacto visual preciso, direcionado para uma conversação normal entre duas pessoas sem restrições de movimento, e usando apenas duas câmaras calibradas e um computador. Este sistema distingue-se por ser passivo, compacto e por ser capaz de estimar a pose da cabeça e a direção do olhar em simultâneo e em tempo real. Durante os resultados experimentais foi observado que o erro na estimativa da direção do olhar nunca ultrapassou os 5° (tanto na direção horizontal como na vertical). O avaliador the contato visual demonstrou a capacidade de detetar com precisão aproximadamente 85% dos casos em que realmente existe contato visual, numa conversa frontal e com poses naturais.

Palavras-chave: Olhar; Íris; Pose da Cabeça; Kalman; Calibração.

Abstract

Eye tracking systems are currently becoming more and more useful and practical. In several areas, such as security, medicine, ergonomics, etc...the study of the human eye using technological equipment is a big advantage. The existing eye gaze trackers either are intrusive methods (using IR illumination) or require an exhaustive personal calibration. Besides, most of gaze estimators assume a fixed head pose so that they can ignore the 6 degrees of freedom that free head motion adds to the problem. What is presented in this thesis is a visual contact evaluator for conversations between two users. This evaluator functionality is based on the gaze direction of the users, under free head movements, using a pair of calibrated cameras. The head pose estimation is based on rigid head models combined with Active Appearance Models (AAM) for head tracking. The gaze direction is estimated based on a 3D geometric model of the eyeball. Finally, it is considered that there are visual contact when both users are looking at each other's eyes. The experimental results show that with the proposed methodology it is possible to achieve an accurate visual contact evaluator for a regular conversation with two individuals with no movement limitation, using only a pair of calibrated cameras and a computer. This system excel for being non-intrusive, compact and for computing head pose and gaze direction simultaneously and in real time. In the observed experiments, the gaze estimation error was less than 5° (in both X and Y directions). The visual contact evaluator has demonstrated the capability to accurately detect visual contact in approximately 85% of eye contact situations within a regular and natural frontal conversation.

Keywords: Gaze; Iris; Head Pose; Kalman; Calibration.

Acknowledgements

First of all, I would like to thank Doctor Jorge Batista for his supervision and for his confidence in my capabilities to do this thesis.

I would also like to thank my laboratory colleagues and Dr. Joao P. Barreto for all the help they gave me.

To my family, friends and girlfriend, for their support and patience along these past months, thank you.

Contents

List of Figures

List of Tables

Abbreviations

AAM	**A**ctive **A**ppearance **M**odel
ROI	**R**egion **O**f **I**nterest
POSIT	**P**ose from **O**rthography and **S**caling with **IT**erations
DOF	**D**egrees of **F**reedom
PCA	**P**rincipal **C**omponents **A**nalysis
GPA	**G**eneralized **P**rocrustes **A**nalysis
SIC	**S**imultaneous **I**nverse **C**ompositional
HMD	**H**ead-**M**ounted **D**isplay
IR	**I**nfra**R**ed
PCCR	**P**upil **C**enter **C**orneal **R**eflection
RPY	**R**oll, **P**itch and **Y**aw
FOV	**F**ield **O**f **V**iew

Chapter 1

Introduction

The eyes are probably the most important data gatherer of human body. Through them is possible to evaluate a wide range of physical and mental conditions, from attention disorders evaluation, to fatigue detection and even people recognition. Being able to analyse these eye related features aloof (using technological equipment) is a big advantage and contribution to science, since it provides fully automatic systems to complement human evaluation in several situations. Besides human behaviour evaluation, these kind of systems are also becoming more and more useful in daily routines and leisure activities. From hand free control in virtual environments (games, applications, etc.), to computer interfaces for disabled people, and even for multitasking, when the hands are busy doing another job. Eye tracking systems are then very desired nowadays. Their contribution to human behaviour analysis is very important. Along with them there are other tracking and recognition systems that help and improve the eye detection and tracking, more specifically head trackers. However, most of these systems usually require expensive hardware, exhaustive calibrations and are based on intrusive methods, what is a problem from the consumer point of view. In this work will be presented several independent systems that when combined will generate an efficient and user-friendly visual contact evaluation system, that runs on small and cheap equipment.

1.1 Thesis Description and Goals

The main idea behind this project is to detect and track the human head and eyes, and with that information compute the eye gaze in the three dimensional space, so that it can

be intersected with any 3D plane/region of interest.

First of all, since the eye gaze will be computed under free head movement, it is necessary to have a tracking system for the head. This can be achieved by using a statistical based template matching method, called Active Appearance Model (AAM). This fitting algorithm is a combination of a shape model with an appearance model, allowing the detection of faces in real time. The AAM model has a fixed number of feature points, including eyelid points that will be useful to extract the region of interest (ROI) of the eyes. The free head motion adds 6DOF to the problem, what would require an unworkable number of training samples. To avoid that, a solution to estimate monocular head pose must be implemented to compensate the gaze direction angle. Using a 3D rigid model of the head with the Pose from Orthography and Scaling with ITerations (POSIT) algorithm it is possible to correspond the 3D points from a rigid head model with the 2D head features of the AAM and find the 6DOF head pose with respect to the camera. The head pose compensation on the gaze direction is done by considering a 3D geometric model of the eye. The eyes can perform independent movements of the head, and for that reason they must be examined separately from the head. This geometric model of the eye will take into account that its anatomical features slightly change from person to person, which will involve a 3 point offline personal calibration. The last parameter required for the gaze calculation is the pixel position of the iris centre. As the visual axis always crosses the pupil (that has the same centre as the iris), its centre becomes very important for the gaze direction calculation. To find this point accurately, it will be used a fast and robust circular edge detector that is applied in the eye image defined by the eyelid points of the AAM.

After computing the gaze direction of a person, it will be created a practical situation where to use that information. With two individuals in a frontal conversation, approximately 1,5 meters apart from each other, and using a pair of calibrated cameras, where each one can only "see" one individual, a novel method will be presented to evaluate whether there is visual contact or not.

Figure 1.1 illustrates how the final application works, step by step, showing the ordered algorithms that the program has to run before being able to evaluate eye contact.

This thesis will then be structured in three main chapters:

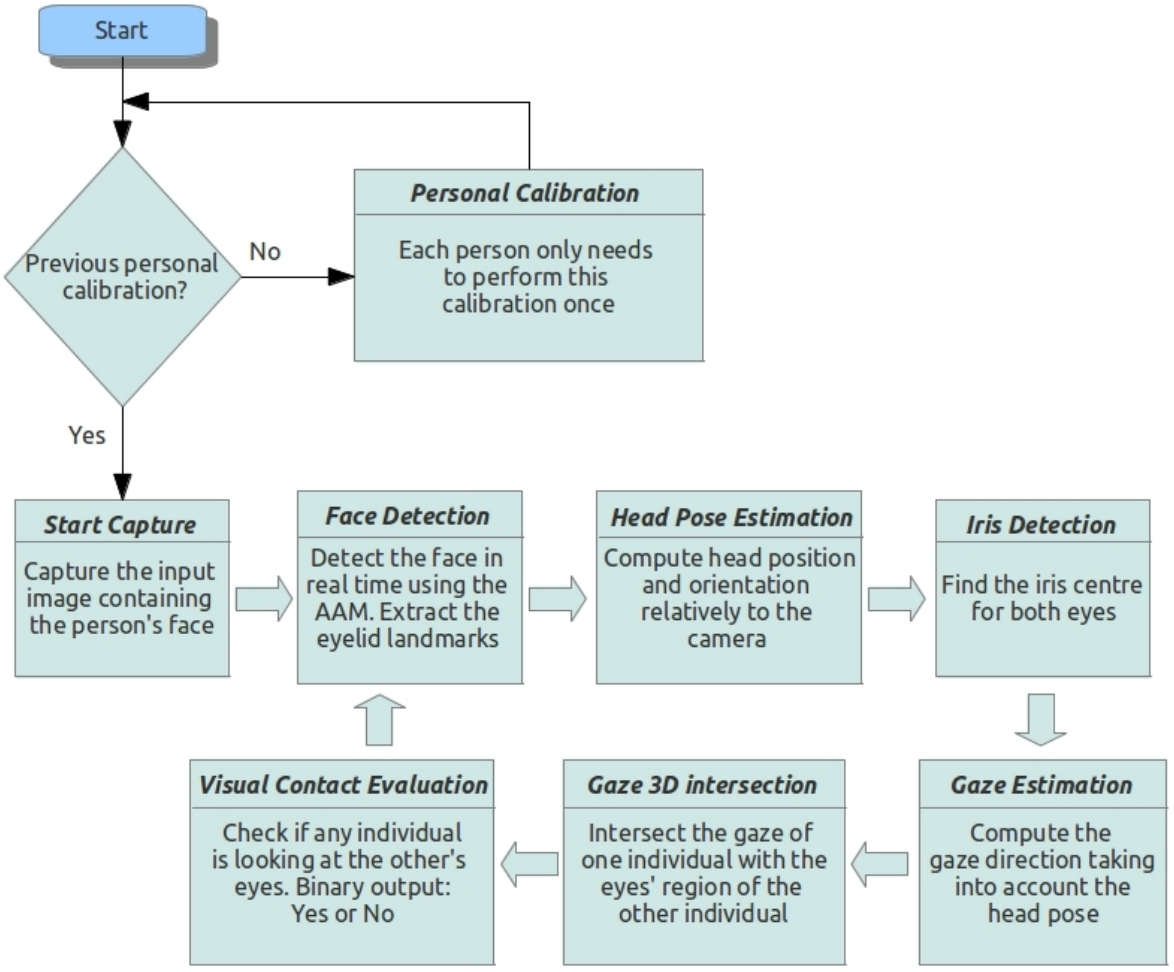


FIGURE 1.1: Scheme of operation of the final application.

- **Head and Eye Tracking**: where will be discussed the methods to track the head and eyes as well the method to estimate the head pose;
- **Gaze Estimation**: where will be presented the geometric model of the eye and the mathematical procedure to compute the gaze direction;
- **Visual Contact Evaluation**: where finally will be discussed the practical application for the gaze estimator, using a pair of calibrated cameras and two individuals.

1.1.1 Work Environment

This work was developed in C++, on UNIX operating system (Ubuntu), and it can be executed from the terminal. It was also used a set of two CCD firewire cameras (Point Grey

FL2-08S2C). To run the final application, it only is required a pair of calibrated firewire cameras (with reasonable quality) and a regular computer running Ubuntu, with OpenCV 2.3.1 installed. The better the specified equipment, the better the performance of the application, obviously.

1.2 Previous and Related Work

Extensive research and work has been made, along these past years, around the head pose and gaze estimation. Most of these works attend to find new and/or better methods to estimate these values. Some of these works have become commercial products, such as magnetic sensors, link mechanisms and red eye tracking systems. The problem with these products is that all of them require expensive hardware or artificial environments, becoming intrusive and uncomfortable to the user. Accurate eye gaze trackers like in [20], with less than 1° of eye gaze estimation error, are based on head-mounted displays (HMD) and IR leds. Despite the accuracy of that method, it is much more profitable to have a slightly bigger error and use only a monocular camera to do the tracking.

Most of the gaze tracking systems (commercial and non-commercial, [21],[22],[23]) are based on Pupil Center Corneal Reflection (PCCR) methods. For example, in [23], the taken approach says that a way to estimate the point of gaze, under free head motion, is to use multiple light sources (and multiple cameras to avoid multiple-point calibration). These PCCR techniques illuminate the eye region with IR lights in order to create a corneal reflection, and at the same time capture those glints with one or multiple cameras. The corneal reflection and the pupil are then enough to infer the visual axis of the eye and respective gaze direction. However, these IR illumination based systems can not work very well in outdoor scenarios. The excessive use of IR lights and cameras imply an exhaustive and sensitive calibration and also, high resolution cameras are needed since the corneal reflections are very small. In addiction to this, the PCCR based tracking systems can be easily affected by unwanted light sources and show signs of reflectivity issues before glasses and contact lenses.

Alternatively, many methods have been proposed to estimate eye gaze directly from the iris or pupil contours using ellipse fitting approaches. Some of those methods either only allow

small head movements [24] or are based in the distance from the iris center to a reference point (e.g. the eye corners, [32]), sometimes requiring a fixed face orientation [25] or limiting the head movements to avoid face features occlusion [26]. There are some appearance based approaches that showed very accurate results on gaze estimation, like [35], [36], [37] and [38]. The problem with these is that they all propose an exhaustive training method (e.g. in [35] a neural network with 2000 labelled training samples is proposed). Besides that, only in [38] free head motion is allowed. Sugano *et al.* [39] proposes an incremental learning method (without offline calibration) for gaze estimation with free head movement, however its estimation accuracy, even after obtaining up to 1000 training samples, isn't very high (approximately 5° of error).

A different approach is taken in [27] where the gaze is classified into five different directions, and each one of them is modelled by a distinct eigenspace. Obviously this approach is bad if the goal is to know the exact point of focus, and also it can fail if the eye images are not well extracted resulting on wrong eigenspace evaluation. It is then known that an accurate visual gaze estimation relies on two factors: the head pose estimation and the eye location. To visually detect the pose of an head, several studies have been reported ([28], [29], [30]). Some of these works estimate eye gaze with free head motion using a stereo vision configuration, like [31]. One recent method [33] consists on computing the visual gaze based on a combination of a cylindrical head model pose tracker and an isophote based eye center locator [34]. This method reported good results, however they assume that the gaze direction does not fall outside of a visual field of view defined by the head pose, eliminating extreme eye positions.

Most modern approaches have started to consider model-based approaches founded on a 3D eye model to compute the gaze direction with free head motion using only monocular video images ([40], [41], [42] and [3]). Each one of them has its own interpretation of the 3D geometric model of the eye but all of them reported good results on the estimated gaze, without requiring IR illumination, exhaustive calibrations and multiple cameras.

The methodology presented in this work is similar to the one proposed in [3]. The same 3D geometric eye model is adopted. However, the face features tracker and head pose estimator used in this work, is the same as the one developed by Pedro Martins in [6].

Chapter 2

Head and Eye Tracking

2.1 Head Tracking

As the main goal is to accurately estimate the gaze under free-head movements, it is absolutely necessary to have a head pose estimation system. Also for the visual contact evaluation, it is necessary to know the ROI of the eyes, on any head pose and in real time. To estimate the head pose, it is assumed that the human head can be compared to a rigid body structure ([6] and [15]). This rigid body structure approximates the human head using a statistical anthropometric 3D rigid model, that combined with a model-based system for facial features extraction and tracking, allows the estimation of the 6DOF head pose. To estimate the head pose the Pose from Orthography and Scaling with ITerations (POSIT) algorithm is used, and for the interpretation of face images an Active Appearance Model (AAM) is used [16].

2.1.1 The AAM

An active appearance model (AAM) [17] is a statistical based template matching method (used, in this case, to model faces) that operates on parametric models of shape and texture, where the variability of the models is captured from a representative training set that is built during an offline phase. More specifically, the AAM addressed here is a combination of two independent AAM's: one AAM to model shape and another AAM to model appearance (or texture).

2.1.1.1 Parametric Model of Shape

The shape of the AAM is defined by a 2D triangulated mesh that is characterized by its vertex locations. Let the shape be represented by s. A single v-point mesh is a $2v$ vector composed by:

$$s = (x_1, \ldots, x_v, y_1, \ldots, y_v)^T \tag{2.1}$$

This AAM offline training data consists of a set of N annotated images with the shape mesh marked by hand. The reference shape mesh used in this work contains 58 vertices (landmarks). Figure 2.1 shows the reference shape used to mark all the trained images. The AAM allows linear shape variation. The standard procedure is to apply a Principal

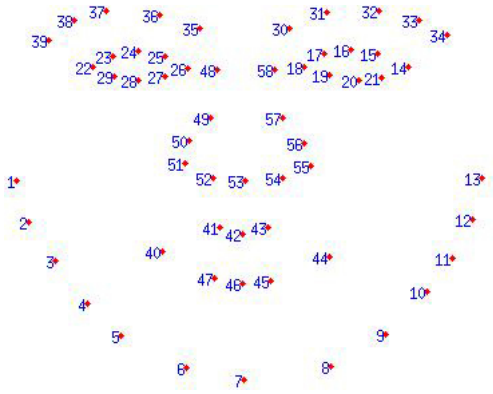


FIGURE 2.1: Reference shape mesh with 58 ordered points.

Component Analysis (PCA) [18] to the trained images. In this case, before applying the PCA, the training meshes are aligned to a common mean shape using a Generalized Procrustes Analysis (GPA), removing location, scale and rotation effects. This way the resulting PCA only has to deal with local and non-rigid shape deformation. The shape s can then be expressed by:

$$s = s_0 + \sum_{i=1}^{n} p_i s_i + \sum_{j=1}^{4} q_j \Psi_j \tag{2.2}$$

Equation (2.2) defines a shape s by deforming a mean shape s_0 using a weighted linear combination of n eigenvectors s_i. The coefficients p_i and q_j are shape and 2D pose parameters, respectively. The parameter Ψ is a 4-column matrix that holds four eigenvectors that linearly model the 2D pose.

2.1.1.2 Parametric Model of Appearance

The texture model is defined within the mean shape s_0. Each training image is warped in order that the control points match those of s_0. The texture mapping procedure is done using a piecewise affine warp that is defined by partitioning the convex hull of s_0 by a set of triangles using the Delaunay triangulation. Then, every pixel inside a specific triangle is mapped into the correspondent triangle in s_0. Defining pixel coordinates that lie inside the mesh s_0 as $\mathbf{x} = (x, y)^T$, the appearance of the AAM is an image $A(\mathbf{x})$, defined over the pixels $\mathbf{x} \in s_0$ such as:

$$A(\mathbf{x}) = A_0(\mathbf{x}) + \sum_{i=1}^{m} \lambda_i A_i(\mathbf{x}) \ , \quad \mathbf{x} \in s_0 \tag{2.3}$$

The coefficients λ_i are appearance parameters. As the AAM allows linear appearance variation, the appearance $A(\mathbf{x})$ can be expressed as a base appearance $A_0(\mathbf{x})$ plus a linear combination of m appearance images $A_i(\mathbf{x})$. As the shape model, the texture model is obtained by applying a low-memory PCA on the normalised textures.

2.1.1.3 AAM Fitting into an Image

Fitting an AAM means that for an image $I(\mathbf{x})$, the goal is to minimize the texture error between the backwarped image onto the base mesh $I(W(\mathbf{x}, p, q))$ and the current model instance $M(W(\mathbf{x}, p, q)) = A(\mathbf{x})$, in a least square sense. The warp W is the piecewise affine warp from s_0 to the current AAM shape s, and that is why W is a function of the shape parameter p and q. At pixel $\mathbf{x}$, the AAM has the appearance shown in equation (2.3). At pixel $W(\mathbf{x}, p, q)$ the input image has the intensity $I(W(\mathbf{x}, p, q))$. So the expression that formulates the fitting problem is given by:

$$\arg\min_{p,q,\lambda} \sum_{\mathbf{x} \in s_0} \left[A_0(\mathbf{x}) + \sum_{i=1}^{m} \lambda_i A_i(\mathbf{x}) - I(W(\mathbf{x}, p, q)) \right]^2 \tag{2.4}$$

There are several algorithms that can be used to minimize equation (2.4), and everyone has its advantages and disadvantages. The Simultaneous Inverse Compositional (SIC) algorithm (see [16] for detailed explanation) minimizes equation (2.4) by performing a Gauss-Newton gradient descent optimization on parameters p, q and λ simultaneously. Compared to some other fitting algorithms, SIC can be rather slow, but on the other hand it achieves a very

good fitting accuracy. Figure 2.2 illustrates the fitting process, showing the input image $I(\mathbf{x})$, the model instance $A(\mathbf{x})$ and the AAM fitting result with SIC algorithm.

(a) (b) (c) (d)

FIGURE 2.2: AAM fitting process.(a) Input face image;(b) Model instance;(c) Warped image I(W(x,p,q));(d) Final result.

2.1.2 Head Pose Estimation

The human head has six degrees of freedom (6DOF), that can be described by two main parameters: the orientation and translation. The POSIT algorithm [19] is a fast and accurate iterative method to find the 6DOF pose of a 3D model with respect to a camera.

Figure 2.3 illustrates the 2D projections, in the image plane of a pinhole camera, of a set of 3D points.

The image plane is at the focal length f of a camera with center of projection C. For now lets assume that the camera is calibrated and therefore these values are known. The points M_i represent the 3D points of a generic model with a coordinate system centred at M_0. The camera frame and the model frame have independent coordinate systems. This means that a 3D point $M_i = [X_i, Y_i, Z_i]^T$ in the model frame, has unknown coordinates in the camera frame. The only thing that the camera frame can "see" is the projections of M_i, m_i, in pixel coordinates (u'_i, v'_i). As the goal is to compute the model pose relatively to the camera, it is considered that the camera frame is the reference coordinate system, and therefore the

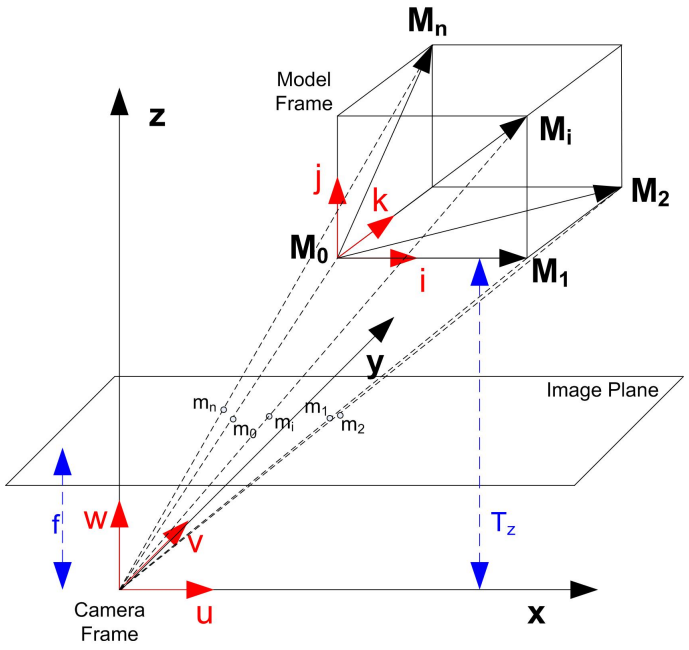


FIGURE 2.3: 2D projections of a generic 3D model. Image courtesy of Pedro Martins, [15].

projection of a 3D point in 2D image coordinates is given by:

$$\begin{bmatrix} u' \\ v' \\ w' \end{bmatrix} = K \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} R & T \\ 0_{1\times 3} & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \tag{2.5}$$

Where R and T are the rotation matrix and translation vector, respectively, expressed in the camera frame relatively to the 3D model coordinate system. The parameter K is the intrinsic parameters matrix that is obtained from the camera calibration (its origin is better explained further on subsection 2.1.3). Considering that normalized image coordinates are used by applying:

$$\begin{bmatrix} u \\ v \\ w \end{bmatrix} = K^{-1} \begin{bmatrix} u' \\ v' \\ w' \end{bmatrix} \tag{2.6}$$

And considering also that:

$$T = \begin{bmatrix} T_x \\ T_y \\ T_z \end{bmatrix} \tag{2.7}$$

and

$$R = \begin{bmatrix} r_1 \\ r_2 \\ r_3 \end{bmatrix} = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} \tag{2.8}$$

equation (2.5) can be written as:

$$\begin{bmatrix} u \\ v \\ w \end{bmatrix} = \begin{bmatrix} r_1/T_z & T_x/T_z \\ r_2/T_z & T_y/T_z \\ r_3/T_z & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \tag{2.9}$$

Solving equation (2.9) in order to $[u \quad v]$:

$$[u \quad v] = \begin{bmatrix} X & Y & Z & 1 \end{bmatrix} \begin{bmatrix} r_1^{\mathbf{T}}/T_z & r_2^{\mathbf{T}}/T_z \\ T_x/T_z & T_y/T_z \end{bmatrix} \tag{2.10}$$

Applying equation (2.10) to n-points:

$$\begin{bmatrix} u_1 & v_1 \\ u_2 & v_2 \\ \vdots & \vdots \\ u_n & v_n \end{bmatrix} = \begin{bmatrix} X_1 & Y_1 & Z_1 & 1 \\ X_2 & Y_2 & Z_2 & 1 \\ \vdots & \vdots & \vdots & \vdots \\ X_n & Y_n & Z_n & 1 \end{bmatrix} \begin{bmatrix} r_1^{\mathbf{T}}/T_z & r_2^{\mathbf{T}}/T_z \\ T_x/T_z & T_y/T_z \end{bmatrix} \tag{2.11}$$

The calculation of the 3D pose is now straightforward:

$$\begin{bmatrix} r_1^{\mathbf{T}}/T_z & r_2^{\mathbf{T}}/T_z \\ T_x/T_z & T_y/T_z \end{bmatrix}_{4\times 2} = M_{4\times n}^{-1} \begin{bmatrix} u_1 & v_1 \\ u_2 & v_2 \\ \vdots & \vdots \\ u_n & v_n \end{bmatrix}_{n\times 2} \tag{2.12}$$

Where:

$$M = \begin{bmatrix} X_1 & Y_1 & Z_1 & 1 \\ X_2 & Y_2 & Z_2 & 1 \\ \vdots & \vdots & \vdots & \vdots \\ X_n & Y_n & Z_n & 1 \end{bmatrix} \tag{2.13}$$

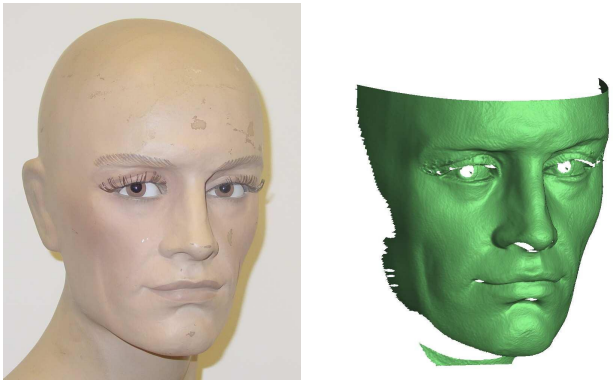
is the model matrix that defines the structure of the 3D model. Notice that r_3 is obtained by setting $r_3 = r_1 \times r_2$, since the rotation matrix rows are orthogonal. This approach determines the object pose for fixed values of w_i. From equation (2.9):

$$w_i = 1 + \frac{r_3}{T_z} \begin{bmatrix} X & Y & Z \end{bmatrix}^{\mathbf{T}} \tag{2.14}$$

The POSIT (POS with ITerations) algorithm works by iterations, re-estimating equation (2.14) until the process converges. The algorithm starts by assuming that $w_i = 1$, $i = 1, \ldots, n$, and using the pose retrieved by equation (2.12), the value of w_i is recomputed in eq. (2.14). When $w_i - w_{i-1} < \Delta$, it is assumed that the estimated pose is accurate enough (Δ is a small error value that is constant and previously defined). With this process is possible to achieve an accurate and fast solution for pose, without requiring an initial pose estimate and in very few iterations (it converges in about 4∼5 iterations and has an average standard deviation in the estimated pose of 2°).

2.1.2.1 The Anthropometric Model

In order to estimate the head pose, a 3D model of the same is required. For the head pose estimation it is used an anthropometric 3D rigid model of the human head. This model was acquired by a frontal laser scan of a physical model (figure 2.4(b)), selecting 58 3D landmarks that match those of the AAM 2D mesh. With the one-to-one point correspondences between the AAM (2D points) and the anthropometric model (3D points), the POSIT algorithm can be easily applied. As figure 2.5 shows, this anthropometric model is similar to the 2D mesh of the AAM. It has 58 points on the same features as the 3D mesh. The only difference is the third coordinate of this model, the Z coordinate.



(a) (b)

FIGURE 2.4: Images courtesy of Pedro Martins, [15].(a) Physical anthropometric model;(b) 3D laser scan data.

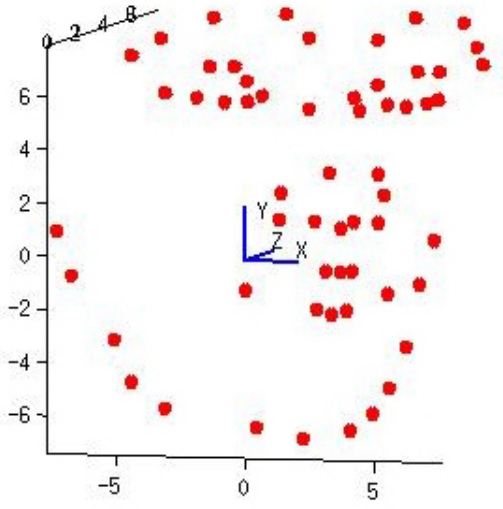


FIGURE 2.5: 3D points of the anthropometric model.

In this work, the orientation of the estimated pose is represented in *RPY* (Roll, Pitch and Yaw) angles. Rotations around the Y-axis are represented by Pitch angles, around the X-axis are represented by Yaw angles and around the Z-axis are represented by Roll angles.

2.1.3 Camera Calibration

As stated above, in order to estimate the head pose, some camera parameters need to be known. These parameters can be obtained from the camera calibration. Camera calibration is the process of finding the true parameters of the camera. These true parameters are represented by a matrix called the camera matrix K. Using this kind of information it is

possible to map points from world coordinates to pixel coordinates for example. The single calibration method that was used in this work is better described in appendix A.

2.2 Tracking the Eye

As the goal is to estimate the gaze under free head conditions, it is also necessary to track de movement of the eyes, independently of the head pose. To track the eyes, a circular edge detector is used in order to find both irises and respective centres.

2.2.1 Finding the Iris

The gaze estimation is based on two points (for each eye). They are: the centre of the eyeball (that will be explained further on chapter 3) and the centre of the iris. Notice that the center of the iris is the same as the center of the pupil.

The circular edge detectors that are most used are the Daugman's integro-differential operator and the Hough transform. There are some drawbacks with the Hough transform [2]. It requires threshold values to be chosen for edge detection, and this may result in critical edge points being removed, resulting in failure to detect circles/arcs. It also requires an intensive computational work, what may not be suitable for real-time applications. Therefore, it was decided to use Daugman's integro-differential operator because it works with raw derivative information, with no need of threshold values. Though, this method can be very sensitive to noise in the eye image, such as reflection and lack of illumination. Basically, the Daugman's algorithm [1] scans the eye image in order to locate the iris and pupil regions. These regions can be found by using a coarse-to-fine strategy with a single-pixel precision estimation of the center coordinates and radius of the iris and the pupil. These parameters can be determined by using the integro-differential operator:

$$max_{(r,x_0,y_0)} \left| G_\sigma(r) * \frac{\partial}{\partial r} \oint_{r,x_0,y_0} \frac{I(x,y)}{2\pi r} ds \right| \tag{2.15}$$

In equation (2.15), *I(x,y)* is an eye image as Fig. 2.6.

The parameter r is the radius to search for, $G_\sigma(r)$ is a Gaussian smoothing function and s is the contour of the circle given by r, x_0, y_0. The integro-differential operator searches for

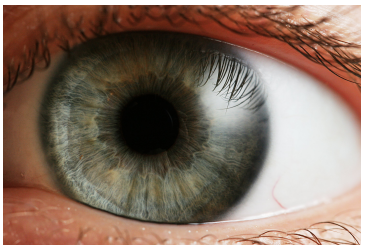


FIGURE 2.6: Example of an input image for the Daugman's algorithm.

the circular path where there is maximum change in pixel values, by varying the radius and centre (x, y) position of the circular contour. Basically, the complete operator behaves like a circular edge detector (it can be seen as a variation of the Hough transform), blurred at a scale set by σ, searching iteratively for the maximal contour integral derivative.

In practice the method is applied in a real-time video capture, so there is no guarantees of having high resolution and noiseless images. Therefore, some changes and small improvements need to be done on the Daugman's algorithm. On this next example, it is possible to understand how does this operator works.

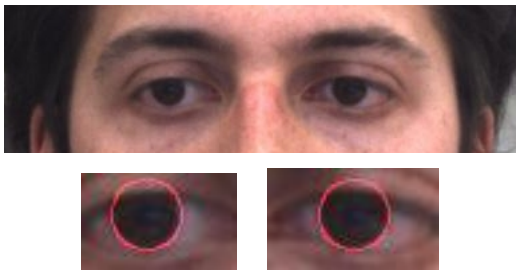
Algorithm 1 Iris Detection Algorithm

1: Read an eye image, **I** (like Figure 2.6)
2: Define a minimum and maximum radius, $\mathbf{R_{min}}$ and $\mathbf{R_{max}}$, for the Iris
3: Convert **I** to grayscale and normalize it to get pixel values between 0 and 1
4: Find the darkest pixels (local minimum pixels) and ignore those near to the border
5: Define $\mathbf{C(x,y)_i}$ = candidate centres
6: **for each** $C(x,y)_i$ **do** ▷ coarse search to find the best iris's center, within $C(x,y)_i$
7: $\sigma = \infty$
8: **procedure** SEARCH CENTER AND RADIUS(sigma)
9: **for each** R between R_{min} and R_{max} **do**
10: Compute **L**=lateral line integral, around a circular contour, in order to mitigate the effect of occlusions
11: **end for**
12: Calculate the approximate derivative **D** = [L(2)-L(1) L(3)-L(2) ... L(n)-L(n-1)]
13: **f(x)** = [1/7, 1/7, 1/7, 1/7, 1/7, 1/7, 1/7] , ▷ special case of the Gaussian filter, with sigma tending to ∞ (see equation (2.16))
14: **Blur** =**convulsion(D,f(x))** , where size(Blur)=size(D)
15: Find the maximum value of Blur , **[b,i] = max(Blur)**
16: return $\mathbf{b} = \mathbf{Blur(i)}$ and $\mathbf{r} = \mathbf{R(i)}$
17: **end procedure**
18: $\mathbf{M(C(x,y)_i)} = \mathbf{b}$
19: **end for**
20: Find the best candidate center, **C(x,y)**, such as $\mathbf{M(C(x,y))} = \mathbf{max(M(C(x,y)_i))}$
21: σ= **0.5** (default standard deviation for the Gaussian filter)
22: Define **Q** as being a jxj neighbourhood around the point **C(x,y)** ▷ in our case, a 10x10 neighbourhood is enough
23: **for each** pixel **in** Q **do** ▷ fine search to find the real center of the iris
24: **procedure** SEARCH CENTER AND RADIUS(sigma)
25: ▷ repeat the same process as describe on lines 9-16
26: **end procedure**
27: $\mathbf{M(C(x,y)_i)} = \mathbf{b}$
28: $\mathbf{Radii(C(x,y)_i)} = \mathbf{r}$
29: **end for**
30: $\mathbf{C_{iris}} = \mathbf{C(p_x, p_y)}$ where $\mathbf{M(C(p_x, p_y))} = \mathbf{max(M(C(x,y)_i))}$
31: $\mathbf{R_{iris}} = \mathbf{Radii(C_{iris})}$

$$f(x) = \frac{1}{2\pi\sigma^2} \cdot e^{-\frac{x^2+y^2}{2\sigma^2}} \tag{2.16}$$

$$\int_{-\infty}^{\infty} f(x)dx = 1, where\ f(x) = Gaussian\ distribution$$

It is possible to see the importance of doing only the lateral line integral (procedure "Search Center and Radius" on lines 8 and 24 of algorithm 1), by looking at figure 2.7 for example.

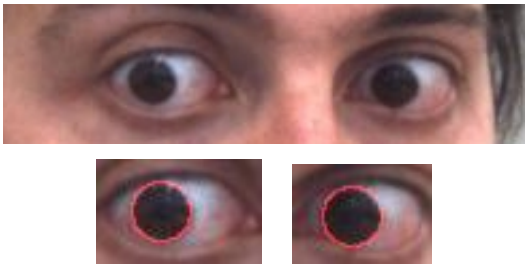


(a) (b)

FIGURE 2.7: Normal ROI of the eyes (showing partial occlusion of the iris). (a) Red circle showing the result on the left eye (b) Result on the right eye.

Another important parameter of the Daugman's algorithm is the range of radii of the iris, as described in line 2 of algorithm 1. The values of R_{min} and R_{max} must be defined according to the size of the eye image, otherwise the algorithm will fail, looking for circular contours with different dimensions of the real iris.

Most times, in a regular conversation, people have their eyelids occluding the circular contour of the iris, so it is not guaranteed that the eye images are always like figure 2.8.



(a) (b)

FIGURE 2.8: Eye's region with occlusions. (a) Red circle showing the result on the left eye (b) Result on the right eye.

As shown on these last figures and on figures from chapter 5 (results chapter), this integro-differential operator proved to be very accurate and robust. Notice that only with large distortion and/or huge reflections it becomes difficult (and sometimes impossible) to find the iris.

Chapter 3

Gaze Estimation

In order to estimate the 3D eye gaze with just a single camera, it is necessary to define a 3D structure for the eyeball. From this geometric structure we will define some anatomical parameters that differ from person to person, and that need to be calibrated at least once. Having these anatomical constants, it is possible to gather all the information obtained from chapter 2 and build an oriented vector that represents the eye gaze. So, this chapter can be divided in three main sections: the definition of the eye structure; the eye gaze calculation; and the calibration of the anatomical parameters.

3.1 The 3D Eye Model

In this work it is assumed that the eyeball is spherical. Actually, the eye isn't quite a sphere, but that doesn't affect the method neither the results. You can see a simple model of the eye defined in figure 3.1, with some anatomical references. For each eye of each person will be defined four anatomical constants (as in [3]):

- R_0: radius of the eyeball .
- L: depth of the centre O of the eyeball relative to the plane containing the eye corners.
- (T_x, T_y): offset between the mid-point of the eye corners and the eyeball centre O, when the face is frontal to the camera (meaning $roll = pitch = yaw = 0°$).

Notice that all of these constants are defined in pixels, in an image where the scale S of the face is 1. There are several ways of estimating the scale, the most obvious way is by using

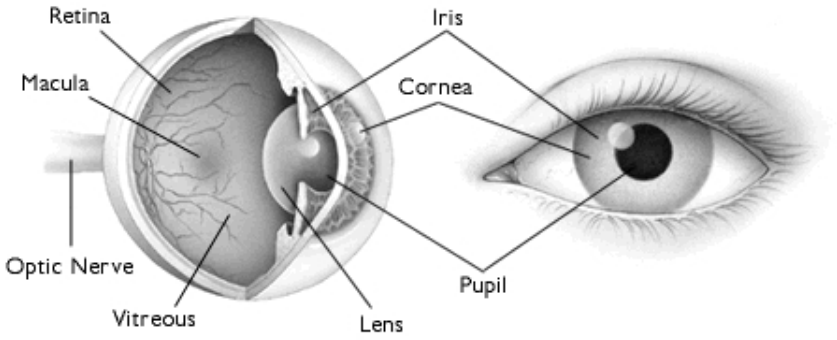


FIGURE 3.1: Eye anatomy

the foreshorten-corrected distance between the eye corners. To parametrize the eye region, it is considered the geometric model represented in figure 3.2.

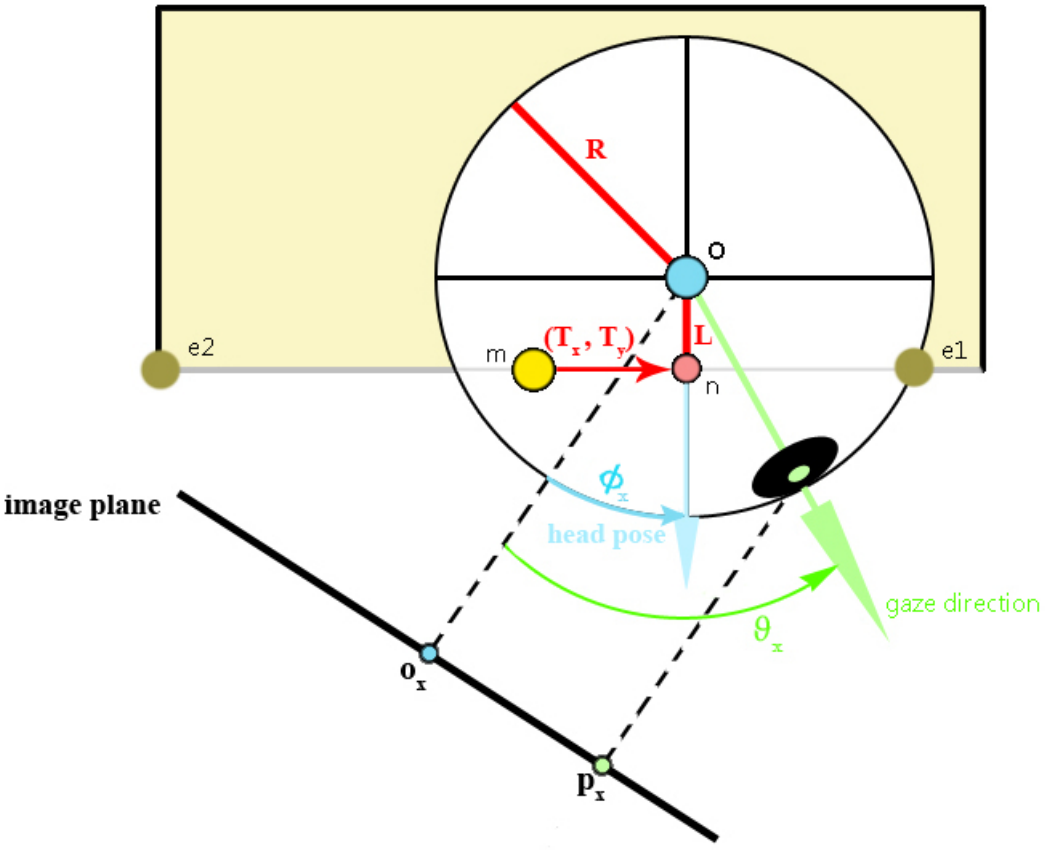


FIGURE 3.2: Geometric model adopted to estimate the gaze direction

3.2 Gaze Estimation

Considering all the variables defined on the geometric model of figure 3.2, the gaze direction can be computed simply by using equation (3.1), as described in [3].

$$\begin{pmatrix} \sin\theta_x \\ \sin\theta_y \end{pmatrix} = \begin{pmatrix} \frac{p_x - o_x}{\sqrt{R^2 - (p_y - o_y)^2}} \\ \frac{p_y - o_y}{\sqrt{R^2 - (p_x - o_x)^2}} \end{pmatrix} \tag{3.1}$$

The variables p_x and p_y are the iris centre (in pixel values) given by the Daugman's algorithm, and the angles θ_x and θ_y represent the gaze orientation of the eye on the X-axis and Y-axis, respectively.

The center of the eyeball $O = \begin{pmatrix} o_x \\ o_y \end{pmatrix}$ is computed by applying two corrections on the mid-point (m_x, m_y) of the eye.

$$\begin{pmatrix} o_x \\ o_y \end{pmatrix} = \begin{pmatrix} m_x \\ m_y \end{pmatrix} + S \begin{pmatrix} T_x \cos\phi_x \\ T_y \cos\phi_y \end{pmatrix} + SL \begin{pmatrix} \sin\phi_x \\ \sin\phi_y \end{pmatrix} \tag{3.2}$$

In equation (3.2), (ϕ_x, ϕ_y) is the head pose. The first correction is a foreshortened offset that compensates for the fact that the mid-point m of the eye isn't necessarily the eye center, even for a frontal image. The second correction is a compensation for the fact that the eyeball center does not lie in the plane of the eye corners.

The mid-point (m_x, m_y) calculation is very straightforward, depending only on the inner corner $(e1_x, e1_y)$ and the outer corner $(e2_x, e2_y)$ of the eye:

$$\begin{pmatrix} m_x \\ m_y \end{pmatrix} = \begin{pmatrix} \frac{e1_x + e2_x}{2} \\ \frac{e1_y + e2_y}{2} \end{pmatrix} \tag{3.3}$$

The eyeball centre estimation, in equation (3.2), requires the use of the scale factor S, because for each frame the scale of the face changes, and the anatomical values (T_x, T_y) and L were only calculated for $S = 1$. As said before, this scale is calculated using the foreshorten-corrected distance between the eye corners:

$$S = \frac{\sqrt{(e1_x - e2_x)^2 + (e1_y - e2_y)^2}}{\cos\phi_x} \tag{3.4}$$

Finally, the radius of the eyeball R in equation (3.1) is the result of:

$$R = SR_0 \tag{3.5}$$

Replacing equation (3.1) in order to obtain an expanded version of it:

$$\begin{pmatrix} \sin\theta_x \\ \sin\theta_y \end{pmatrix} = \begin{pmatrix} \frac{p_x - m_x - ST_x\cos\phi_x - SL\sin\phi_x}{\sqrt{(SR_0)^2 - (p_y - o_y)^2}} \\ \frac{p_y - m_y - ST_y\cos\phi_y - SL\sin\phi_y}{\sqrt{(SR_0)^2 - (p_x - o_x)^2}} \end{pmatrix} \tag{3.6}$$

Remember that the anatomical values (T_x, T_y), L and R_0 are constant and unique for each person, and obtained on an offline calibration phase.

After knowing all the variables values, it is possible to finally compute the gaze direction simply by manipulation of equation (3.1):

$$\begin{pmatrix} \theta_x \\ \theta_y \end{pmatrix} = \begin{pmatrix} \sin^{-1}\left(\frac{p_x - o_x}{\sqrt{R^2 - (p_y - o_y)^2}}\right) \\ \sin^{-1}\left(\frac{p_y - o_y}{\sqrt{R^2 - (p_x - o_x)^2}}\right) \end{pmatrix} \tag{3.7}$$

3.2.1 Filtering the Gaze Direction values

Working with a real-time capture makes most of the output data vulnerable to noise, perturbations and other factors that can induce error into the system.

The gaze estimator mainly depends on the output of two different estimators: the head-pose estimator and the iris center estimator. Both of them have associated errors, that in practice will cause a bigger error and instability on the gaze output. To attenuate this error, some kind of filter must be used. The goal is to suppress gaze spikes and steady the gaze direction value. The most adequate filter for this situation is the discrete Kalman filter, which has two main steps: the estimation of the subsequent model state (prediction) and the adjustment of the model state (correction) [11]. Basically this filter acts like an optimal estimator [10], minimizing the mean square error of the estimated parameters. This filter is widely used in online real time processing (e.g. in tracking systems).

The Kalman filter is explained in detail in appendix B.

3.3 Training the anatomical constants

To train the anatomical constants, it was initially adopted the method proposed in [3]. This method consists in saving several frames with different head poses. Basically, the user is asked to look at N_x points on the X-axis (with no head movement on the Y-axis), and N_y points on the Y-axis (with no head movement on the X-axis):

$$(\theta_x, \theta_y, \phi_x, \phi_y) = (\alpha_x^i, 0, \beta_x^i, 0) \tag{3.8}$$

and:

$$(\theta_x, \theta_y, \phi_x, \phi_y) = (0, \alpha_y^j, 0, \beta_y^j) \tag{3.9}$$

where α_x^i represents the known angle of the sample point i, and β_x^i the pitch angle (X-axis) given by the POSIT. The same analogy is applied to α_y^j and β_y^j, but on the vertical axis.

As it is known that this offline calibration is only done in the forms presented on equations (3.8) and (3.9), it is possible to linearise equation (3.6) by assuming that equation (3.8) implies $p_y - o_y = 0$ and that equation (3.9) implies $p_x - o_x = 0$. This linearisation results on a simple matrix equation:

$$\begin{bmatrix} \frac{p_x^1 - m_x^1}{S_1} \\ \frac{p_x^2 - m_x^2}{S_2} \\ \vdots \\ \frac{p_x^{N_x} - m_x^{N_x}}{S_{N_x}} \\ \frac{p_x^1 - m_x^1}{S_1} \\ \frac{p_y^2 - m_y^2}{S_2} \\ \vdots \\ \frac{p_y^{N_y} - m_y^{N_y}}{S_{N_y}} \end{bmatrix} = \begin{bmatrix} \sin\alpha_x^1 & \sin\beta_x^1 & \cos\beta_x^1 & 0 \\ \sin\alpha_x^2 & \sin\beta_x^2 & \cos\beta_x^2 & 0 \\ \vdots & \vdots & \vdots & \vdots \\ \sin\alpha_x^{N_x} & \sin\beta_x^{N_x} & \cos\beta_x^{N_x} & 0 \\ \sin\alpha_y^1 & \sin\beta_y^1 & 0 & \cos\beta_y^1 \\ \sin\alpha_y^2 & \sin\beta_y^2 & 0 & \cos\beta_y^2 \\ \vdots & \vdots & \vdots & \vdots \\ \sin\alpha_y^{N_y} & \sin\beta_y^{N_y} & 0 & \cos\beta_y^{N_y} \end{bmatrix} \begin{bmatrix} R_0 \\ L \\ T_x \\ T_y \end{bmatrix} \tag{3.10}$$

The least squares solution of this equation gives the anatomical constants (R_0, L, T_x, T_y).

To train these samples, a board with fixed points was built, parallel to the camera XY plane. This board had, originally, ten points on the X-axis and other ten points on the Y-axis. Each one of them had a respective known angle α. However, for some unknown reason, the resulting anatomical constants from this training method weren't what was expected. The

resulting anatomical values of each offline training were always very different for the same person, and the system was very sensitive to the head position. It was also considered that this training method can be very exhaustive due to the amount of samples that is needed. A different method can be applied in this situation.

The proposed method will replace all of the points on the board by only two points. It is also added an additional point in the center of the camera. Using part of equation (3.10), it is true that:

$$T_x = \frac{p_x - m_x}{S \cos \beta_x} \tag{3.11}$$

and

$$T_y = \frac{p_y - m_y}{S \cos \beta_y} \tag{3.12}$$

Now, as in [5], if it is mandatory that the user keeps the head still with $roll = pitch = yaw = 0°$, it is possible to estimate T_x and T_y using just one point (that is the center of the camera), and equations (3.11) and (3.12) can be simplified to:

$$T_x = \frac{p_x - m_x}{S} \tag{3.13}$$

and

$$T_y = \frac{p_y - m_y}{S} \tag{3.14}$$

The other two points are situated on the horizontal axis of the camera, one on each side of it. These points have a well known distance to the camera. This configuration allows to use the method described in [4] to calculate the radius of the eyeball R_0. If it is assumed that the eye is rotating about the center of the eyeball, from point A to B through an angle γ, then the pupil center will be directly related to the radius of the eyeball. This process is illustrated in figure 3.3.

In figure 3.3, the points X and Y correspond to the projection of the iris center, that is easily estimated using the Daugman's algorithm described in Chapter 2. So, the radius R_0 can be calculated by:

$$R_0 = \overline{CX} = \frac{\overline{XY}}{2 \sin \frac{\gamma}{2}} \tag{3.15}$$

Finally, the depth L is taken as an anatomical average, since it is a very small value and its variation is very small from person to person. These average anatomical values are easy

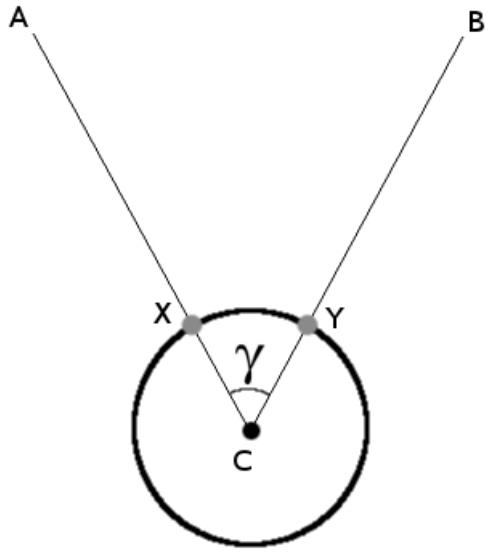


FIGURE 3.3: Rotation of the eye around C, from a fixed point A to a fixed point B through a known angle γ.

to find in medical literature. Figure 3.4 shows the average dimensions of an adult human eye. From this figure it is possible to define a constant λ that relates the radius of the eyeball R_0 with the depth L.

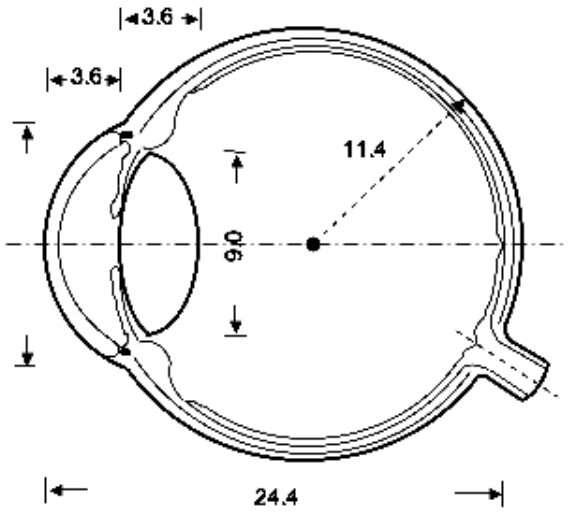


FIGURE 3.4: Adult eye dimensions, in millimetres (image from Craig Blackwell M.D. Ophthalmology website)

$$\lambda = \frac{R_{avg}}{L_{avg}} = \frac{11,4}{3,6} \approx 3,1667 \tag{3.16}$$

It is true that using only three points to calibrate the anatomical constants isn't the most appropriate method, since the slightest failure in one of the training samples can cause a large error on the resulting parameters. On the other hand, the use of a small number of samples reduce the probability of the user doing mistakes during the training. And also, by imposing some restrictions to the user, it is possible to make this method more resistant to human errors.

Resuming, the training process consists on the following steps:

1. Ask the user to sit directly in front of the camera and look at its lens, holding his head parallel to the image plane, i.e. with $roll = pitch = yaw = 0°$;
2. Calculate and save T_x and T_y;
3. Ask the user to keep the same head pose, and using only the eyes, look at a fixed point situated 40 cm to the left of the camera;
4. On the same conditions, look at a fixed point situated 40 cm to the right of the camera;
5. Calculate and save R_0;
6. Calculate and save $L = \frac{R_0}{\lambda}$.

Chapter 4

Visual Contact Evaluation

In this last phase, the goal is to be able to evaluate the visual contact between two persons in a frontal conversation. Meaning that the goal is to know whether one person is looking or not to the other person's eyes. This kind of evaluation requires the combination of all the information cited in the previous chapters plus the person-camera spacial information. To achieve this it is necessary to set up a specific stereo camera configuration.

4.1 Stereo Camera Calibration

In order to assess the situations of visual contact between two actors in the process of conversation, it is required to have a proper stereo configuration with frontal cameras. This kind of configuration is illustrated in figure 4.1. Notice that both cameras are only aware of the position (or even existence) of the person that lies inside their field of view (FOV).

It is absolutely essential to calibrate the stereo system in order to obtain the rotation and translation matrices that relate both cameras. The principle of stereo calibration with patterns says that both cameras must collect the same pattern at the same time for the calibration to be correct. The usual methods to calibrate two cameras use a chess pattern similar to the one shown in figure A.2. However, this stereo configuration makes impossible to use such board because only one camera would be able to "see" the pattern. There are some methods for multi-camera calibration ([7] for example), using light sources, that could possibly be adapted to the configuration of figure 4.1, but there are no guarantees of success.

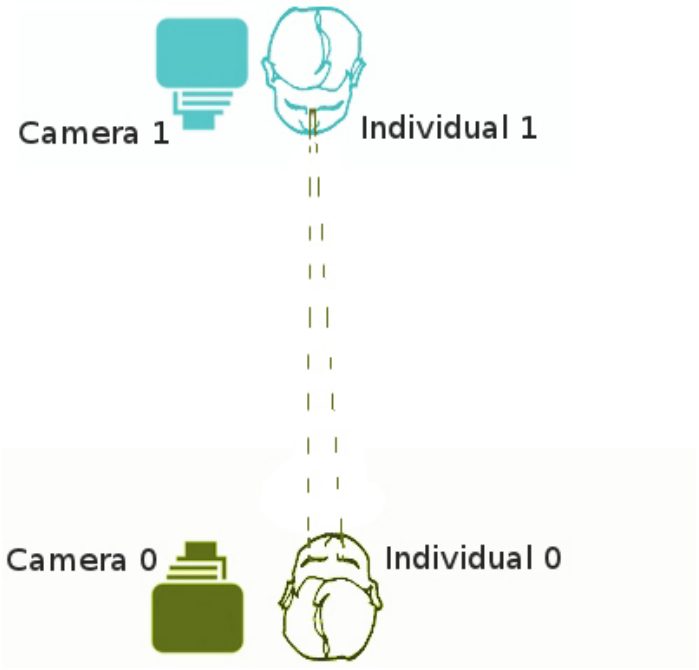


FIGURE 4.1: Example of the desired stereo configuration.

Instead, a new methodology was created from the adaptation of the regular method used to calibrate a pair of coplanar cameras.

It is possible to make the chess pattern of figure A.2 visible to both cameras at the same time, if a acetate sheet is used instead of a regular paper. So basically, if the printing is done on a "transparent paper", the pattern will be visible on both sides of the paper. The chess pattern used to calibrate the stereo system has black and white squares with size dX and dY, along de horizontal and vertical directions. To make sure that the acetate sheet do not bend when collecting the frames for calibration, the pattern was placed between two acrylic plates. The resulting pattern is shown in figure C.1.

Having this chess frame, it is possible to use the standard stereo calibration on this frontal camera system.

Using the Camera Calibration Toolbox developed by Jean-Yves Bouguet [8], the stereo calibration becomes quite easy to perform. The first thing to do is to place the cameras in the desired configuration, knowing that they should no longer be moved thereafter. Using the pattern of figure C.1, it is necessary to collect a set of frames for each camera (always having the patterns inside the field of view). There are no maximum number of frames and it is advisable to acquire at least fifteen images in order to get a good calibration. Finally, using

the intrinsic parameters of each camera and their transformations regarding every pattern pose that was made, and using the Jean-Yves Bouguet toolbox, the stereo calibration can finally be done.

On figure 4.2 is possible to see the real configuration of the stereo system. The camera 0 is the reference camera for the stereo calibration, meaning that the resulting rotation R and translation t will have the referential system of camera 0 as base referential: ${}^{0}_{1}R$ and ${}^{0}_{1}t$. Basically, the Bouguet's toolbox [8] estimates the transformation between the two cameras

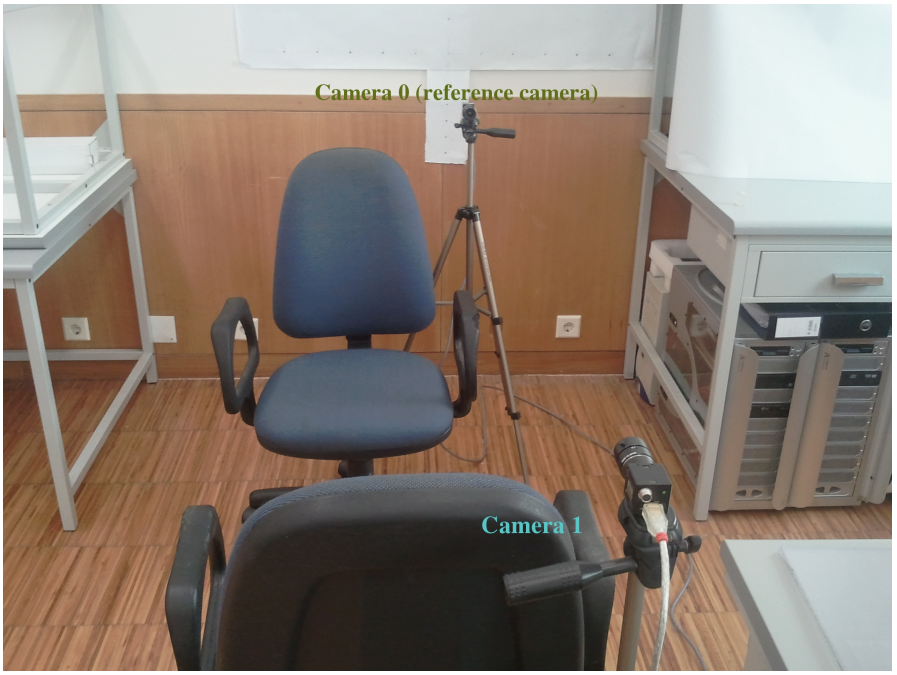


FIGURE 4.2: The real assembly of the stereo system.

making a stereo par.

Despite the method viability, this configuration revealed not to be very comfortable to the user due to its lack of movement freedom. The user has to be very careful not to get in the way of his closest camera field of view, and at the same time he has to make sure that he lies inside the field of view of the other camera. Furthermore, as the cameras are supported by tripods, it is very easy to accidentally touch them and change the configuration, messing the stereo calibration. The ideal was to have both cameras attached to a suspended framework, over the individuals (which is hard to accomplish).

To avoid future complications with this stereo configuration, a new one was adopted. Both cameras were attached to a framework and brought to the middle of the conversation, like figure 4.3 illustrates.

FIGURE 4.3: New configuration of the stereo system.

This configuration has several advantages: as the cameras are fixed to the same framework, it is possible to move them without changing their position relatively to each other; as they stay in the middle of the conversation, the user will have much more movement freedom than before; both cameras will stay much closer to their opposite individual. The main disadvantage of this configuration is the calibration. In this configuration the cameras don't have a common FOV, what makes impossible to calibrate them with the usual methods. Specifically for this kind of situations, a different calibration method is proposed in [43]. This calibration algorithm is based on the mirror reflections of a calibration pattern. Having a N number of images ($N \geq 3$) of the pattern reflected in the mirror, it is possible to estimate the camera pose with respect to the pattern position. For the specific camera configuration of figure 4.3, let's imagine that camera 0 can "see" the pattern and camera 1 can't. Using a mirror and the method of [43] it is possible to compute ${}^{pattern}T_1$. Performing a single camera calibration for camera 0 (with Bouguet's toolbox) it is easy to compute ${}^{0}T_{pattern}$. Then, the transformation between the cameras, ${}^{0}_{1}T$, is straightforward:

$$ {}^{0}T_1 = {}^{0}T_{pattern}\ {}^{pattern}T_1 \tag{4.1} $$

The stereo calibration will allow to correspond image points between cameras. To read more about the meaning of stereo calibration, see appendix C.

4.2 Eye Contact Evaluation

To be able to know if there is any visual contact between the two users, both cameras must know the exact locations of both users. Considering the configuration shown in figure 4.1, it is obvious that the cameras do not know the location of their respective users (i.e. camera 0 doesn't know where individual 0 is). Such knowledge is possible do acquire if the information provided by the POSIT (chapter 2) and the stereo calibration are combined.

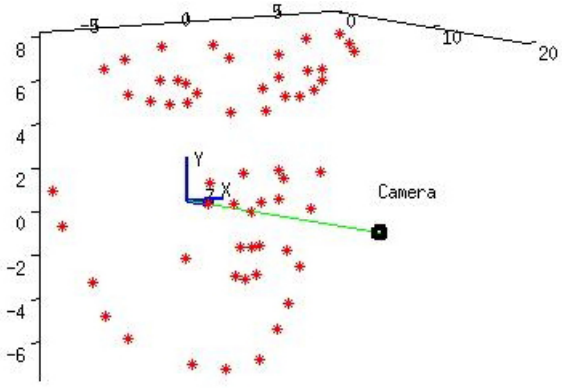


FIGURE 4.4: Direction vector (in green), from the camera to the head's coordinate system, that describes the translation given by the POSIT.

If it is possible to know the head coordinate system position relatively to the camera, then it is also possible to access any head point real position (since it belongs to the anthropometric model).

4.2.1 3D Projection of the Iris

The visual contact evaluation will be defined by the intersection of the gaze vector in 3D space with the opposite user's eyes. This gaze vector is defined by a 3D point plus an orientation. The gaze orientation of each eye is already known, but the 3D point isn't. This point can be either the center of the eyeball or the center of the iris. However, these points are only available in pixel coordinates. Let's use the iris center instead of the eyeball center.

The projection from a 2D image point to a 3D point can be easily done by using the intrinsic and extrinsic parameters returned by the single camera calibration. See appendix C, section C.2, for a better understanding on the subject.

For simplicity let's assume that the depth Z_w of the iris is the same as one specific point of the eyelid. Using the anthropometric model defined in chapter (2), it is possible to extract the 3D position of one point on each eyelid. Both eyelid points, $eyeLid_1$ and $eyeLid_2$, have

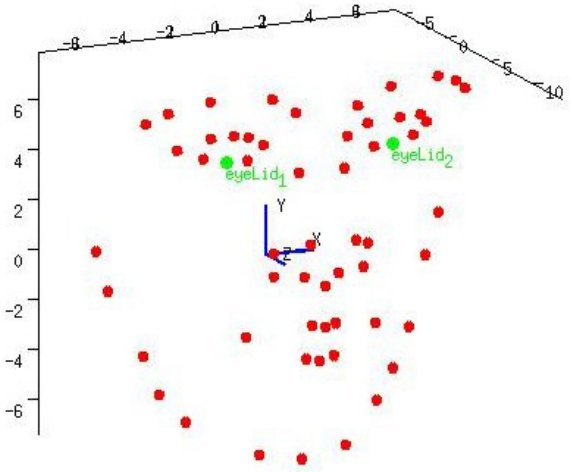


FIGURE 4.5: Two 3D points extracted from the eyelids (green points).

X, Y and Z coordinates (the calculation of these coordinates is explained in detail on the next section "Eye Contact Evaluation"). Assuming then that $Z_w = Z_{w_{iris}} = Z_{eyeLid}$ and using the equations defined in appendix C, section C.2, it is possible to simplify equation (C.15) as follows:

$$\begin{cases} \frac{\lambda x_c}{\lambda} = \frac{fX_w + c_x Z_w}{Z_w} \\ \frac{\lambda y_c}{\lambda} = \frac{fY_w + c_y Z_w}{Z_w} \end{cases} \Leftrightarrow \begin{cases} x_c = \frac{fX_w}{Z_w} + c_x \\ y_c = \frac{fY_w}{Z_w} + c_y \end{cases} \tag{4.2}$$

Now the calculation of the 3D position of the iris becomes straightforward:

$$\begin{cases} X_{w_{iris}} = \frac{(x_{c_{iris}} - c_x) Z_w}{f} \\ Y_{w_{iris}} = \frac{(y_{c_{iris}} - c_y) Z_w}{f} \\ Z_{w_{iris}} = Z_w \end{cases} \tag{4.3}$$

4.2.2 ROI of the Eyes in 3D Space

It's already known that for any point of the 3D rigid head model is possible to estimate its position with respect to the camera coordinate system. To define the ROI of the eyes of an head, the same eyelid points of figure 4.5 are going to be used. Thus, like figure 4.4, the vectors from the camera origin to the eyelid points are known (figure 4.6).

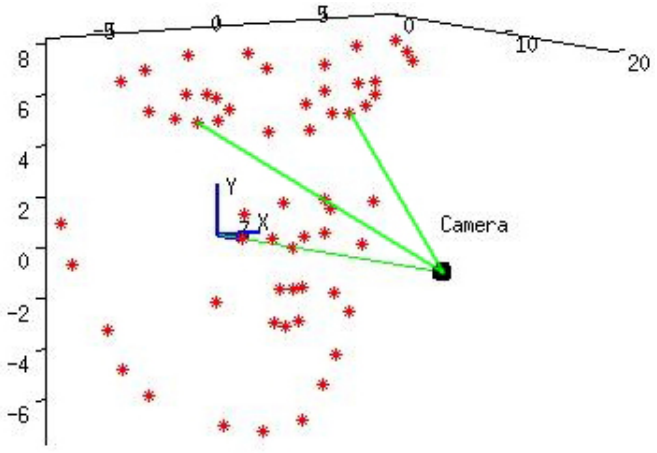


FIGURE 4.6: Two new vector from the camera coordinate system to both eyelid points.

From the eyelid points, a virtual rectangle is defined in order to cover, by excess, both eyes. 4.6).

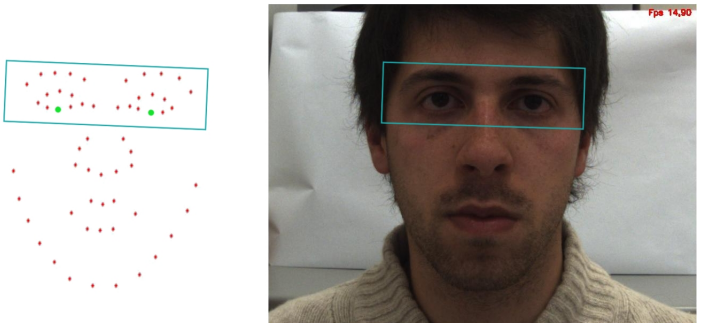


FIGURE 4.7: Virtual rectangle defined around the eyes.

For simplicity, the eyes region is always defined as being parallel to the image plane. This is done by considering the depths Z_1 and Z_2 of each eyelid, and defining the ROI depth by:

$$Z_{roi} = \frac{Z_1 + Z_2}{2} \tag{4.4}$$

In practice this means that if the user is performing *Pitch* rotations, the eyelids will have different depths, but the eyes region will continue to be parallel to the image plane, and won't intersect any eyelid. This situation is illustrated in figure 4.8.

FIGURE 4.8: Top view of the user, with the eyes virtual rectangle represented by the blue line.

This assumption won't affect the accuracy of the system and will make it simpler.

4.2.3 Visual Contact Classification

To conclude whether there is visual contact or not, the gaze vector needs to be intersected with the eyes Z-plane. This plane is defined by the depth of the virtual rectangle around the eyes. Notice that the cameras are unaware of the position of their respective users, which means, for example, that Camera 0 doesn't know where user 0 is, and consequently where his eyes are. However, with the data returned by the stereo calibration mentioned earlier, it is possible to know this information. camera 0 knows the gaze direction of individual 1, but is unaware of the eyes ROI of individual 0. However, camera 1 knows where the individual 0 eyes are. As said before, the eyes ROI are defined by a rectangle. This rectangle has width w, height h, a top left corner q_{left}, a bottom right corner q_{right} and a depth Z_{roi}. Camera 1 has access to all of this information since it can "see" individual 0. Therefore let's assume that the position of the camera 1 in the camera 0 coordinate system is described by ${}^{0}_{1}R$ and

${}^0_1t$, and that in camera 1, the eyes ROI of individual 0 is given by:

$$ {}^1q_{left} = (X_{left}, Y_{left}, Z_{roi}) \tag{4.5} $$

and

$$ {}^1q_{right} = (X_{right}, Y_{right}, Z_{roi}) \tag{4.6} $$

The position of this ROI in camera 0 will then be given by:

$$ {}^0q_{left} = {}^0_1 R \cdot {}^1 q_{left} + {}^0_1 t \tag{4.7} $$

and

$$ {}^0q_{right} = {}^0_1 R \cdot {}^1 q_{right} + {}^0_1 t \tag{4.8} $$

At this point, camera 0 knows the gaze direction of individual 1 and where the eyes of individual 0 are. Now, it is just a question of intersecting the gaze vector with the eyes plane. A straight line is well defined by a point and an orientation, therefore the gaze direction (in both eyes) can be defined by the 3D point of the iris (already computed in subsection 4.2.1) and the gaze direction (computed in chapter 3). The line equation of gaze in the 3D space can be represented by the parametric form:

$$ \begin{cases} X = at + X_{w_{iris}} \\ Y = bt + Y_{w_{iris}} \\ Z = ct + Z_{w_{iris}} \end{cases} \tag{4.9} $$

Where (a, b, c) is the direction vector. On figure 4.9 the geometric definition of the direction vector is represented. Notice that C is the camera center, and that the coordinate system is left handed due to the flipping of the image (if it was right handed, the equations would remain the same). The direction vector will then be given by:

$$ \begin{cases} a = sin(\theta_x) \\ b = tan(\theta_y)cos(\theta_x) \\ c = -cos(\theta_x) \end{cases} \tag{4.10} $$

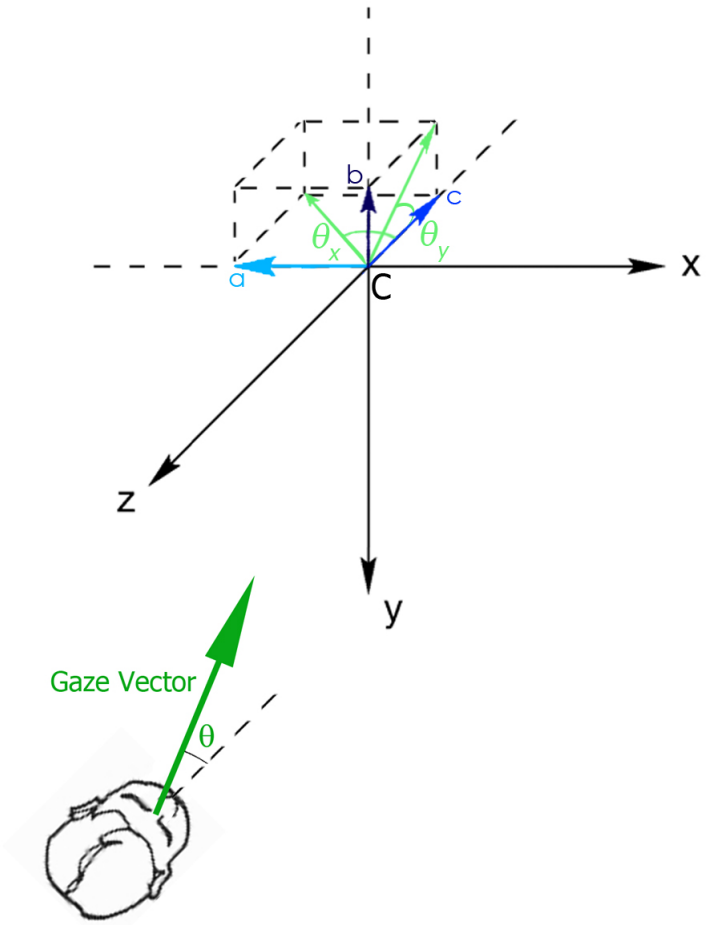


FIGURE 4.9: Geometric definition of the direction vector. The vector defined by θ_x is unitary.

The intersection of the gaze line (for each eye) of individual 1, with the eyes ROI of individual 0, will define whether there is unilateral eye contact or not:

$$Z_{roi} = ct_{intersect} + Z_{w_{iris}} \Leftrightarrow t_{intersect} = \frac{Z_{roi} - Z_{w_{iris}}}{-cos(\theta_x)} \tag{4.11}$$

$$\begin{aligned} X &= sin(\theta_x)t_{intersect} + X_{w_{iris}} = -tan(\theta_x)(Z_{roi} - Z_{w_{iris}}) + X_{w_{iris}} \\ Y &= tan(\theta_y)cos(\theta_x)t_{intersect} + Y_{w_{iris}} = -tan(\theta_y)(Z_{roi} - Z_{w_{iris}}) + Y_{w_{iris}} \end{aligned} \tag{4.12}$$

The point of focus $P_f = (X, Y, Z_{roi})$ in the Z_{roi} plane is given by the middle point between the intersection point of the left gaze line and the intersection point of the right gaze line.

If the point of focus P_f of individual 0, lies inside the eyes ROI of individual 1 (and vice-versa), then there is visual contact.

Chapter 5

Experimental Results

This chapter will be divided in four main sections that sequentially correspond to the work developed along all the project.

5.1 AAM and POSIT Results

Recalling chapter 2, the head pose estimation is computed by making correspondences between 3D points of an anthropometric rigid model and 2D points of the AAM. The active appearance model needs an offline training of the shape model and the POSIT algorithm needs the camera focal distance and principal point as input. Thus, a camera calibration must be done before computing the head pose.

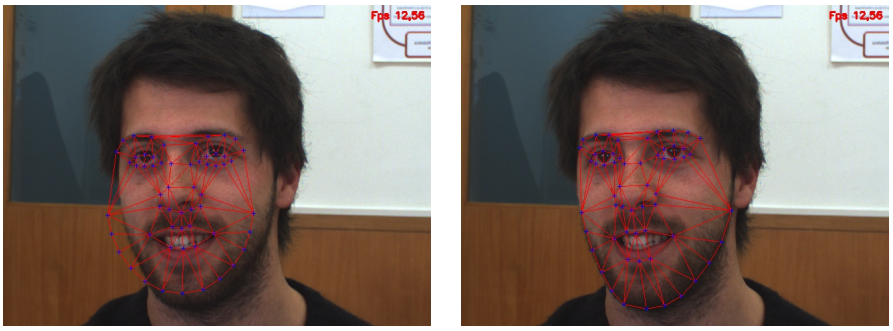
5.1.1 Offline Training of the Shape Model

The user is asked to perform a set of head poses and expressions, in different lightning conditions and with different face appearances (e.g. with and without facial hair), in order to introduce more variability in the model. Figure D.1 shows an example of a good training set. Then, the training images must be marked by hand (all the 58 points on each image).

After annotating all the training set, a shape model file is created to be used by the AAM.

5.1.2 AAM Fitting

Using the shape model created in the offline phase, it is possible to detect and track every 58 feature points of the face that are included in the shape model. As figure 5.2 illustrates, the



(a) (b)

FIGURE 5.1: Annotation of a training image; (a) base shape mesh.(b) after marking all the 58 points by hand.

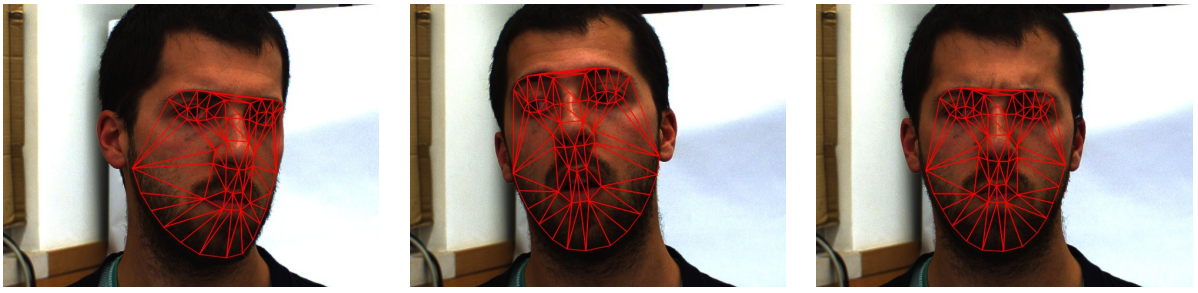


FIGURE 5.2: Some examples of AAM perfect fitting, with SIC algorithm, in frames that don't belong to the training set.

AAM will detect the shape mesh of the user based on his training shape model. The problem with this active appearance model is that it only works for people within the database (it is not adaptive). Figure D.2 illustrates this issue.

Notice that a more robust and accurate AAM fitting algorithm could be used in this project, but it would be much slower than SIC algorithm. Nevertheless, in figure A.3 it is possible o see the difference between SIC algorithm and the robust version of the same.

5.1.3 Single Camera Calibration

Using the calibration pattern of figure A.2, 25 images were collected, like figure A.1 illustrates, in a camera resolution of 640x480 pixels.

Then, the four grid corners were extracted (manually) from each one of them (keeping the grid origin on the same corner for every image). The pattern grid has 6 squares in the

X direction and 9 in the Y direction. Each square is 20 mm on each side. See figure D.3 for better understanding of the corner extraction process.

After doing the corner extraction process for all 25 images, the calibration parameters are directly computed from the Bouguet's toolbox [8]. The calibration returns many parameters. Among them, there is one transformation matrix for each image, that relates the camera with the pattern. Having these matrices, is easy to confirm if the calibration is coherent with the captured set of images. In figure 5.3 are represented the extrinsic parameters of the camera, relatively to every image.

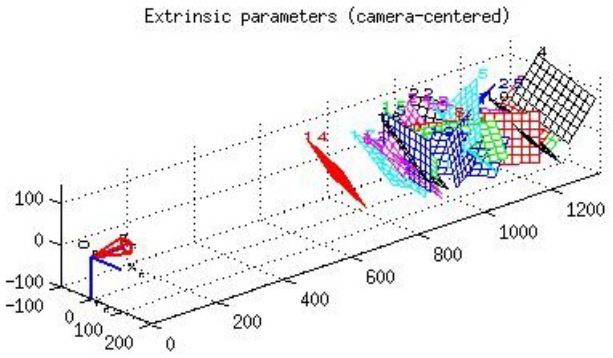


FIGURE 5.3: Relative positions of the grids with respect to the camera.

The relevant camera parameters in this phase are the camera focal distance f and principal point $C = (c_x, c_y)$. In this calibration, the extracted parameters for the POSIT algorithm were:

$$f \approx 1805.881\ px \tag{5.1}$$

and

$$C \approx (289.6, 248.0)\ px \tag{5.2}$$

The reprojection error of this calibration was in the order of sub-pixels as figure D.4 illustrates. Despite the C value that was obtained, the principal point was considered to be the same as the image center. In this case: $C = (320, 240)\ px$.

5.1.4 Head Pose Estimation

Having the camera parameters and the 3D rigid anthropometric model defined, it is possible to compute the monocular head pose using the POSIT algorithm.

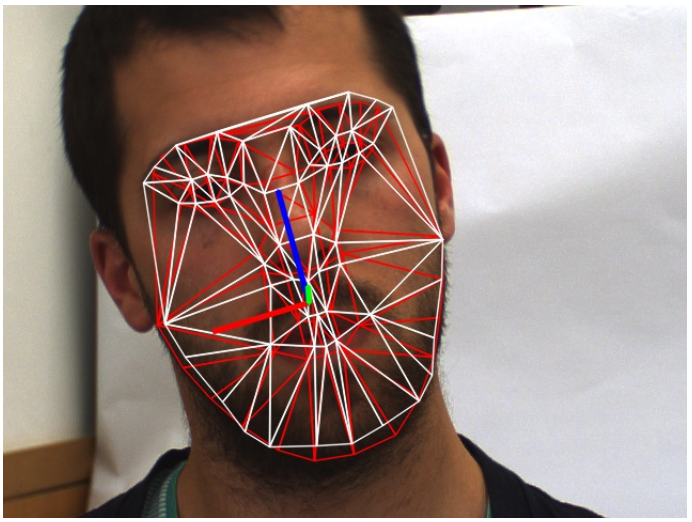


FIGURE 5.4: Reprojection of the 3D points (white mesh) over the 2D shape mesh.

In figure 5.4 is shown the re-projections of the 3D points of the rigid model over the shape mesh of the AAM. Notice that the coordinate system has the X-axis flipped relatively to the anthropometric model. In reality, it is the same as the 3D model, it just looks this way because the image itself was entirely flipped. The POSIT algorithm returns a rotation matrix and a translation vector of the face with respect to the camera. Recalling chapter 2, the head rotation can be described by RPY angles: *Roll* angles around the Z-axis, *Pitch* angles around the Y-axis and Yaw angles around the X-axis.

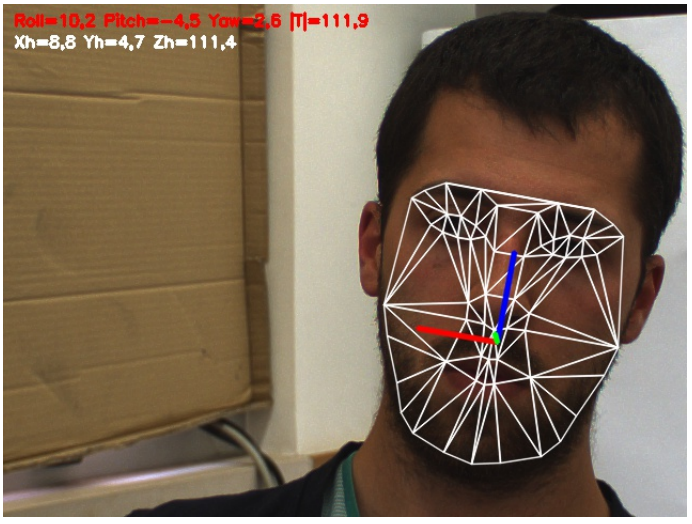


FIGURE 5.5: Examples of RPY angles in an arbitrary pose.

The value $|T|$ is the length of the vector from the camera origin to the head coordinate system. Let $P_h = (X_h, Y_h, Z_h)$ be the 3D point in the camera coordinate system, that represents the position of the head.Then:

$$|T| = \sqrt{X_h^2 + Y_h^2 + Z_h^2} \tag{5.3}$$

From this point on the 2D (red) and 3D (white) grids won't appear any more, to simplify the resulting images and reduce the computational cost.

5.2 Iris Detection and Tracking

The iris detection is done in every frame of the capture, resulting on a real time irises tracking. The iris tracker works with an eye image as input, which is defined by the AAM.

5.2.1 Locating the Eyes

From the AAM shape model, it is known that the 58 landmarks of the face always follow the same order. In figure 5.6 is shown the respective number (in the shape model) of each face feature.

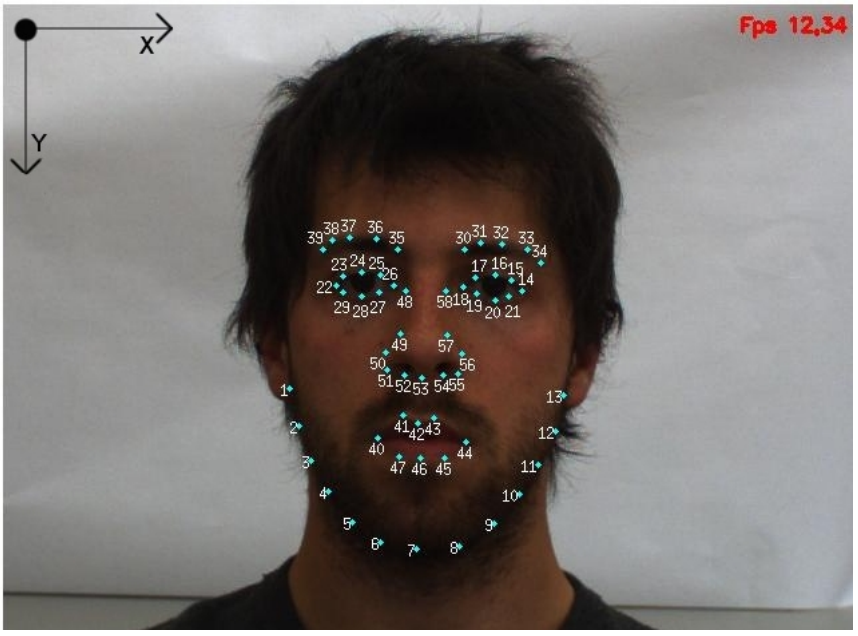


FIGURE 5.6: Number of each feature tracked by the AAM.

Knowing this classification, it is straightforward to extract any desired feature. Let's assume the position of all 58 features represented, in pixels, by:

$$L_i = (Lx_i, Ly_i) \ , \ i = 1, \ldots, 58 \tag{5.4}$$

Where Lx_i and Ly_i are the position, in pixels, of the feature i, in the image. Then, the ROI of the eyes is well defined by a rectangle with a top left vertex (v_{x_1}, v_{y_1}) given by:

$$(v_{x_1}, v_{y_1}) = (min_{i\in[14,29]}\{Lx_i\}, min_{i\in[14,29]}\{Ly_i\}) \tag{5.5}$$

and with a bottom right vertex (v_{x_2}, v_{y_2}) given by:

$$(v_{x_2}, v_{y_2}) = (max_{i\in[14,29]}\{Lx_i\}, max_{i\in[14,29]}\{Ly_i\}) \tag{5.6}$$

Applying (5.5) and (5.6) to image 5.6, the resulting ROI is shown in figure 5.7. The same

FIGURE 5.7: ROI of the eyes, from figure 5.6.

methodology can be applied to extract the eyes individually.

(a) (b)

FIGURE 5.8: Left and right eyes extracted from figure 5.6.

5.2.2 Iris Tracking with Daugman's Algorithm

The images shown in figure 5.8 are the input arguments for the Daugman's algortihm.

Knowing that the user has motion freedom, it must be considered that the movements in the Z-axis reduce the scale of the face, therefore reducing the size of eye images. This will modify the range of radii in which the integro-differential will operate. Assuming that

a clean eye image, with no other features but the eye itself, is extracted from the user's face during a regular conversation (about 1 meter away from the camera, approximately), then the R_{min} and R_{max} values (in pixels) can be fixed at:

$$R_{min} = 9\,px \tag{5.7}$$

and

$$R_{max} = 13\,px \tag{5.8}$$

As this is a discrete operation, the lateral line integral pointed in algorithm 1 of chapter 2 is computed incrementally. For a better understanding, imagine the iris as a circle cut into several slices. The integral is the sum of every point in the circle contour. As this is a discrete system, a number of sample points in the circle contour must be defined. Obviously, the more sample points, the slower the algorithm gets. Let's represent the number os sample points in the circular contour by N. Due to the small dimension of the eye images, the value of N does not need to be very high, so: $N = 200$.

An example of the iris detection from a regular face image is shown in figure 5.9.

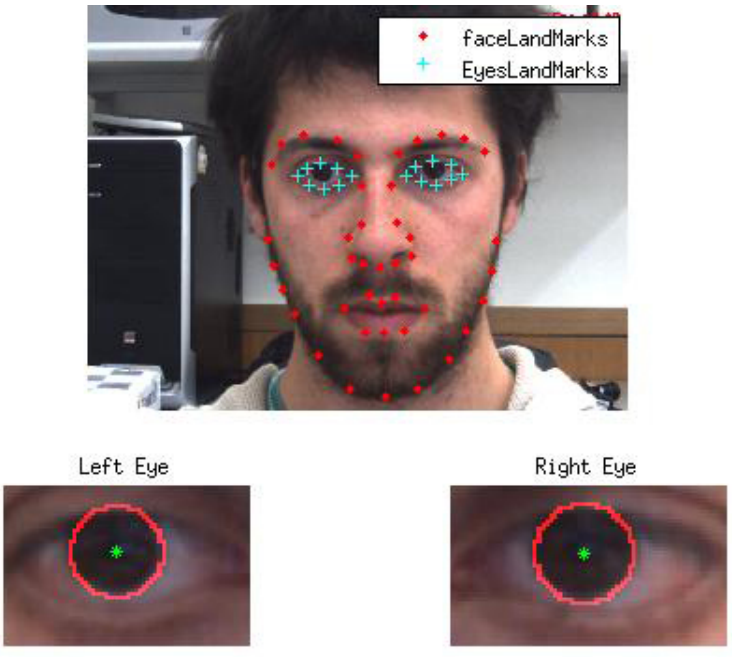


FIGURE 5.9: Iris detection process, starting with a face image, followed by the ROI extraction and iris center localization.

On Fig. 5.10 is possible to see some results on several eye images. These images vary in size and lighting conditions. Even with distortion and iris occlusion, it is still possible to accurately detect the iris centre. The Daugman's algorithm is computed in every frame of

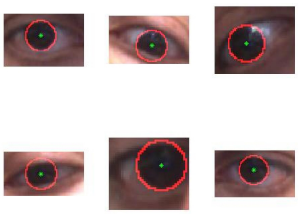


FIGURE 5.10: Daugman's algorithm on several eye images, with and without occlusions, and on different light conditions and face scales. The red line represents the iris contour, and the green cross the iris center.

a real time capture. Figure 5.11 show the detection of the irises centres in a random frame from a real time capture.

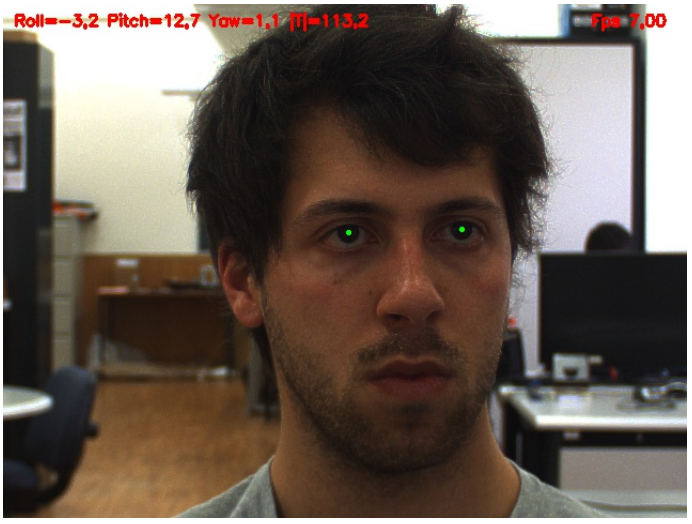


FIGURE 5.11: Detection of the iris center (green point) in a frame of a real time capture.

5.3 Gaze Estimation

The gaze direction estimation is divided in three separated parts: the offline calibration of the anatomical constants; the construction of the 3D eyeball; and the gaze estimation.

5.3.1 Training the Anatomical Constants

The proposed method to calibrate the anatomical constants only need to be executed once per person. It only consists on looking at three known points. Every time the application runs, the user is asked if he wants to make a new or use an old calibration (like figure 5.12 shows). The anatomical constants are saved in a "*.txt*" file. For this calibration step, the

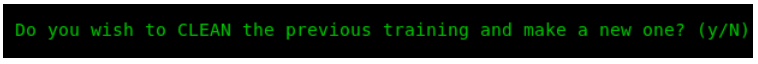


FIGURE 5.12: Terminal interface for the user to choose if he wants to do a new calibration or not.

multi-point board of figure 5.13 was built.

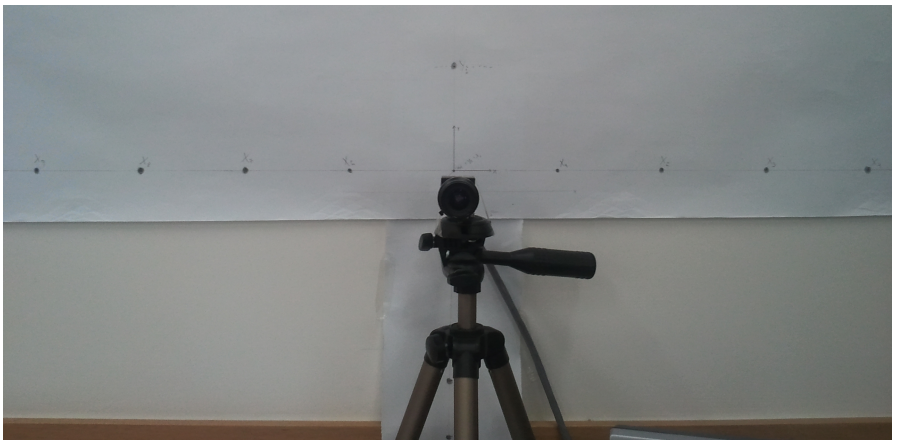
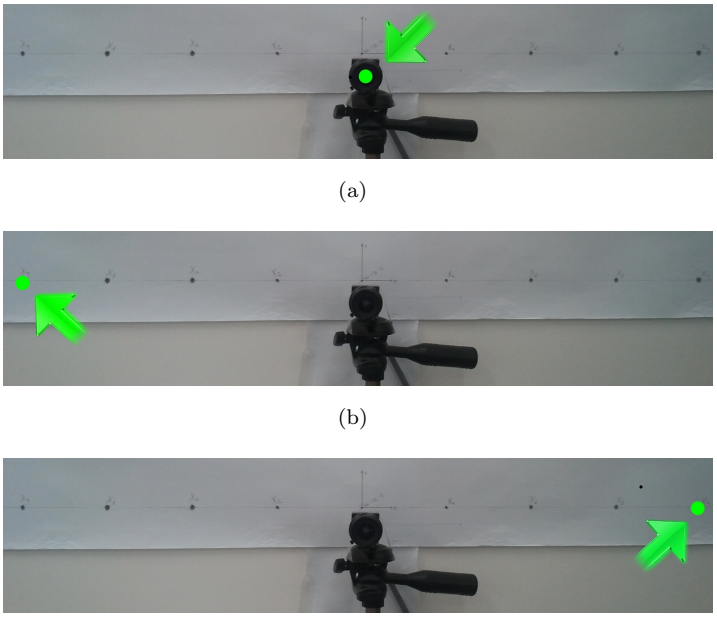


FIGURE 5.13: White board placed on the wall, parallel to the camera XY plane.

Let's consider the generic situation where the user is standing at D meters away from the board illustrated in figure 5.13:

- The first point of the calibration, the only one needed to estimate T_x and T_y, is located at the center of the camera (figure 5.14(a));
- The second calibration point is the first one for the eyeball radius estimation, and it is 40 cm to the left of the camera (like figure 5.14(b) illustrates);
- The third and last calibration point is 40 cm to the right of the camera (like figure 5.14(c) illustrates);

- The eyeball radius is then computed, for each eye and consequently the depth L for each eye also.



(a)

(b)

(c)

FIGURE 5.14: Three-point offline calibration.

5.3.2 Building the 3D Eyeball

The eyeball is defined by its center $O = (o_x, o_y)$ and its radius R_0. In figure 5.15 the eyeball center is connected to the iris center by a white line. The orientation of this line defines the eye gaze direction.

FIGURE 5.15: The eyeball centre is the red dot on the beginning of the white line.

5.3.3 Gaze Direction Results

The gaze estimation from the 3D eye model is a simple geometric issue, formulated by the equations referred in chapter 3.

The Kalman filter that was applied on the gaze direction values can be seen in appendix D, section D.3. In figure 5.16 a comparison between filtered and unfiltered gaze direction values is made for one eye.

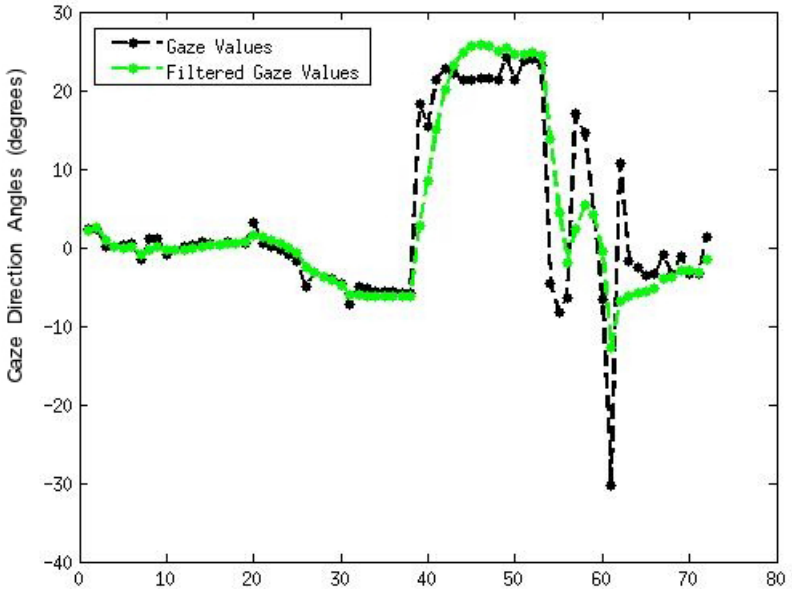


FIGURE 5.16: Filtered and unfiltered gaze values, θ_x.

To test the accuracy of the gaze estimator, a white board with known points was placed behind the camera, parallel to the image plane. The user was asked to sit a little more than one meter away from the board and look at some of those points. Two experiments were made: first, asking the user to look at some points, with small head movements and under controlled conditions (lightning, head position, clear and visible iris, etc.), in order to test the system within an optimal environment, to get the best results; second, asking the user to look at some points, without having to worry about the results, under natural head movements, in order to find the maximum error of the system.

The first results were quite impressive. As shown in figure 5.17, the user gazed 19 points and did his best to be perfectly detected by the face and iris detector. The black dots are the known points $P_i = (X, Y)$ in the white board, and around them are the N_i number of gaze samples taken, $i = 0, \ldots, 18$. These sample points represent the intersection of the gaze

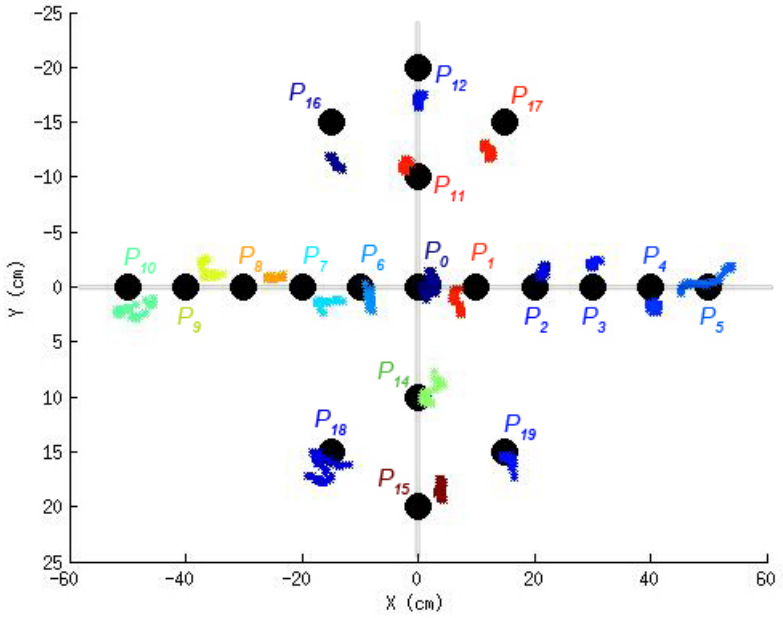


FIGURE 5.17: Gaze accuracy test, in ideal conditions.

line with the board plane. With these N_i samples per point, a mean intersection point G_i is calculated and the absolute average gaze error is computed based on G_i and on the real position of P_i. The results of this first experiment are shown in table 5.1.

In the second experiment, the user gazed at 18 points, and the gaze values were collected without caring if the the user's face and eyes were rightly detected and if the user was performing regular head poses. Also, the anatomical constants that were used in this experiment weren't very accurate, as result of a fast offline calibration. The results are shown in figure 5.18. The results of the second experiment are quantified in table 5.2.

The bigger error shown in this second experiment is mainly caused by the uncertainties in the anatomical constants. Because of being a small and fast calibration, the anatomical constants can be a bit inaccurate, resulting on a bad head movement compensation. To solve this inaccuracy and keep the fast offline calibration, an headrest should be placed directly in front of the camera, so that the user's head is still (in $Roll = Pitch = Yaw = 0$) during the 3-point training.

$\boldsymbol{P}_i$ **(cm)**	$\boldsymbol{N}_i$	$\boldsymbol{G}_i$ (cm)	**Mean Gaze Gaze** (°)	**Expected Direction** (°)	**|Mean Error|** $(\bar{\mathbf{e}}_{\theta_\mathbf{x}}, \bar{\mathbf{e}}_{\theta_\mathbf{y}})$ (°)
$P_0 = (0; 0)$	45	(2.43;-0.18)	(1.41;-0.1)	(0;0)	(1.41;0.1)
$P_1 = (10; 0)$	29	(6.91;1.54)	(3.27;0.73)	(4.73;0)	(1.46;0.73)
$P_2 = (20; 0)$	24	(21.61;-1.47)	(10.05;-0.69)	(9.32;0)	(0.73;0.69)
$P_3 = (30; 0)$	19	(30.13;-2.13)	(15.44;-1.12)	(15.38;0)	(0.07;1.12)
$P_4 = (40; 0)$	29	(40.66;1.9)	(20.95;1.02)	(20.64;0)	(0.31;1.02)
$P_5 = (50; 0)$	25	(49.67;-0.71)	(23.81;-0.36)	(23.95;0)	(0.14;0.36)
$P_6 = (-10; 0)$	24	(-8.34;1.1)	(-4.44;0.59)	(-5.32;0)	(0.88;0.59)
$P_7 = (-20; 0)$	15	(-15.71;1.41)	(-8.34;0.75)	(-10.58;0)	(2.24;0.75)
$P_8 = (-30; 0)$	14	(-24.63;-0.89)	(-12.03;-0.45)	(-14.54;0)	(2.52;0.45)
$P_9 = (-40; 0)$	17	(-36.08;-1.6)	(-19.01;-0.88)	(-20.9;0)	(1.89;0.88)
$P_{10} = (-50; 0)$	24	(-49;2.05)	(-26.46;1.19)	(-26.92;0)	(0.46;1.19)
$P_{11} = (0; -10)$	31	(-2.12;-11.2)	(-1.18;-6.2)	(0;-5.54)	(1.18;0.66)
$P_{12} = (0; -20)$	28	(0.17;-17.1)	(0.09;-9.19)	(0;-10.72)	(0.09;1.53)
$P_{13} = (0; 10)$	32	(2.14;9.61)	(1.23;5.5)	(0;5.72)	(1.23;0.22)
$P_{14} = (0; 20)$	32	(3.88;18.33)	(2.24;10.45)	(0;11.37)	(2.24;0.93)
$P_{15} = (-15; -15)$	23	(-14.08;-11.22)	(-7.71;-6.16)	(-8.2;-8.2)	(0.5;2.05)
$P_{16} = (15; -15)$	33	(12.06;-12.38)	(7.15;-7.33)	(8.87;-8.87)	(1.72;1.53)
$P_{17} = (-15; 15)$	40	(-16.29;16.25)	(-8.96;8.94)	(-8.26;8.26)	(0.7;0.68)
$P_{18} = (15; 15)$	24	(15.73;15.79)	(8.63;8.66)	(8.23;8.23)	(0.4;0.43)
				$\lvert Global\ \bar{e}\rvert \approx$	(1.03°;0.81°)

TABLE 5.1: First experiment to test the gaze estimator. All the values are rounded to two decimal places.

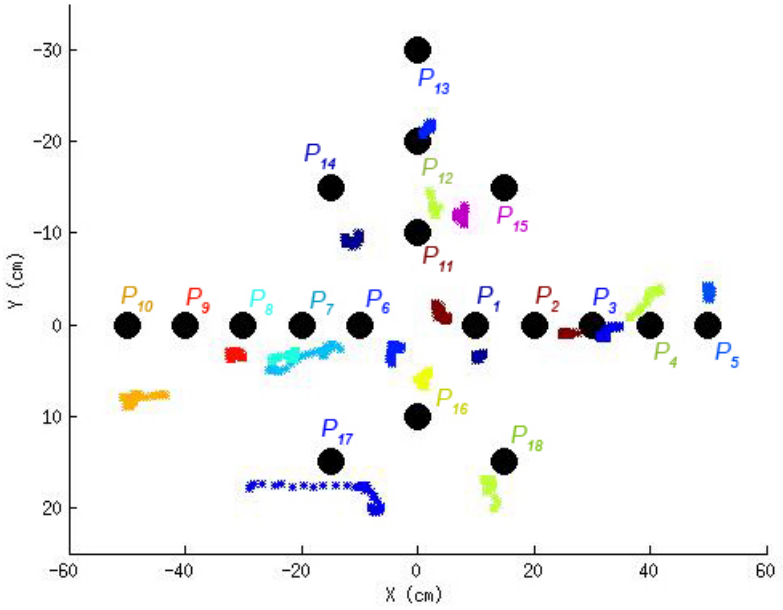


FIGURE 5.18: Fast uncontrolled test for gaze accuracy.

$\boldsymbol{P}_j$ **(cm)**	$\boldsymbol{N}_j$	$\boldsymbol{G}_j$ (cm)	**Mean Gaze Gaze** (°)	**Expected Direction** (°)	**|Mean Error|** $(\bar{\mathbf{e}}_{\theta_\mathbf{x}}, \bar{\mathbf{e}}_{\theta_\mathbf{y}})$ (°)
$P_1 = (10; 0)$	32	(10.64;3.35)	(5.67;1.79)	(5.33;0)	(0.34;1.79)
$P_2 = (20; 0)$	26	(25.65;0.95)	(13.77;0.52)	(10.82;0)	(2.95;0.52)
$P_3 = (30; 0)$	38	(32.36;0.98)	(16.88;0.53)	(15.71;0)	(1.17;0.53)
$P_4 = (40; 0)$	29	(39.15;-2.49)	(19.54;-1.29)	(19.93;0)	(0.39;1.29)
$P_5 = (50; 0)$	29	(50.13;-3.47)	(25.71;-1.91)	(25.65;0)	(0.06;1.91)
$P_6 = (-10; 0)$	33	(-4.03;2.81)	(-2.12;1.48)	(-5.25;0)	(3.13;1.48)
$P_7 = (-20; 0)$	31	(-18.98;3.55)	(-9.78;1.85)	(-10.29;0)	(0.51;1.85)
$P_8 = (-30; 0)$	47	(-22.29;3.43)	(-11.58;1.81)	(-15.42;0)	(3.84;1.81)
$P_9 = (-40; 0)$	41	(-31.36;3.23)	(-15.75;1.67)	(-19.78;0)	(4.04;1.67)
$P_{10} = (-50; 0)$	48	(-48.68;8.11)	(-23.97;4.24)	(-24.54;0)	(0.58;4.24)
$P_{11} = (0; -10)$	41	(4.11;-1.24)	(2.17;-0.66)	(0;-5.27)	(2.17;4.62)
$P_{12} = (0; -20)$	15	(2.75;-12.99)	(1.54;-7.26)	(0;-11.1)	(1.54;3.84)
$P_{13} = (0; 10)$	23	(1.98;-21.63)	(1;-10.82)	(0;-14.85)	(1;4.03)
$P_{14} = (0; 20)$	33	(0.94;5.97)	(0.49;3.14)	(0;5.24)	(0.49;2.11)
$P_{15} = (-15; -15)$	30	(-11.21;-9.2)	(-6.39;-5.25)	(-8.52;-8.52)	(2.13;3.27)
$P_{16} = (15; -15)$	26	(7.44;-11.85)	(4.33;-6.88)	(8.69;-8.69)	(4.35;1.8)
$P_{17} = (-15; 15)$	38	(-12.5;18.42)	(-7.24;10.6)	(-8.66;8.66)	(1.43;1.93)
$P_{18} = (15; 15)$	38	(12.29;17.57)	(6.76;9.63)	(8.24;8.24)	(1.48;1.39)
				$\lvert Global\ \bar{e}\rvert \approx$	(1.8°;2.22°)

TABLE 5.2: Second experiment to test the gaze estimator. All the values are rounded to two decimal places.

5.4 Eye2Eye Human Interaction Evaluation

As mentioned in chapter 4, to determine whether there is visual contact or not, a previous calibration of a stereo configuration is required. Only after that it is possible to process and intercalate information from both cameras simultaneously, and consequently evaluate if there is visual contact.

5.4.1 Calibrating the Stereo System

Two different configurations were tested. In the first configuration, the cameras were placed approximately 1.5 meters away from each other. As this configuration is not the one that was used for the final visual contact evaluation system, its calibration is explained in appendic D, section D.4.

The second configuration that was tested have the cameras side by side, attached on a metal framework, like figure 5.19.

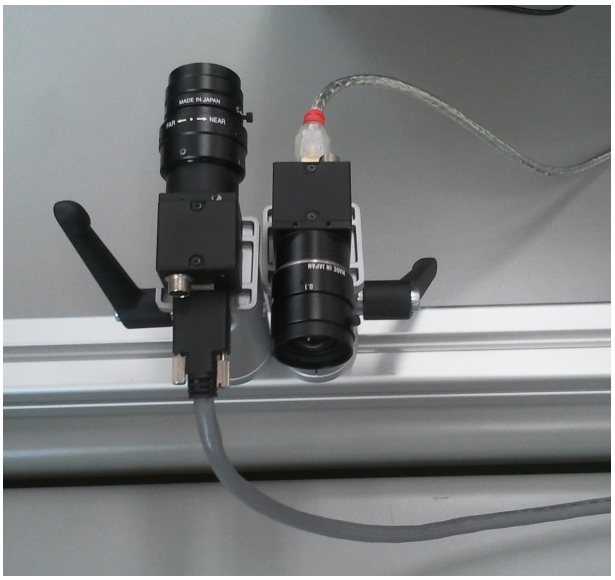


FIGURE 5.19: Disposal of the cameras in the framework.

As explained earlier on the report, this calibration method is based on mirror reflections. While one camera "sees" the chess pattern normally (figure 5.20(a)), the other camera "sees" it through a mirror (figure 5.20(b)). After applying the method described in chapter 4, and

(a) (b)

FIGURE 5.20: (a) is the image of the pattern and (b) is the mirror reflection of the pattern (a).

after using equation (4.1), the relation between camera 0 and camera 1 is:

$$ {}^{0}_{1}T \approx \begin{bmatrix} -0.9999 & 0 & -0.0021 & -3.0109 \\ 0.0004 & 0.9805 & -0.1963 & 0.7237 \\ 0.002 & -0.1963 & -0.9805 & 8.0052 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{5.9} $$

This configuration (figure 5.19) is the one that will be used in the final system to evaluate visual contact in a conversation.

5.4.2 3D Position of the Face Features

Using the anthropometric head model and the POSIT, it is possible to know the 3D position of any face feature (of the available 58 of the model) in real time. In this project, an eyelid point position from each eye is extracted to assist in the calculation of both eyes 3D position. The resolution in which these results were obtained was 800x600 pixels, therefore the features position are computed with respect to the camera center $C = (c_x, c_y)$, that is defined as being $c_x = 400$ and $c_y = 300$.

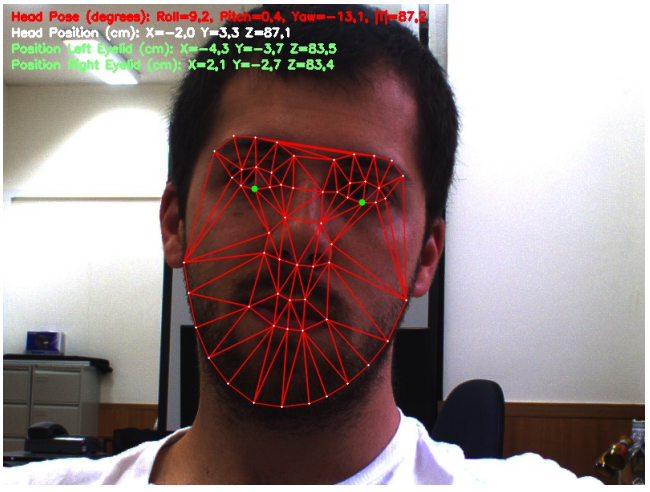
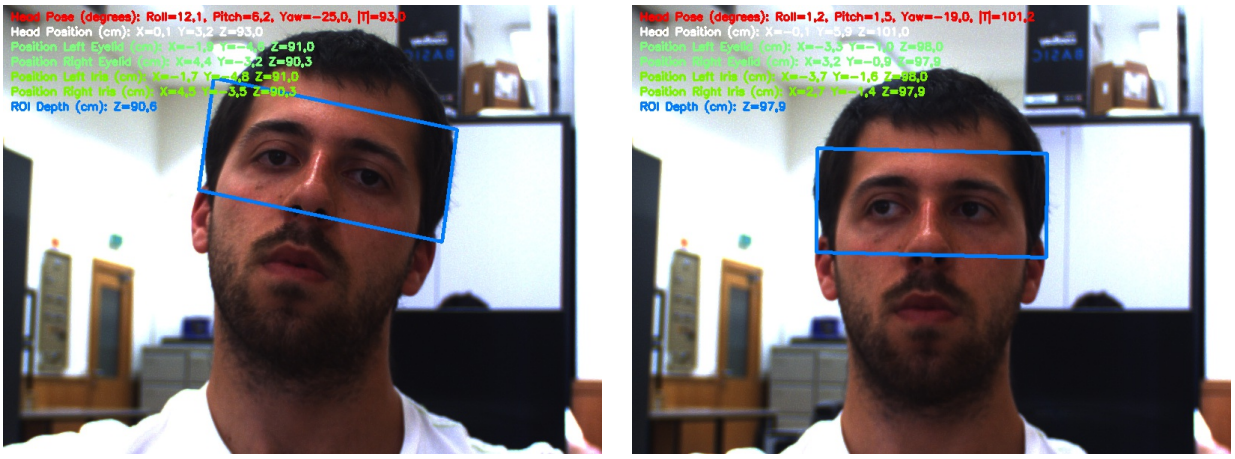


FIGURE 5.21: Example of the extraction of the 3D position of the eyelid points (green points in the face mesh).

As described in chapter 4, the eyelids position illustrated in figure 5.21 is essential to the calculation of the 3D position of the irises and the ROI of the eyes (see more examples in appendix D).



(a) (b)

FIGURE 5.22: Examples of the eyes ROI, with different head poses (blue rectangle). The Z position of the ROI is shown in the left side of the image (blue text).

The ROI of the eyes that will be used as region of gaze intersection for the visual contact evaluation, is defined by the eyelid points of figure 5.21, the *Roll* angle of the head and the face scale in the image (figure 5.22). Figure 5.23 illustrates the irises contours in several head poses and with different face scales, showing their 3D position written in the top left corner.

5.4.3 Visual Contact Classification

The gaze evaluation in the stereo configuration can be unintentionally modified, due to the calibration uncertainties. A small error in the transformation matrix between the cameras will add inconsistency in the point of focus. Basically, camera 0 needs to know where the eyes of individual 0 are, and vice-versa, and this information is achieved by using the calibration matrix ${}^{0}_{1}T$. If this transformation matrix isn't 100% correct, the projection of the eyes from one camera to the other won't be 100% correct as well. In figure 5.24 are shown some situations were the point of focus achieved high accuracy in the stereo system. Some other tests, where the point of focus wasn't very accurate, are shown in figure 5.25. The experiments from figures 5.24 and 5.25 were performed by asking a person to look at a specific mark in front of him. The pair of cameras was placed between the user and the test dummy so that one camera captures the user and the other captures the dummy. The camera that captures the test dummy knows its head position (returned by the POSIT algorithm) and projects that information to the other camera through the calibration matrix ${}^{0}_{1}T$. The

(a) (b)

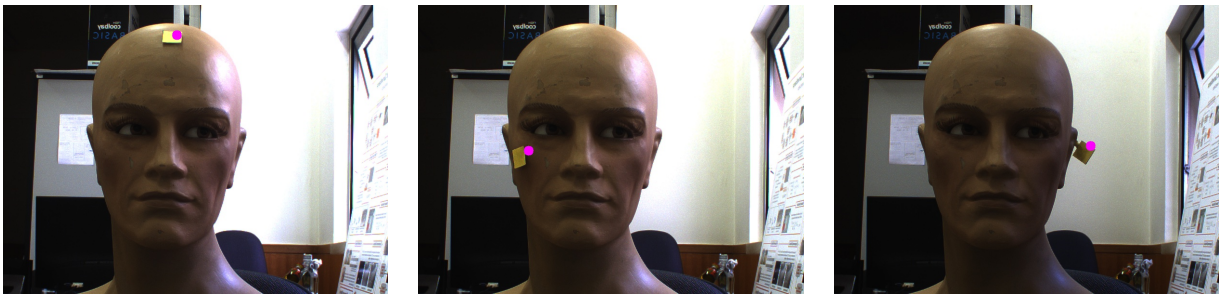
(c)

FIGURE 5.23: Three examples of the extraction of the 3D position of both irises, based on the (x, y) pixel position of the irises and the 3D Z position of the eyelids.

other camera (that captures the user) can then intersect the user's gaze with the Z-plane of the dummy's head and compute the point of focus (pink point on figures 5.24 and 5.25).

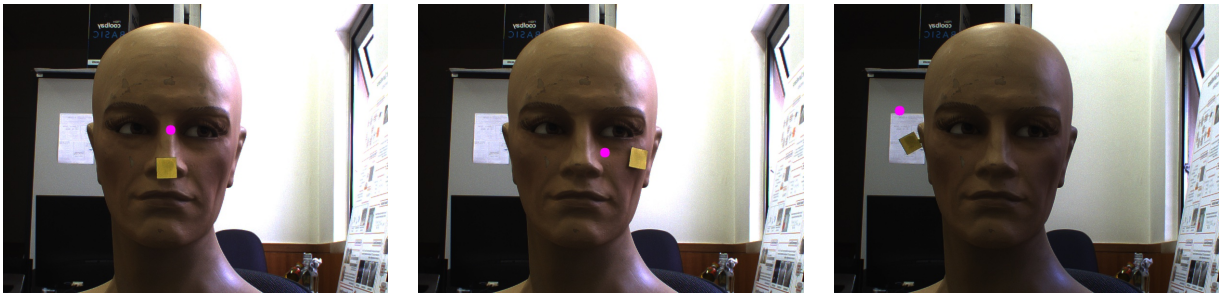
When all of the previous information is combined, the visual contact evaluation is straightforward. To signal when one individual is gazing at the other's eyes, a large green dot is drawn in the captured image of the individual that is causing the unilateral visual contact.

To test how accurate is the method when presented with a situation of eye contact, a large set of frames were collected, where one person (test user) is asked to look at the eyes region of another person (reference user). The reference user performed five poses (illustrated in figure 5.26), and for each one of them, the test user collected a set of frames where he was



(a) (b) (c)

FIGURE 5.24: The individual 0 is asked to look at the yellow square. The pink dot represents the point of focus in camera 0 coordinate system.



(a) (b) (c)

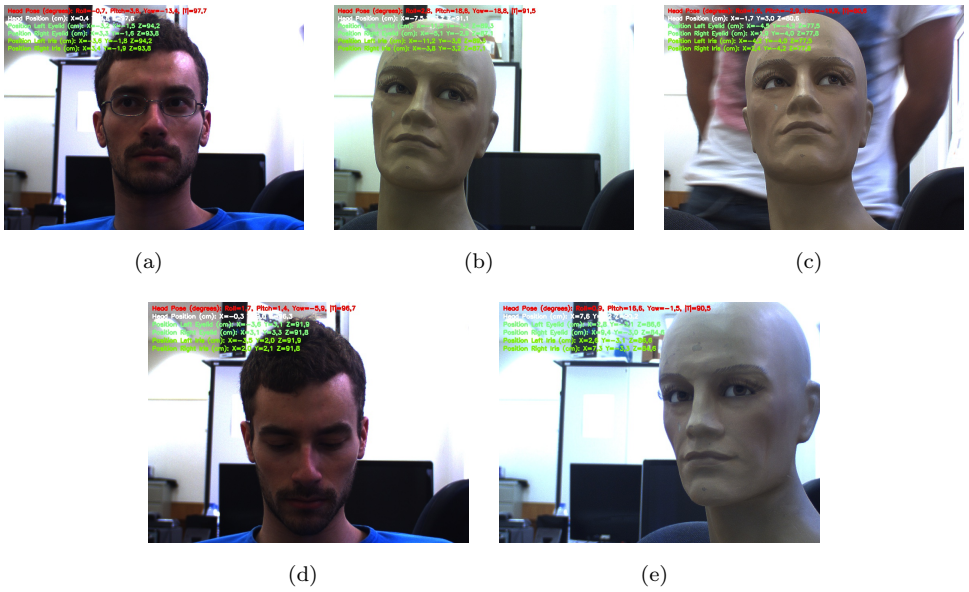
FIGURE 5.25: The same point of focus experiment as in figure 5.24, but showing some defective results.

looking at the other user's eyes while performing head rotations and translations.

Some of the head poses that the test user performed are shown in figure 5.27.

The bottom right dot in the images from figure 5.27 is is green when the algorithm detects that the test user is looking at the other user's eyes. The red dot indicates that the algorithm has failed (in that frame).

After collecting a set of 587 test frames, it is possible to conclude that the eye contact evaluation system has an approximated detection rate ("hit rate" or "success rate") of 70%. This "hit rate" takes into account unnatural head poses, and it can be increased to an optimal value of approximately 85% if both users are completely frontal and perform a conversation without extreme, unnatural and odd poses. The system can better detect situations of visual contact when both individuals are frontal to each other and in the centre of the image (this way the gaze direction has less influence from the head pose, which has angles near 0°, resulting on a smaller error on the gaze estimation).



(a) (b) (c)

(d) (e)

FIGURE 5.26: Different poses performed by the reference user for the visual contact evaluation experiment.

FIGURE 5.27: Some head poses of the test user during the visual contact experiment.

Notice that the success rate of the visual contact evaluator significantly depends on the distance between the individuals, during the conversation. For a better understanding of this "hit rate", let's imagine that two individuals are 170 centimetres away from each other. And that a regular human head has an eye region of 15 *cm* wide and 5 *cm* high. An error of 4° can cause a displacement in the point of focus of approximately 12 *cm*, which is enough

to perform a wrong evaluation of whether there is visual contact or not.

5.5 Final Analysis on the Eye Contact Evaluator

It is already known that the visual contact evaluator is able to detect up to 85% of the situations where there is, in fact, visual contact. A last experiment must be made in order to measure the final system error. In this test it is desired to know what is displacement of the point of focus relatively to a known 3D position in space.

Two users were placed approximately 1.5 meters away from of each other with the cameras framework in the middle of them. One of them (user 1) was asked to move freely, within the FOV of the camera. The other user (user 0) was asked to always look at a specific point at the face of the opposite user (white point between the eyebrows in figure 5.28). To measure the final error, a video sequence of 160 frames (approximately 30 seconds) was saved along with the point of focus of user 0 and the fixed feature/point position of user 1.

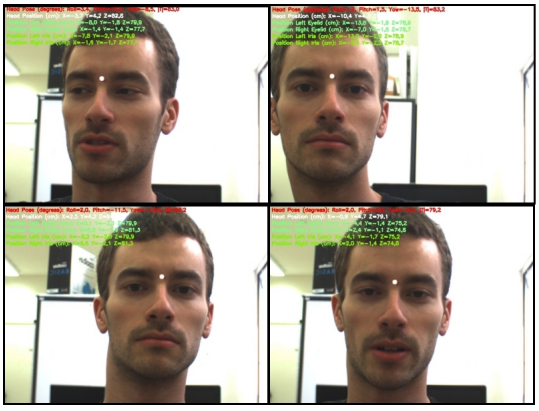


FIGURE 5.28: Some of the poses that user 1 performed during the test sequence.

The results are shown in the graphics from figures 5.29 and 5.30.

From figures 5.29 and 5.30 is possible to extract the absolute mean error and the maximum error (red vertical line in both graphs) that were obtained in the 30 seconds video sequence. In figure 5.29, the maximum error between the real point position and the point of focus is $e_{x_{max}} \approx 4.8995\ cm$, while the absolute mean error of the sequence is $|e_x| \approx 1.3157\ cm$. In figure 5.30, the maximum error is $e_{y_{max}} \approx 3.5464\ cm$ and the absolute mean error is $|e_y| \approx 1.1927\ cm$.

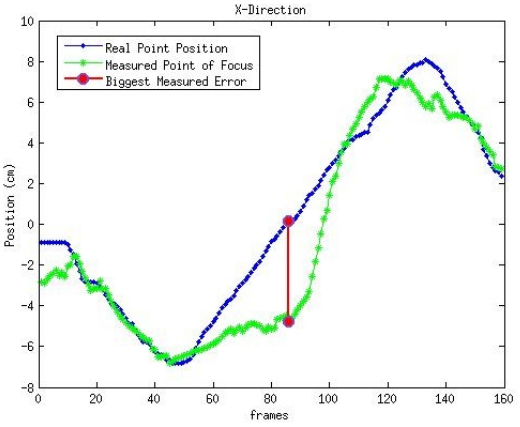


FIGURE 5.29: Measured information in the horizontal direction.

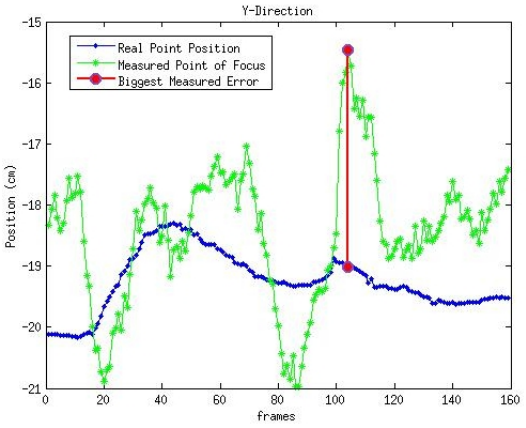


FIGURE 5.30: Measured information in the vertical direction.

It is then possible to conclude that the point of focus estimation in the stereo configuration is quite good in the X-direction, showing a low absolute mean error and very good consistency. The accuracy on the Y-direction, despite its lower mean error, is more defective, showing signs of inconsistency, with higher data variance between adjacent samples.

The error spikes that are shown in figures 5.29 and 5.30 are enough to mislead the program to perform a wrong evaluation on the existance of visual contact, originating the "hit rate" values that were reported earlier.

Notice that user 0 and user 1 didn't perform any extreme poses, in order to get the best results possible. To test the system in more rough conditions, another experiment was made. This time, user 1 was placed about 2 meters away from user 0, and was asked to perform faster movements and more extreme rotations and translations. A set of 293 frames were

collected for this experiment.

The results of this second experiment are shown in figures 5.31 and 5.32, where a bigger error was obtained in both directions.

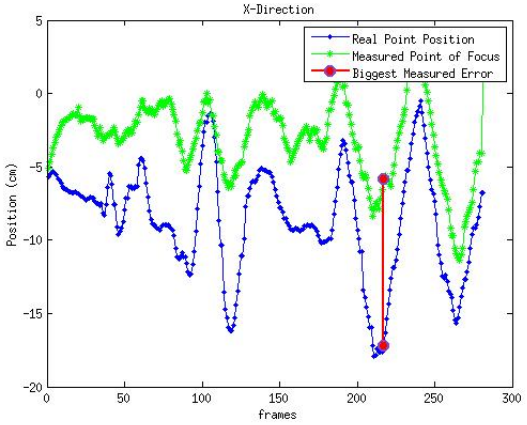


FIGURE 5.31: Measured information in the horizontal direction, on the second experiment.

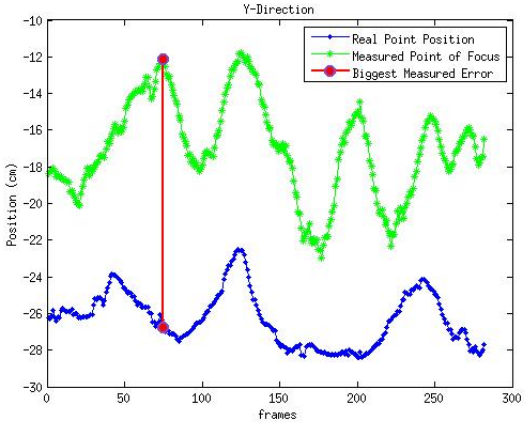


FIGURE 5.32: Measured information in the vertical direction, on the second experiment.

For figure 5.31: $e_{x_{max}} \approx 11.3134\ cm$ and $|e_x| \approx 5.586\ cm$. For figure 5.32: $e_{y_{max}} \approx 14.6128\ cm$ and $|e_y| \approx 9.2946\ cm$.

Notice that the error seems to be systematic which can be an indication of stereo calibration errors or anatomical constants inaccuracy. Besides, as user 1 was too far away from user 0, the gaze error greatly affected the point of focus.

Nevertheless, a 10 cm error in a plane that is 2 meters away from the user means an approximate gaze error of 3°, which is a small gaze estimation error.

Chapter 6

Conclusion

In this project is presented a system that, based on active appearance models, head pose and gaze estimators, is capable of detect whether there is or not visual contact in a frontal conversation between two players. Using a pair of calibrated cameras, placed in the middle of the two persons, this method will, simultaneously, track both heads, estimate their pose and compute the gaze direction for both players. The gaze estimation is based on a 3D geometric model of the eyeball, which needs an unique offline calibration (for each person), that consists of looking at three different known points. This geometric model will allow the estimation of the gaze direction under free head movements. Finally, to evaluate the visual contact, the gaze line on one person is intersected with the eyes plane of the other person, and if the intersection point lies inside the eyes region, there is visual contact.

The visual contact evaluation system is an aggregation of several independent methods. Any improvement on any of those methods will also improve the global system error. This global error can be decreased if any of the following improvements is done: force the anatomical constants calibration to be more accurate, implemented an headrest to force the user to have $Roll = Pitch = Yaw = 0$ and to use only the eyes; make the AAM fitting more flexible to face expressions and head poses, so that the eye images can always be extracted in perfection; improve the anthropometric model (or even create one for each user) so that the 3D eye model can perfectly fit in the 3D head model; having better hardware (such as cameras) can always decrease de system error, since the clearest the image is, the better the integro-differential will operate in the iris detection; improve the stereo calibration to have less error in the transformation matrix between cameras.

Concluding, in this work it is proposed a method that can estimate the gaze direction under natural head movements, with an approximate mean absolute error of $\pm 1.8^\circ$ in the X direction and $\pm 2.2^\circ$ in the Y direction. The maximum gaze error that was found during the experimental tests has never been over 5° in both directions (generally happened with severe head poses and/or extreme eye occlusions and distortions). A novel system of eye contact evaluation is introduced, using the information provided by the gaze estimation. From the experimental results it was concluded that for a regular frontal conversation between two persons, under natural head rotations and translations, this system is capable of detect visual contact in approximately 85% of cases. And for situation where the user might do some extreme or odd poses, the "hit rate" of the visual contact evaluator is about 70%. This percentage increases for smaller distances between the individuals (during the conversation).

With some additional work, this system could be implemented in a multi-person conversation, using a multi-camera configuration, in order to identify separate situations of eye contact within the conversation. A system like this can be a great advantage in areas like medicine and neurology. To study fatigue and attention disorders, the eyes movement is a major factor. A particular example is the analysis of the autism symptoms and condition evolution. One of the autism major symptoms is the lack of eye contact. Having the ability to evaluate this kind of situations automatically would be extremely helpful in the diagnosis and monitoring of people with this kind of disorder.

Appendix A

Head and Eye Tracking: Appendix

A.1 Single Camera Calibration

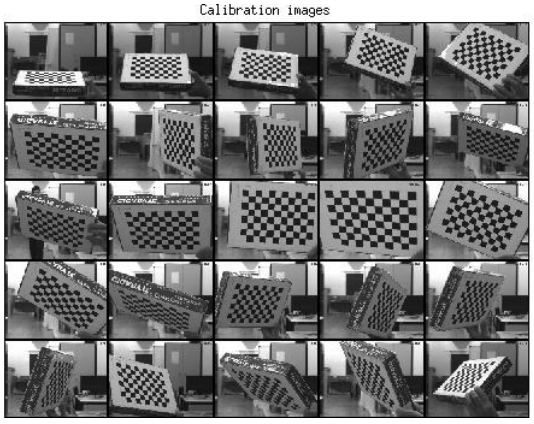


FIGURE A.1: Set of 25 images captured for the calibration.

There are several tools and methods to calibrate cameras. In this work, the "Camera Calibration Toolbox for Matlab" by Jean-Yves Bouguet [8] is the one that was used.
Lets describe the method step by step:

1. Generate and print check board pattern. As many squares as possible, although it is not good to have them too small (20mm x 20mm is good).

2. Paste the pattern in a completely flat panel.

3. Capture a reasonable number of images with the pattern in different positions.

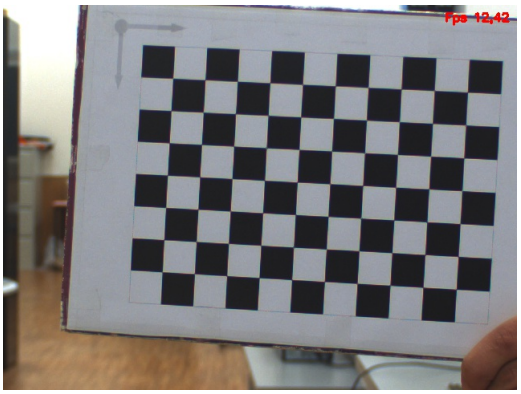


FIGURE A.2: Planar checker board pattern used to calibrate a single camera.

4. Image by image, select the grid corners (always in the same order). The toolbox has a corner extraction engine, otherwise the number of squares in the grid must be mentioned.

5. After extract the grid corners in all images, the toolbox will do the rest and return all the intrinsic parameters of the camera.

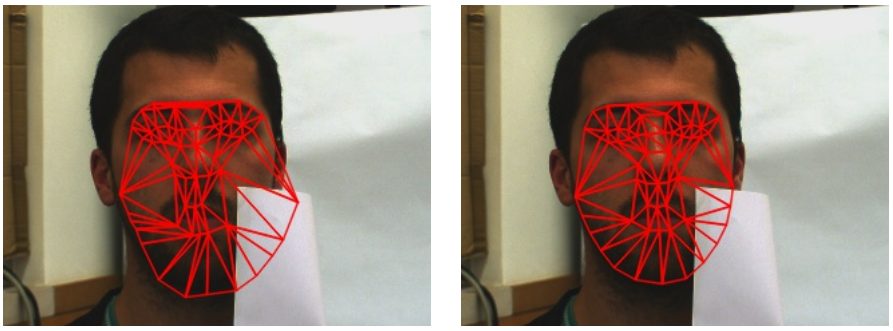
This calibration is done so that it is possible to relate world and image points. This relation is achieved by the camera matrix K:

$$K = \begin{bmatrix} f & 0 & c_x \\ 0 & f & c_y \\ 0 & 0 & 1 \end{bmatrix} \tag{A.1}$$

Where f is the focal distance of the camera (in pixel coordinates) and $C = [c_x \ c_y]$ is the camera center.

A.2 AAM Fitting

Example of a robust fitting algorithm in comparison with SIC algorithm:



(a) (b)

FIGURE A.3: AAM fitting with some occluded features, using (a) SIC algorithm and (b) Robust SIC algorithm.

Appendix B

Gaze Estimation: Appendix

B.1 The Kalman Filter

The state of the system is denoted as a vector of real numbers and it is represented by the variable x_k, where k is the time step. For example, in the particular situation where measurements of a moving point in the 2-dimensional space are being taken, the state of the point could be summarized by four variables:

$$x_k = \begin{bmatrix} x \\ y \\ v_x \\ v_y \end{bmatrix}_k \tag{B.1}$$

In this case, the goal is to filter a 2-D vector that contains the horizontal and vertical gaze of an eye. So the state vector can be represented by:

$$x_k = \begin{bmatrix} \theta_x \\ \theta_y \\ \omega_x \\ \omega_y \end{bmatrix}_k \quad , \quad \omega = angular\ velocity \tag{B.2}$$

The equation of the state at time step k is considered to be function of the state at time step $k-1$:

$$x_k = Fx_{k-1} + Bu_k + w_k \tag{B.3}$$

Here x_k is a 4-dimensional vector of state components and F, the transition matrix, is a 4-by-4 matrix. The vector u_k is there to allow external controls on the system at the moment k. As an external control is not wanted or needed, the equation (B.3) can be simplified to:

$$x_k = Fx_{k-1} + w_k \tag{B.4}$$

The transition matrix, like equations (B.1) and (B.2) suggest, will take the following form:

$$F = \begin{bmatrix} 1 & 0 & \delta t & 0 \\ 0 & 1 & 0 & \delta t \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{B.5}$$

This way the state variable x_k will depend on all components from the state variable x_{k-1}.

The variable w_k is a random variable (usually called the process noise) associated with random events or forces that directly affect the actual state of the system. It is assumed that the components of the process noise have Gaussian distribution for some covariance matrix $Q_k = E[w_k w_k^T]$ that does not vary with time.

$$w_k \sim N(0, Q_k) \tag{B.6}$$

In general, measurements z_k may or may not be direct measurements of the state variable x_k. So the m-dimensional vector of measurements z_k is given by:

$$z_k = H_k x_k + v_k \tag{B.7}$$

Here H_k is an m-by-n matrix and v_k is the measurement error, which is assumed to be zero mean Gaussian white noise with covariance $R_k = E[v_k v_k^T]$.

$$v_k \sim N(0, R_k) \tag{B.8}$$

Also like Q_k, H_k and R_k do not vary in time, so:

$$Q = Q_k\,; \quad R = R_k\,; \quad H = H_k \tag{B.9}$$

However, as it is only desired to measure the gaze values θ_x and θ_y, there is only need to consider these two variables for equation (B.7), so the vector of measurements will be:

$$z_k = \begin{bmatrix} z_x \\ z_y \end{bmatrix}_k \tag{B.10}$$

And H will be something like:

$$H = \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \tag{B.11}$$

All these expressions can now be used in the Kalman filter process. The first thing to do, as already said, is to compute the a priori estimate x_k^-. This value is given by:

$$x_k^- = F x_{k-1} \tag{B.12}$$

Using P_k^- to denote the error covariance, the a priori estimate for this covariance at time step k is obtained by:

$$P_k^- = F P_{k-1} F^T + Q_{k-1} \tag{B.13}$$

The prediction step tell us "what we expect" based on "what we have already seen". The update part starts on this point by computing the Kalman gain K, which tells us how to weight new information against what we think we already know:

$$K_k = P_k^- H^T (H P_k^- H^T + R)^{-1} \tag{B.14}$$

The gain K of equation (B.14) allows to optimally compute the updated values for x_k and P_k when a new measurement is available:

$$x_k = x_k^- + K_k(z_k - H x_k^-) \tag{B.15}$$

$$P_k = (I - K_k H)P_k^- \tag{B.16}$$

On the state variable x_k is where the filtered gaze direction will be, $\theta_{x_{filtered}}$ and $\theta_{y_{filtered}}$.

Appendix C

Visual Contact Evaluation: Appendix

C.1 Stereo Calibration

If it is known that the position of the cameras are fixed and if it is possible to compute the position of a point P_0 in the reference camera, and a point P_1 in the other camera, then it is also possible to relate both positions, being only necessary to have the orientation and translation (R, T), of the second camera relative to the main one. The relation between P_0 and P_1 is given by:

$$P_0 = RP_1 + T \tag{C.1}$$

From equation (C.1) it is true that

$$P_0 \cdot (T \times P_0) = P_0 \cdot (T \times RP_1) = 0 \tag{C.2}$$

The corresponding image coordinates of the points $P_0 = (X_0, Y_0, Z_0)$ and $P_1 = (X_1, Y_1, Z_1)$ are

$$p_0 = f_0 \frac{P_0}{Z_0} \; ; \; p_1 = f_1 \frac{P_1}{Z_1} \tag{C.3}$$

The equation (C.3) can be used to rewrite equation (C.2)

$$p_0 \cdot (T \times Rp_1) = 0 \tag{C.4}$$

Replacing the cross product on equation C.4 by a matrix multiplication we get

$$p_0 \cdot [T_\times] R p_1 = 0 \tag{C.5}$$

Knowing that

$$T = [T_0, T_1, T_2]^T \tag{C.6}$$

Then, the matrix $[T_\times]$ is given by

$$[T_\times] = \begin{bmatrix} 0 & -T_3 & T_2 \\ T_3 & 0 & -T_1 \\ -T_2 & T_1 & 0 \end{bmatrix} \tag{C.7}$$

Now, to simplify equation (C.5) it is possible to define the essential matrix E by setting

$$E = [T_\times] R \tag{C.8}$$

And equation (C.5) becomes

$$p_0^T E p_1 = 0 \tag{C.9}$$

Thanks to the stereo pair calibration it is possible to correspond image points between cameras, given the rotation matrix and translation vector.

C.1.1 Frontal Configuration

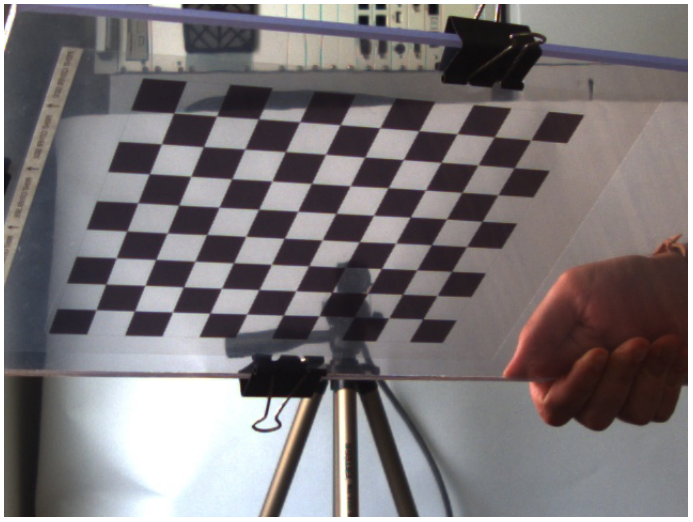


FIGURE C.1: Special chess pattern for the stereo calibration already printed on the acetate sheet and placed between the acrylic plates.

C.2 Pixel Coordinates to World Coordinates

Based on the model of a pinhole camera ([8], [12], [13] and [14]), the projection of a 3D point described in homogeneous coordinates as $P_w = \begin{bmatrix} X_w & Y_w & Z_w & 1 \end{bmatrix}^T$ is given by:

$$\lambda \cdot p_c = M P_w \tag{C.10}$$

where λ is a scale factor ([9]), p_c is the respective 2D point of P_w in homogeneous coordinates: $p_c = \begin{bmatrix} x_c & y_c & 1 \end{bmatrix}^T$ and M is the 3x4 projection matrix that represents a map from 3D to 2D.

$$M = \begin{bmatrix} m_{11} & m_{12} & m_{13} & m_{14} \\ m_{21} & m_{22} & m_{23} & m_{24} \\ m_{31} & m_{32} & m_{33} & m_{34} \end{bmatrix} \tag{C.11}$$

This projection matrix is obtained by setting:

$$M = K[R|t] \tag{C.12}$$

Where K is the intrinsic parameters matrix (or camera matrix) returned by the single camera calibration discussed in chapter 2. The rotation matrix R and translation vector t relate the world and camera reference frames. In this case, we consider the camera coordinate system as being the world coordinate system. This will cause the rotation matrix and translation vector to be like:

$$R = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{C.13}$$

and

$$t = \begin{bmatrix} 0 & 0 & 0 \end{bmatrix}^T \tag{C.14}$$

Using expressions (C.12), (C.13) and (C.14), it is possible to rewrite equation (C.10) to be like the following:

$$\begin{bmatrix} \lambda x_c \\ \lambda y_c \\ \lambda \end{bmatrix} = \begin{bmatrix} f & 0 & c_x & 0 \\ 0 & f & c_y & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix} \tag{C.15}$$

From equation (C.15) it is obvious that $\lambda = Z_w$.

Appendix D

Experimental Results: Appendix

D.1 AAM Results

D.1.1 Creation of the Shape Model

Example of a good training set for the AAM shape model illustrated in figure D.1.

FIGURE D.1: Example of a good training set for the shape model.

D.1.2 AAM Fitting Results

With a person that is not included in the training model, the AAM fitting will fail (see figure D.2):

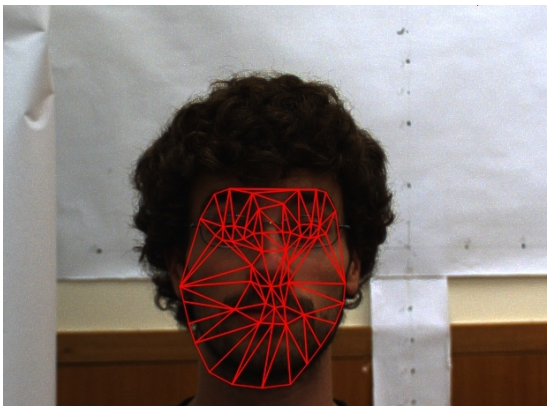
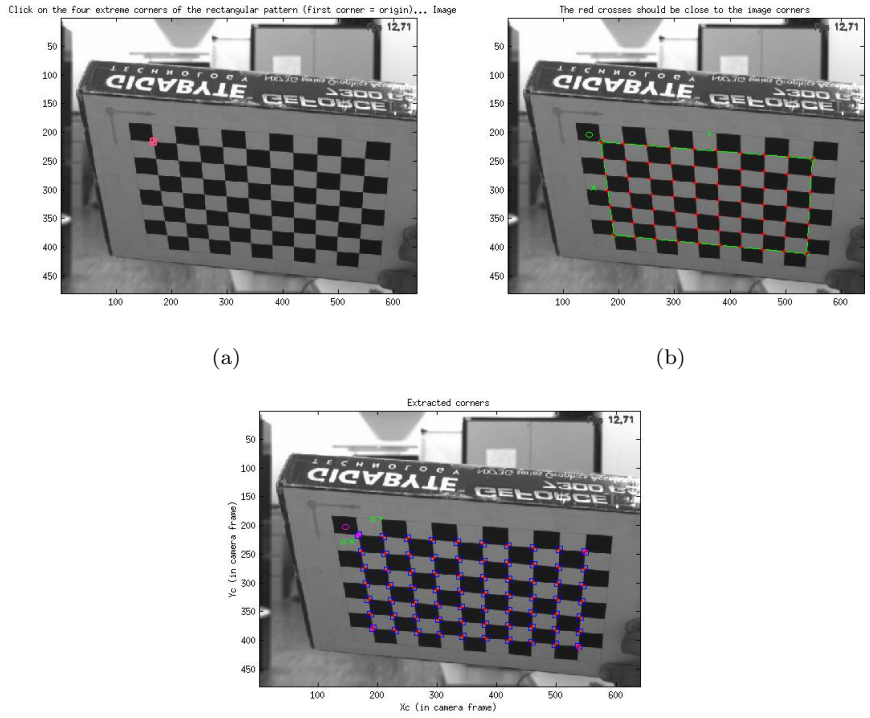


FIGURE D.2: Failure example of the AAM.

D.2 Camera Calibration Results

The corner extraction process for the single camera calibration and the respective reprojection error are shown in figure D.3(c) and D.4, respectively.



(a) (b)

(c)

FIGURE D.3: Corner extraction of one image.(a) First click on each image is the origin and must be consistent with all the other images.(b) Result of the corner detection, with the red crosses over the pattern corners. (c) Extracted corners with the blue squares around the corner points showing the limits of the corner finder window.

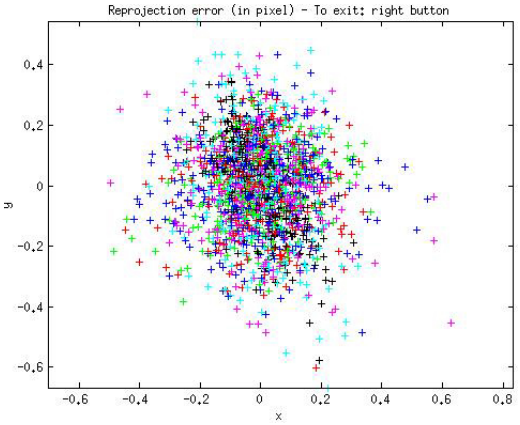


FIGURE D.4: Reprojection error (in pixel coordinates).

D.3 Kalman Filter Initialization

The gain factor K_k grows if the measurement covariance R becomes smaller, thus putting more weight on the measurement residual (difference between predicted an actual measurement). Also if the noise covariance P_k^- becomes smaller, less emphasis is put on the measurement residual. Therefore, to avoid having a slow response to the gaze output and at the same time to be able to filter gaze spikes (due to occasional iris detection failures), the Kalman measurement matrix, process and measurement covariance matrices are initialized with the following values:

$$H = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} \tag{D.1}$$

$$Q = \begin{bmatrix} 0.0001 & 0 & 0 & 0 \\ 0 & 0.0001 & 0 & 0 \\ 0 & 0 & 0.0001 & 0 \\ 0 & 0 & 0 & 0.0001 \end{bmatrix} \tag{D.2}$$

$$R = \begin{bmatrix} 0.01 & 0 & 0 & 0 \\ 0 & 0.01 & 0 & 0 \\ 0 & 0 & 0.01 & 0 \\ 0 & 0 & 0 & 0.01 \end{bmatrix} \tag{D.3}$$

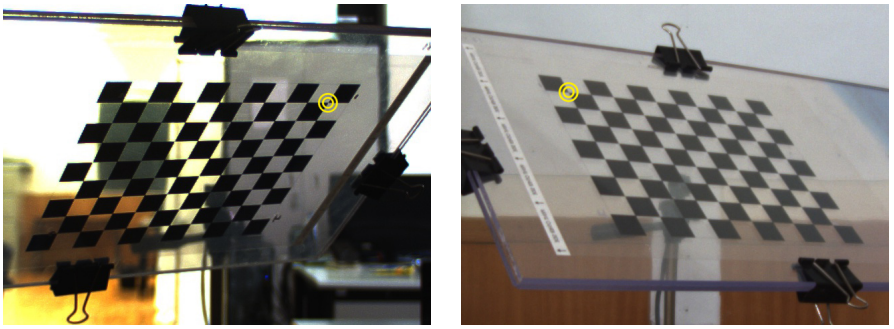
The noise covariance is initialized with small values, assuming that there isn't much noise in the environment: $P_0 = 0.5$.

D.4 Calibration of the Frontal Stereo Configuration

After collecting 20 images, per camera (in the configuration of figure D.5), of the transparent pattern related in chapter 4, figure C.1, the stereo calibration was performed, using the Bouguet's toolbox. Notice that a small change must be done so the regular stereo calibration works, because in every pair of collected images, the grid origin must be the same, in both cameras. To illustrate this situation, let's consider the pair of images shown in figure D.6.

This means that, if for one camera the grid Z-axis points upward, then for the other camera it must point downward. And that is the small modification required for this calibration to

FIGURE D.5: First stereo configuration that was calibrated.

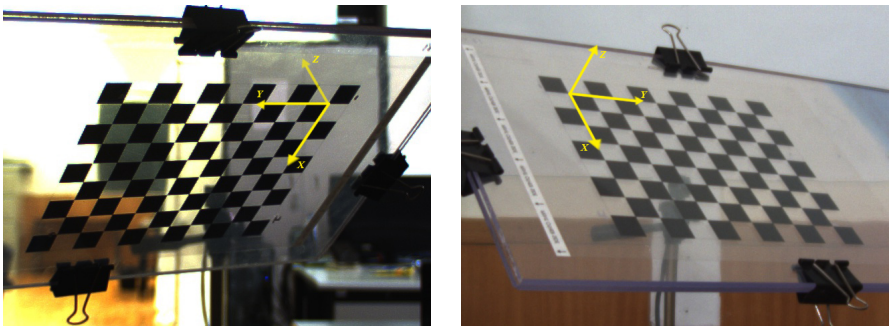


(a) (b)

FIGURE D.6: Example of an image of the pattern collected at the same time by both cameras in the frontal configuration; (a) was collected by Camera 0 and (b) was collected by Camera 1. The grid origin is represented by the yellow mark.

work properly.

In figure D.7 is represented the axis modification.



(a) (b)

FIGURE D.7: Grid coordinate system seen by both cameras; The Z-axis in figure (a) is pointing downward and in figure (b) is pointing upward.

The stereo calibration returned the following transformation matrix:

$$ {}^{0}_{1}T \approx \begin{bmatrix} -0.9994 & -0.0008 & 0.0340 & -0.4508 \\ -0.0012 & 0.9999 & -0.0134 & 1.4717 \\ -0.0340 & -0.0135 & -0.9993 & 153.3355 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{D.4} $$

Notice that the translation values are in centimetres. The intrinsic parameters for each camera remain the same. The extrinsic parameters of the stereo configuration can be better understood by looking at the 3D representation of figure D.8.

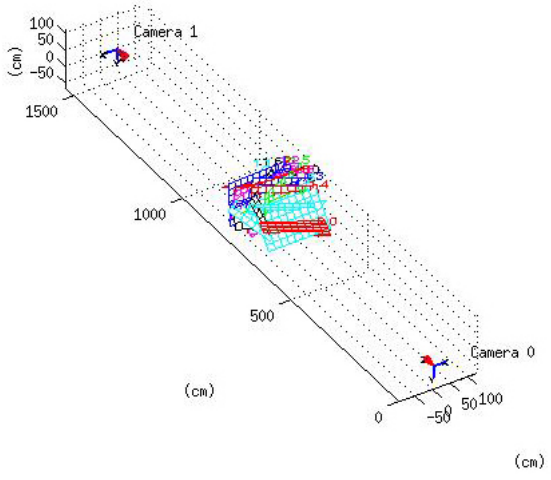
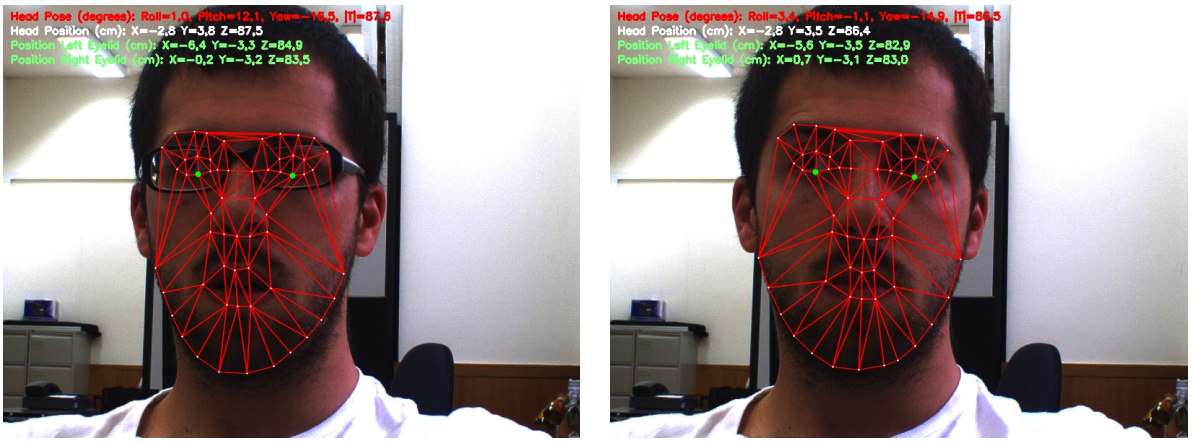


FIGURE D.8: Extrinsic parameters of the first stereo calibration.

D.5 Extraction of Features Position

Two eyelid points are extracted in every frame of the capture (see figure D.9).



(a) (b)

FIGURE D.9: More examples of the extraction of the 3D position of the eyelid points (green points in the face mesh).

Bibliography

[1] John Daugman. How Iris Recognition Works. *IEEE Transactions on Circuits and Systems for Video Technology,* vol. 14, no. 1, January 2004.

[2] Libor Masek. Recognition of Human Iris Patterns for Biometric Identification. *Bachelor report*, 2003.

[3] Takahiro Ishikawa, Simon Baker, Iain Matthews and Takeo Kanade. Passive Driver Gaze Tracking with Active Appearance Models. *Proceedings of the 11th World Congress on Intelligent Transportation Systems,* October, 2004.

[4] E. S. Perkins, B. Hammond and A. B. Milliken. Simple method of determining the axial length of the eye. *British Journal of Ophthalmology,* 1976 6o, 266.

[5] Paul Ivan. Active Appearance Models for Gaze Estimation. *Masters Thesis ,* August 21, 2007.

[6] Pedro Martins, Jorge Batista. Accurate Single View Model-Based Head Pose Estimation. *IEEE International Conference on Automatic Face and Gesture Recognition*, 2008.

[7] Tomas Svoboda, Daniel Martinec, and Tomas Pajdla. A Convenient Multi-Camera Self-Calibration for Virtual Environments. *Teleoperators and Virtual Environments,* pp 407-422, 14(4), August 2005.

[8] Jean-Yves Bouguet. Camera Calibration Toolbox for Matlab. URL: http://www.vision.caltech.edu/bouguetj/calib_doc/.

[9] OpenCV Documentation: Camera Calibration and 3D Reconstruction. URL: http://opencv.willowgarage.com/documentation/camera_calibration_and_3d_reconstruction.html

[10] Lindsay Kleeman. Understanding and Applying Kalman Filtering. *Department of Electrical and Computer Systems Engineering, Monash University, Clayton.*

[11] OpenCV Documentation: Motion Analysis and Object Tracking. URL: http://opencv.willowgarage.com/documentation/motion_analysis_and_object_tracking.html

[12] Gary Bradski and Adrian Kaehler. Learning OpenCV, September 2008.

[13] Michael Langer. Fundamentals of Computer Vision Lectures. *McGill University*, Sept. 20, 2010.

[14] Paulo Menezes. Camera calibration Lecture. *DEEC, University of Coimbra*, 2011.

[15] Pedro Martins. Active Appearance Models for Facial Expression Recognition and Monocular Head Pose Estimation. *MSc. Thesis, Faculty of Sciences and Technology - University of Coimbra*, 2008.

[16] Pedro Martins, Jorge Batista. Identity and Expression Recognition on Low Dimensional Manifolds. *IEEE International Conference on Image Processing*, 2009.

[17] Iain Matthews, Simon Baker. Active Appearance Models Revisited. *The Robotics Institute, Carnegie Mellon University* , 2004.

[18] M.Kirby and L.Sirovich. Application of the Karhunen-Lokve Procedure for the Characterization of Human Faces. *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 12, no. I , January 1990.

[19] Daniel F. DeMenthon and Larry S. Davis. Model-based object pose in 25 lines of code. *International Journal of Computer Vision ,* 1995.

[20] Eui Chul Lee and Kang Ryoung Park. A robust eye gaze tracking method based on a virtual eyeball model. *Machine Vision and Applications,* 20, 5 (June 2009), 319-337.

[21] Z. Zhu and Q. Ji. Novel eye gaze tracking techniques under natural head movement. *19th International Conference on Pattern Recognition (ICPR 2008),* December 8-11, 2008.

[22] C. H. Morimoto and M. R. Mimica. Eye gaze tracking techniques for interactive applications. *Computer Vision and Image Understanding, Special Issue on Eye Detection and Tracking,* 98:424, 2005.

[23] E. D. Guestrin and M. Eizenman. General theory of remote gaze estimation using the pupil center and corneal reflections. *IEEE Transactions on Biomedical Engineering,* 53:11241133, 2006.

[24] C. Colombo, D. Comanducci, and A. Del Bimbo. Robust tracking and remapping of eye appearance with passive computer vision. *ACM Transactions on Multimedia Computing, Communications, and Applications,* vol. 3, pp. 1-20, December 2007.

[25] J. Zhu and J. Yang. Subpixel Eye Gaze Tracking. *International Conference on Automatic Face and Gesture Recognition,* pp. 124-129, May 2002.

[26] Jing Xie and Xueyin Lin. Gaze Direction Estimation Based on Natural Head Movements. *Proceedings of the Fourth International Conference on Image and Graphics,* pp. 672-677, 2007.

[27] George Bebis and Kikuo Fujimura. An Eigenspace Approach to Eye-Gaze Estimation. *ISCA 13th International Conference on Parallel and Distributed Computing Systems,* pp. 604–609, 2000.

[28] A.Zelinsky and J.Heinzmann. Real-time Visual Recognition of Facial Gestures for Human Computer Interaction. *In Proc. of the Int. Conf. on Automatic Face and Gesture Recognition,* pp. 351–356, 1996.

[29] Stan Birchfield. Elliptical Head Tracking Using Intensity Gradients and Color Histograms. *IEEE Conference on Computer Vision and Pattern Recognition,* June 1998.

[30] Kentaro Toyama. Look, Ma No Hands! Hands-Free Cursor Control with Real-Time 3D Face Tracking, January 1998.

[31] Matsumoto, Yoshio and Zelinsky, Alexander. An Algorithm for Real-Time Stereo Vision Implementation of Head Pose and Gaze Direction Measurement. *Proceedings of the Fourth IEEE International Conference on Automatic Face and Gesture Recognition 2000,* 2000.

[32] Jochen Heinzmann and Er Zelinsky. 3-D Facial Pose and Gaze Point Estimation using a Robust Real-Time Tracking Paradigm. *In Proceedings of the IEEE International Conference on Automatic Face and Gesture Recognition,* pp. 142–147, 1998.

[33] Valenti, Roberto and Lablack, Adel and Sebe, Nicu and Djeraba, Chabane and Gevers, Theo. Visual Gaze Estimation by Joint Head and Eye Information. *Proceedings of the 2010 20th International Conference on Pattern Recognition,* pp. 3870–3873, 2010.

[34] Roberto Valenti and Theo Gevers. Accurate Eye Center Location and Tracking Using Isophote Curvature. *IEEE Conference on Computer Vision and Pattern Recognition,* 2008.

[35] Baluja, Shumeet and Pomerleau, Dean. Non-Intrusive Gaze Tracking Using Artificial Neural Networks. *In Proceedings of Advances in Neural Information Processing Systems,* vol. 6, pp 753760, 1994.

[36] Tan, Kar-Han and Kriegman, David J. and Ahuja, Narendra. Appearance-based Eye Gaze Estimation. *Proceedings of the Sixth IEEE Workshop on Applications of Computer Vision,* 2002.

[37] L. Q Xu, D. Machin, and P. Sheppard. A novel approach to real-time non-intrusive gaze finding. *In Proceedings of British Machine Vision Conference (BMVC 1998),* pp 428437, 1998.

[38] Feng Lu, Takahiro Okabe, Yusuke Sugano, Yoichi Sato. A Head Pose-free Approach for Appearance-based Gaze Estimation. *Proceedings of the British Machine Vision Conference,* pp 126.1–126.11, 2011.

[39] Sugano, Yusuke and Matsushita, Yasuyuki and Sato, Yoichi and Koike, Hideki. An Incremental Learning Method for Unconstrained Gaze Estimation. *ECCV 2008,* pp 656-667, 2008.

[40] Reale, M. and Hung, T. and Lijun Yin. Viewing direction estimation based on 3D eyeball construction for HRI. *Computer Vision and Pattern Recognition Workshops (CVPRW), 2010 IEEE Computer Society Conference,* pp 24-31, June 2010.

[41] Yamazoe, Hirotake and Utsumi, Akira and Yonezawa, Tomoko and Abe, Shinji. Remote gaze estimation with a single camera based on facial-feature tracking without special calibration actions. *Proceedings of the 2008 symposium on Eye Tracking Research and Applications,* pp 245–250, 2008.

[42] Chen, Jixu and Ji, Qiang. 3D gaze estimation with a single camera without IR illumination. *ICPR,* pp 1–4, 2008.

[43] Rui Rodrigues and João P. Barreto and Urbano Nunes. Camera Pose Estimation Using Images of Planar Mirror Reflections. *ECCV (4),* pp 382-395, 2010.