

Illumination Independent Head Pose and Pupil Center Estimation for Gaze Computation

**A Thesis
Submitted to the
Temple University Graduate Board**

**In Partial Fulfillment
Of the requirements for the Degree
Masters of Science in Electrical Engineering**

**By
Ralph Oyini Mbouna
January 2012**

**Dr. Seong G. Kong
Thesis Advisor**

**Dr. Dennis Silage
Committee Member**

**Dr. Joseph Picone
Committee Member**

**Dr. Saroj Biswas
Committee Member**

ABSTRACT

This thesis presents an economical, non-invasive and user friendly approach of determining the subject's gaze on a computer screen. Along with facial feature tracking, gaze estimation allows a user to control a computer by eye and head movements. Gaze estimation has many applications in surveillance, video gaming, advertisements, monitoring driver alertness, and medical applications. The objective of gaze estimation is to determine where the user is looking at on the computer screen. However, current gaze estimation strategies tend to be expensive, impractical, and invasive. The suggested method in this paper is able to track the user's eye and head movements in real time under various lighting conditions with a regular webcam. The head pose is tracked by matching the user's detected facial features to a 3D head model that enables the system to estimate the user's overall eye gaze. Experimental results show robustness and speed in determining the eye gaze in real time for users of various ethnicities, under different lightening conditions, at a normal distance from the webcam, with reasonable head movements and with standard resolution images. The proposed eye pupil center detection method outperforms other algorithms by up to 1.2%. The head pose tracking has a 3.9°, 5.1° and 5.3° error for tilting, shaking and nodding respectively. Both eye pupil center detection and head pose tracking have been tested with widely used public databases. In addition, despite using only a regular webcam, our overall gaze estimation system has an average gaze error of 2.9° horizontally and 3.7° vertically.

TABLE OF CONTENTS

ABSTRACT.....	I
LIST OF FIGURES.....	IV
LIST OF TABLES.....	VI
CHAPTER 1	1
INTRODUCTION TO GAZE ESTIMATION	1
1.1 Research Objective	1
1.2 Contributions	2
CHAPTER 2	5
BACKGROUND STUDY	5
2.1 Commercial and Invasive Gaze Trackers.....	5
2.2 Non-Invasive Gaze Trackers.....	6
2.2.1 IR Gaze Trackers.....	6
2.2.2 IR Free Gaze Trackers.....	7
CHAPTER 3	9
HEAD POSE ESTIMATION	9
3.1 Viola-Jones Face Detector.....	10
3.1.1 Detection of Face	10
3.1.2 Optical Flow	11
3.2 Mapping of Features to 3D Head Model	16
3.3 POSIT Algorithm.....	17
3.4 Head Pose with Euler Angles	21
3.5 POSIT Algorithm Tracking Failure and Correction	23
CHAPTER 4	27
EYE PUPIL CENTER DETECTION	27

4.1	Eye Detection	27
4.2	Pre-Emphasis: Skin Removal	28
4.3	Pupil Center Detection	30
4.4	2D Eye Model	33
CHAPTER 5		35
GAZE POINT COMPUTATION.....		35
5.1	Fusion of Head Pose and 2D Eye Model	35
5.2	Calibration.....	36
CHAPTER 6		38
EXPERIMENTAL RESULTS.....		38
6.1	Head Pose Tracking Results	38
6.2	Pupil Detection Results	44
6.3	Gaze Estimation Results.....	48
CHAPTER 7		53
CONCLUSION.....		53
CHAPTER 8		55
FUTURE WORK		55
REFERENCES.....		56

LIST OF FIGURES

Figure 1: Overall System Block Diagram	4
Figure 2: Head Pose Block Diagram	9
Figure 3: Face Detection using Viola-Jones.....	11
Figure 4: Optical Flow Sample Image.....	12
Figure 5: Optical Flow Assumptions.....	12
Figure 6: Aperture problem	14
Figure 7: Facial Features Extraction	15
Figure 8: 3D Head Model created in Blender	16
Figure 9: Mapping of 2D image features to 3D model	17
Figure 10: Relationship between a 3D point $(X, Y, Z)^T$ and its corresponding 2D projection $(u, v)^T$ onto the image plane	18
Figure 11: POSIT Algorithm Inputs/Outputs	19
Figure 12: Perspective projection (mi) and scaled orthographic projection (pi) for an object point Mi and a reference point $M0$	19
Figure 13: Head Pose with Euler Angles	23
Figure 14: 3D Head Model next to the face image in different poses.....	25
Figure 15: Head pose at different distances d	26
Figure 16: Pupil Center Detection Block Diagram.....	27
Figure 17: Eye detection from face detection	28
Figure 18: HSV Color space	29
Figure 19: HSV to RGB intensity (0-255) component comparison.....	30
Figure 20: Pupil Center Detection.....	32

Figure 21: Eye Blink.....	33
Figure 22: 2D Eye Model.....	34
Figure 23: System Setup.....	35
Figure 24: Gaze Point Estimation.....	36
Figure 25: Calibration Markers.....	37
Figure 26: Uniform lighting.....	39
Figure 27: Varying lighting.....	40
Figure 30: Euclidean distance errors.....	43
Figure 31: Eye Processing Results.....	45
Figure 32: BioID sample images.....	45
Figure 33: Real Time Detection Results for Different Subjects.....	46
Figure 34: Pupil Center Detection Experiment.....	47
Figure 35: Test Markers.....	48
Figure 36: Gaze Estimation Stages.....	49
Figure 37: Gaze estimation errors at different distances away from the camera.....	50

LIST OF TABLES

Table 1. Head Pose Tracking MAE Results	41
Table 2. Head Pose Tracking MAE of Different Algorithms	44
Table 3. Pupil Center Detection Rate.....	48
Table 4. Gaze Estimation Error (Average).....	51
Table 5. Gaze Estimation Current State of Art.....	52

CHAPTER 1

INTRODUCTION TO GAZE ESTIMATION

Eyes allow us to see and gather information about the environment. Eyes mainly act as an input organ as they collect light, but they also can be considered an output organ as they indicate the subject's gaze direction. Using the orientation of the head and the position of the eyes, it is possible to estimate the gaze path of an individual. Gaze estimation is a fast growing technology that track a person's eyes and head movements to "pin point" where the subject is looking at on a computer screen. The gaze direction is described as a person's line of sight. The gaze point, also known as the focus point, is defined as the intersection of the line of sight with the screen.

Gaze tracking has an infinite number of applications such as monitoring driver alertness or helping track a person's eyes with a psychological disorder that cannot communicate his/her issues. Gaze tracking is also used as a human-machine interface for disabled people that have lost total control of their limbs. Another application of gaze estimation is marketing. Companies use the information given by the gaze estimation system from their customers to design their advertisements and products.

1.1 Research Objective

The objective of this thesis is to develop and implement a gaze estimation system that tracks the head and eyes of a person in front of a computer in real time using only a single regular webcam. Opposed to infrared (IR) camera which forms an image using infrared radiation, a regular webcam forms images using visible light only.

The proposed system should be able to display the gaze point onto the monitor of the person sitting in front of the screen. The main constraint of such a project is to design a system that

requires no additional/special hardware aside from a webcam. The ideal solution would consist of a personal computer and a single regular webcam. To date, there are a couple of commercial gaze estimation systems on the market. Unfortunately, the biggest disadvantage of these existing solutions is the need of special equipment, such as dedicated devices mounted on the head or infrared cameras. These limitations prevent those invasive systems to go main stream. Thus it is easy to see the benefits of the proposed system that requires no additional hardware other than a computer, desktop or laptop, equipped with a normal webcam. However, one must remember that even though the use of a webcam makes the proposed system very convenient, it is less accurate than the ones commercially used. Another issue is that the proposed system needs to work in real time therefore it is vital to employ fast algorithms which limit computational complexity. Like every image processing task, lighting is always a major obstacle that needs to be taken into consideration, especially when the goal is to design a system as convenient as possible.

Because of these several issues, gaze trackers are still not part of our everyday life. It is not an easy task to create a gaze estimation system that is convenient by using only a webcam. The ideal gaze system is to operate automatically and be precise for all types of people and environments, which is a very tough challenge.

1.2 Contributions

In developing a universal gaze estimation system, two main contributions have been made which are applicable. The main contributions in this thesis are as follows.

The first contribution of the proposed gaze estimation system is that it takes in consideration head movements by using a head pose tracking algorithm with automatic re-initialization. The proposed algorithm goes further than OpenGazer, one of the most popular gaze estimation

systems, which works only by eye movements. We developed a solution that not only works with a standard webcam but also works with head movements. The proposed method determines the head pose, finds the eye gaze and computes the overall gaze vector of the user. The gaze point is shown on the computer screen as the intersection of the gaze vector and the monitor plane. To determine the head pose, the face is detected using the Viola-Jones algorithm also called Haar classifier. The features that are the most traceable are selected inside the face region to be the ones located at the pixels with the highest eigenvalues. Once those facial corners are detected, they are mapped to 3D features onto a 3D face model. The pose (rotation and translation) is then determined from the relationship between 2D image and 3D face model using the POSIT algorithm. When the program starts, the user is asked to look at the center of the monitor so that the pose is initialized. The 2D features are tracked across multiple frames using the Lucas-Kanade optical flow technique. The image features are mapped back onto the 3D model and the pose is updated. The POSIT algorithm accumulates error over time. Therefore, an extension of the algorithm is proposed by automatically resetting the algorithm using specific constraints. By re-initializing the algorithm, errors caused by illumination changes are reduced.

The second main contribution is the detection of the eye pupil center. To detect the eyes, the Viola-Jones technique is used again. The Haar classifier is now trained to detect eyes within the face region. Once the eyes are detected, their features are extracted for pre-processing to locate the pupil. The pre-processing involves color space conversion, histogram normalization, Otsu thresholding, and dilation to facilitate the detection of the pupil. The eye detected image by the Haar Classifier still contains a lot of useless information such as the skin. Because the skin is illuminated intensively, it influences the thresholding when detecting the eyeball. To remove the skin, the eye image is converted into HSV color space. Knowing that almost all humans have the same hue, it is easy to remove the skin by modifying the hue value. As a result, the eye

image only contains a little amount of skin. The image is cropped to a dimension approximate to that of the eye and then converted to gray. Finally, the resulting image is iteratively thresholded using constraints based on the anatomy of human eyes. At the end, the pupil is detected as the center of gravity of the resulting image. The use of geometric constraints, as well as the normalized center of gravity, helps in avoiding miscalculations of the pupil center due to lighting variations.

This paper presents a system that is based exclusively on one standard webcam using recent image processing breakthrough. It works in real time at different distances from the camera and the system can be used successfully when using personal laptops with built in webcams under any lighting condition.

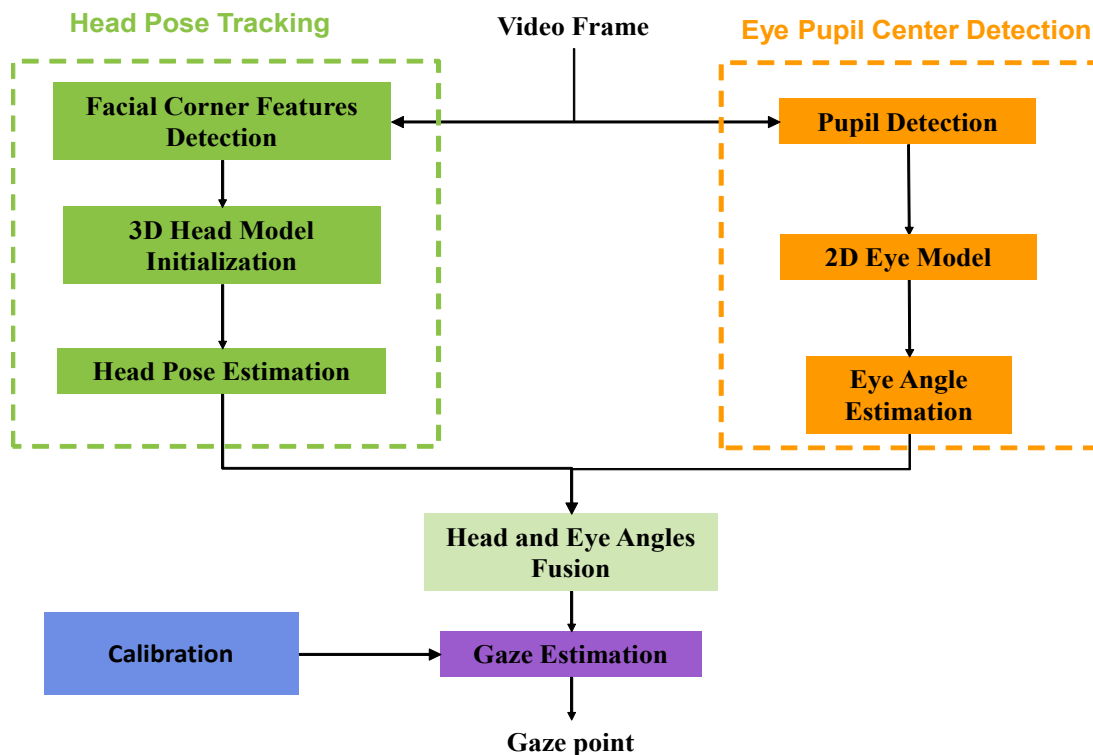


Figure 1: Overall System Block Diagram

CHAPTER 2

BACKGROUND STUDY

There are many methods to estimate the gaze of a person in front of a monitor. The existing research in gaze estimation includes intrusive and non-intrusive systems. Selecting the optimal method depends on the purpose of the system. The proposed gaze estimation system must be accurate, fast, low-cost and user friendly. In order to build the right system, it is important to first review the different methods used to estimate the gaze.

2.1 Commercial and Invasive Gaze Trackers

On the market, there is a whole range of commercial systems for eye tracking. The leading products are Tobii and Senso Motoric Instrument (SMI). These systems owe their popularity to the fact that they are very accurate despite the fact that they require extra infrared (IR) light sources or mounted devices on the user's head. Their functionality is based on the use of cameras and carefully positioned one or multiple IR light sources. The gaze is estimated by calculations between the center of the user's pupil and position of the reflection on the iris caused by the IR emitter.

The world's leading seller of gaze tracking devices is called Tobii. The Tobii tracker uses pupil corneal reflection to estimate the gaze of the user. Although the accuracy of Tobii is admirable it is certainly not user friendly. The Tobii system has to be transported in a special carrying case that includes a huge monitor equipped with IR emitters and a separate laptop. In addition, a special Tobii agent is needed to supervise the entire experiment. The Tobii system also includes an extended calibration step that frequently involves a questionnaire.

Senso Motoric Instrument (SMI) is one of Tobii biggest competitors. Their gaze system consists of a special mounted pair of glasses that is equipped of multiple HD cameras. The SMI eye tracking glasses are worn like a normal pair of glasses and the binocular tracking gives a remarkable accuracy of the user's gaze. The HD scene camera mounted on the pair of glasses records the user's visual at all times and distances. Just like Tobii, the SMI fails to deliver a completely natural, user friendly and comfortable solution to gaze tracking because of the need of special accessory equipment.

Intrusive systems uses special equipment such as electrodes, radar range finder or special glasses with a small webcam mounted on them to track the user's eyes. Those systems tend to be expensive and unfriendly as they require special hardware.

2.2 Non-Invasive Gaze Trackers

To make gaze estimation more natural, non-intrusive techniques have been proposed. Non-intrusive systems fall into two main categories; pupil center corneal reflection technique that use infrared illumination and normal video cameras technique that do not require infrared.

2.2.1 IR Gaze Trackers

IR Gaze trackers use the pupil corneal reflection method. The pupil corneal reflection technique uses at least one infrared light to illuminate the eye. A camera then captures images of the eye for analysis and the pupil and the infrared corneal reflection positions are detected. Based on the distance between the pupil and the corneal reflection, the gaze direction is calculated using geometry. The problem with this technique is that it only works in an indoor-controlled environment, which makes it inconvenient. Indeed, the infrared system requires intensive calibration because the position between the infrared and the camera needs to be known in advance. In addition, these systems are very sensitive to light because they are based solely on

the position between the pupil and the infrared illumination. Therefore, the pupil corneal reflection technique is not adequate to human-computer interactions because of its impracticality.

2.2.2 IR Free Gaze Trackers

To overcome those difficulties, techniques using only video cameras have been developed: appearance-based methods and model-based methods.

Appearance based methods [Ono et al. 2006; Morency et al. 2006] consist of estimating the gaze from comparing a new image of an eye to a set of eye images contained in a database in order to find the best match. One of the main disadvantages of appearance-based methods is that extracting the pupil position precisely is very difficult. The accuracy of such a system is often based on how much data is used. Therefore, those systems require a lot of data to be accurate and the system tends to be too slow and fails in real time tracking.

Model based methods consist of using the shape of the eye to build an eye model. The position of the eye features as well as some other facial features is utilized to update the eye model and estimate the current gaze. Various techniques are used to estimate the gaze from an eye model. The two most popular techniques are the circle algorithm [Wang and Sung 2001] and the 3D eye-model [Matsumoto and Zelinsky 2000]. The circle algorithm fit the iris into an ellipse shape and from ellipse parameters, the gaze is estimated. Unfortunately, the circle algorithm depends highly on the resolution of the eye images in order to shape the ellipse correctly. Therefore when used with a regular webcam, the circle algorithm performs poorly. The 3D eye model estimates the gaze as a vector from the eyeball center to the iris center. The disadvantage of the 3D eye model is that it often requires calibration to estimate the relationship between the eye and the 3D eye model.

This thesis presents a non-intrusive infrared free hybrid model for gaze estimation, which combines the advantages of both appearance and model based models. The proposed method detects the face and eye region using an appearance based approach, and estimates the gaze using a simplified 2D eye model that requires a limited amount of calibration and deals with head movements.

CHAPTER 3

HEAD POSE ESTIMATION

A large number of systems that are available for eye tracking imply limited head movements, which is not a practical solution. In human-computer interaction, a person is not able to keep his head still for a long period of time when looking at a computer screen because of breathing or muscle exhaustion. Additionally, applications such as driver awareness and marketing require head movements; therefore a system that does not tolerate head motion would be inadequate. Consequently, it is necessary to design a gaze estimation system that takes in consideration voluntary and involuntary head movements.

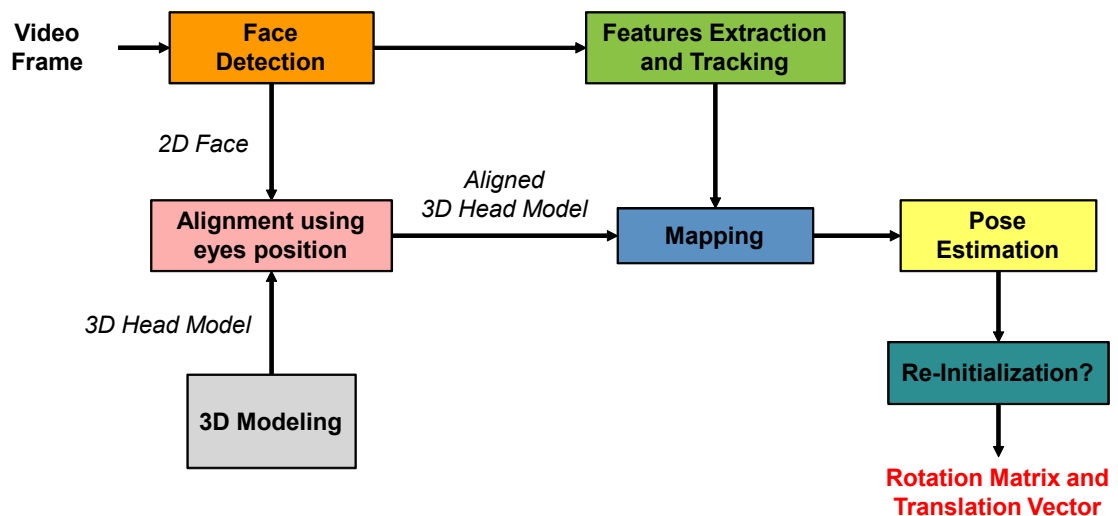


Figure 2: Head Pose Block Diagram

This paper presents one of the most popular approaches in recent years to track the head position. The face is detected using Viola/Jones face detector. The detected corner features within the face region are mapped to a 3D head model. The initial pose using POSIT algorithm is then estimated using the 2D feature points in the image and their corresponding 3D points on the head model. The pose of the 3D head model is then correlated with the user's head pose.

The pose is then a combination of a 3D rotation matrix R and a 3D translation vector T . The changes in position are tracked by applying optical flow to the detected facial features and the pose is updated. In case the pose does not satisfy the desired constraints, the 3D head model is re-initialized automatically.

3.1 Viola-Jones Face Detector

3.1.1 Detection of Face

The face is detected using the Viola-Jones detector also called the Haar classifier. It is one of the most popular solutions used for face detection because of its speed and efficiency. It uses the Haar transform features that consist of adding and subtracting rectangular image regions before thresholding the result. The Haar classifier detector is trained in this case to recognize faces. The Viola-Jones algorithm starts with positive (faces) and negative (non-faces) sample gray scale images. The Haar features are extracted from the sample images and a cascade of classifiers is trained using Adaboost which is a weighted voting method. As a result, we have a set of selected features and the classifier is applied to each subwindow of the given image until the face is found.

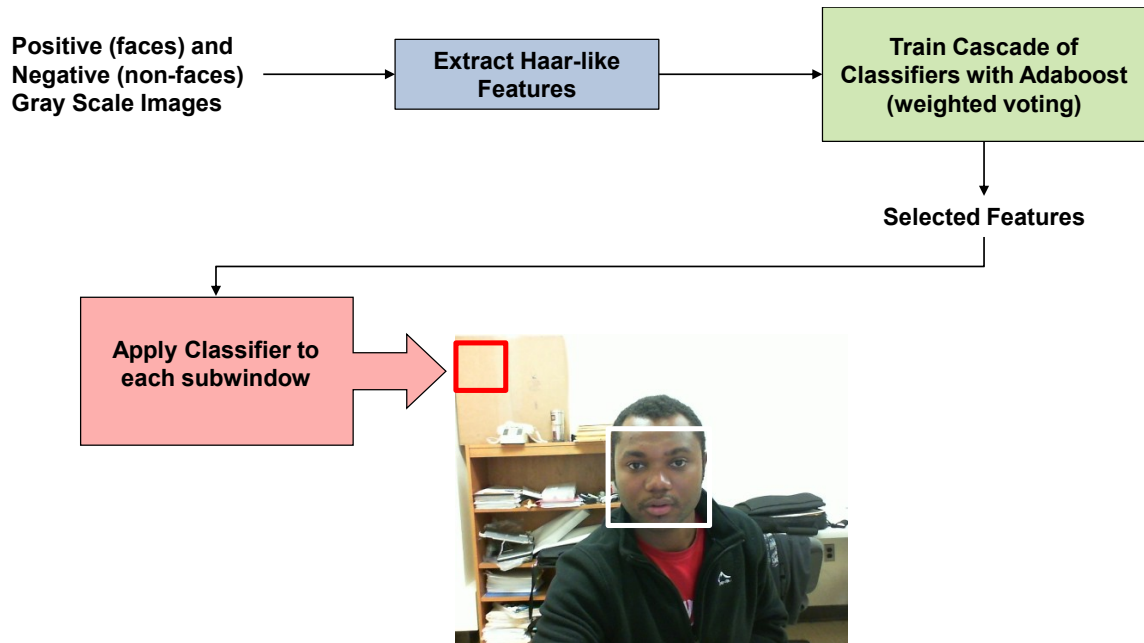


Figure 3: Face Detection using Viola-Jones

3.1.2 Optical Flow

Commercial detection systems for facial expressions and head movements generally use a marker-based technology, especially in Hollywood. This technology is known as facial motion capture where movements of a person's face are converted into a digital database. In movies such as Avatar or The Polar Express, actors such as Tom Hanks have several LED markers placed to their faces and are tracked with high resolution cameras. This solution is extremely accurate but not practical because it requires expensive additional equipment. The presented solution is marker-less; it is based solely on a standard webcam without the use of any additional support such as markers.

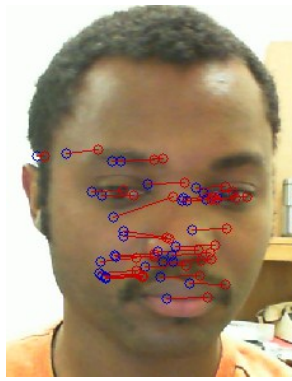


Figure 4: Optical Flow Sample Image. The red points represent features in the current frame and the blue points in the previous frame.

To track the changes in head position, we apply the Lucas-Kanade optical flow method. Optical flow is the apparent motion of brightness patterns in the image. Optical flow estimates pixel motion from one image to another.

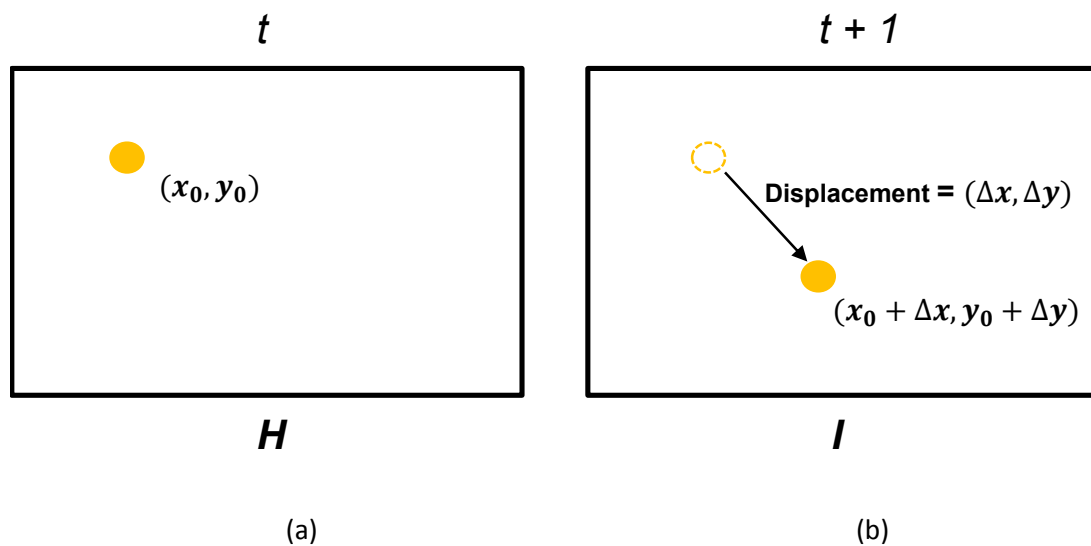


Figure 5: Optical Flow Assumptions. The point with brightness at time t is the same one moving to $(x + \Delta x, y + \Delta y)$ at time $t+1$ with approximately the same brightness value. The displacement between successive frames must be small.

Two assumptions are necessary to be made when applying optical flow. The first assumption is that brightness is constant which translates to:

$$H(x, y) = I(x + \Delta x, y + \Delta y) \quad (1)$$

The second assumption is that motion is small so we can derive:

$$\begin{aligned} I(x + \Delta x, y + \Delta y) &= I(x, y) + \frac{\partial I}{\partial x} \Delta x + \frac{\partial I}{\partial y} \Delta y + \text{higher order terms} \\ &\approx I(x, y) + \frac{\partial I}{\partial x} \Delta x + \frac{\partial I}{\partial y} \Delta y \end{aligned} \quad (2)$$

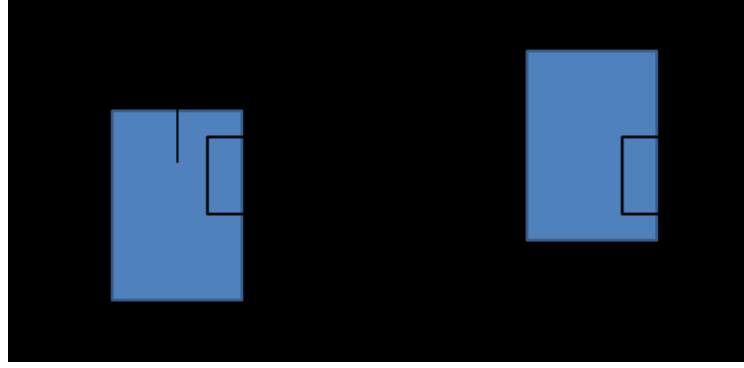
Then we compute optical flow by combining the two equations above such as:

$$\begin{aligned} 0 &= I(x + \Delta x, y + \Delta y) - H(x, y) \approx I(x, y) + I_x \Delta x + I_y \Delta y - H(x, y) \\ &\approx \underbrace{(I(x, y) - H(x, y))}_{I_t} + I_x \Delta x + I_y \Delta y \end{aligned} \quad (3)$$

When Δx and Δy goes to zero, the optical flow equation becomes exact:

$$0 = I_t + I_x \Delta x + I_y \Delta y \quad (4)$$

The optical flow equation shows that at each pixel we have one equation and two unknowns. This means that only the flow component in the gradient direction can be determined. In other words, the motion parallel to the edge cannot be determined; this is called the aperture problem.



(a)

(b)

Figure 6: Aperture problem. The motion was not detected because the dark window moves in a direction parallel to the edge.

At this stage we need to make one last assumption to solve the equation; the $(\Delta x, \Delta y)$ is constant within a small neighborhood of a pixel. Therefore if we use a 5*5 window of brightness values, the optical flow equation becomes:

$$\begin{array}{c}
 \begin{bmatrix} I_x(\mathbf{p}_1) & I_y(\mathbf{p}_1) \\ I_x(\mathbf{p}_2) & I_y(\mathbf{p}_2) \\ \vdots & \vdots \\ I_x(\mathbf{p}_{25}) & I_y(\mathbf{p}_{25}) \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = - \begin{bmatrix} I_t(\mathbf{p}_1) \\ I_t(\mathbf{p}_2) \\ \vdots \\ I_t(\mathbf{p}_{25}) \end{bmatrix} \\
 \underset{25 \times 2}{A} \quad \quad \quad \underset{2 \times 1}{d} \quad \quad \quad \underset{25 \times 1}{b}
 \end{array} \tag{5}$$

We have now 25 equations for only two unknowns. To solve for this system, we set up a least-squares minimization of the equation, where $\min \|Ad - b\|^2$ is solved as;

$$\begin{array}{c}
 (A^T A) d = A^T b \\
 \begin{matrix} 2 \times 2 & 2 \times 1 & 2 \times 1 \end{matrix}
 \end{array}$$

$$\begin{array}{c}
 \begin{bmatrix} \sum I_x I_x & \sum I_x I_y \\ \sum I_x I_y & \sum I_y I_y \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = - \begin{bmatrix} \sum I_x I_t \\ \sum I_y I_t \end{bmatrix} \\
 \underset{A^T A}{} \quad \quad \quad \underset{A^T b}{}
 \end{array} \tag{6}$$

This equation is only solvable when $A^T A$ is invertible which happens when it has full rank. In addition, the two eigenvalues of $A^T A$ should be large. Therefore, it is vital that we pick the right set of features, “good features”, for the Lucas-Kanade to work.

To decide which facial points are good to track, the Shi and Tomasi method is used. A good feature needs texture and corner. Texture exhibits local non-homogeneity in an image while a corner is more like the intersection of two edges within a local neighborhood. Features that have texture and corner have big eigenvalues. Using the Shi and Tomasi technique, the pixels with big eigenvalues are then selected in the search area.

Features with strong edges are usually located around the mouth, eyes and nose. In a situation where the subject makes a lot of different quick facial expressions, this algorithm will have a hard time to track those features which may lead to incorrect results. The number of features selected varies between 20 and 40. In one hand if the number of features is greater than 40, the quality of the features is not high enough and the points are not robust enough for tracking. In the other hand if the number of features is less than 20, the loss of a point greatly affect the tracking result because there are few features.



Figure 7: Facial Features Extraction

3.2 Mapping of Features to 3D Head Model

The head is modeled using a sinusoidal 3D head model constructed in Blender, a 3D computer graphics software. The head model does not take into account the detailed estimation of the shape of the user's head, but gives satisfactory results. Using a universal average human head model has several advantages such as it does not demand prior preparation to fit the user which means automatic initialization. Furthermore, the speed is maximized due to the simplicity of the model.

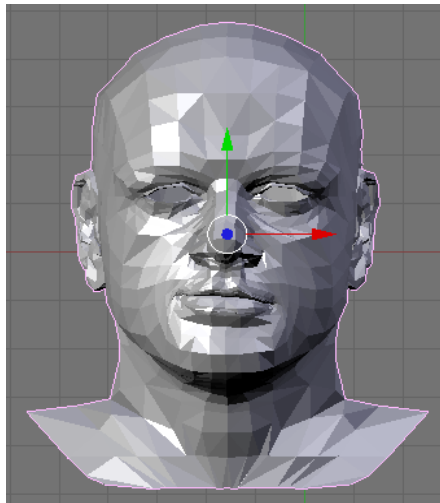


Figure 8: 3D Head Model created in Blender

The alignment and scaling between the face image and head model is done according to the position and distance between the two eyes. In addition, the mapping of the 2D facial corner feature points on the 3D head model is possible thanks to the assumption that during initialization the user's head is facing straight at the monitor. In other words, we make the 3D head face model coincide with the image obtained from the camera. From that initial pose, the change in features between successive frames are tracked and mapped back onto the 3D face model and the head pose is updated. The 3D model assesses changes in head position relative to the preliminary fixed position during initialization.

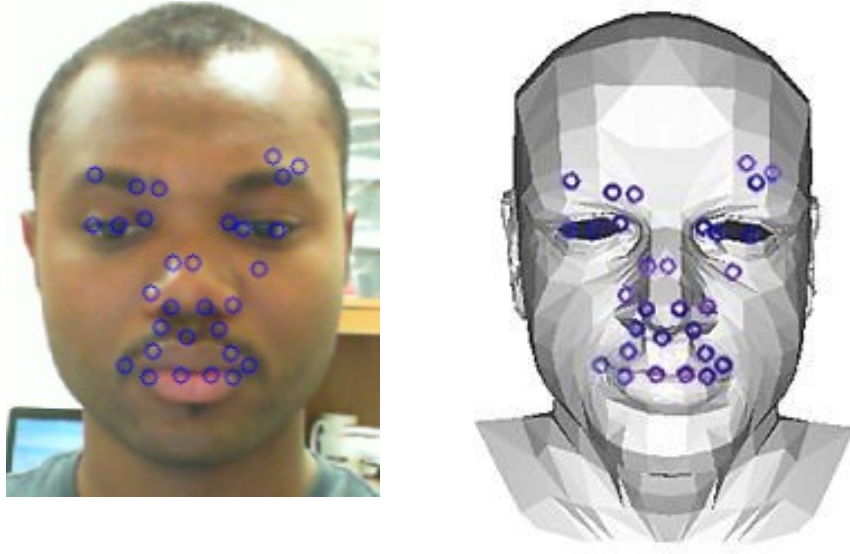


Figure 9: Mapping of 2D image features to 3D model

From the mapping of the 2D facial features in the image and their corresponding points on the 3D head model, the next step is to determine the head pose (rotation and translation) of the user using POSIT algorithm.

3.3 POSIT Algorithm

The relationship between a 3D point (X, Y, Z) on the head model and a 2D image point (u, v) is expressed as;

$$s \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \quad (7)$$

where (X, Y, Z) are the coordinates of a 3D point in the world coordinate space, (u, v) are the coordinates of the projection point in pixels. A is the camera matrix containing the intrinsic parameters; (c_x, c_y) is the image center, s is the scale factor and f_x, f_y are the focal lengths expressed in pixel units. $[R/t]$ is the joint rotation-translation matrix, also called extrinsic parameters.

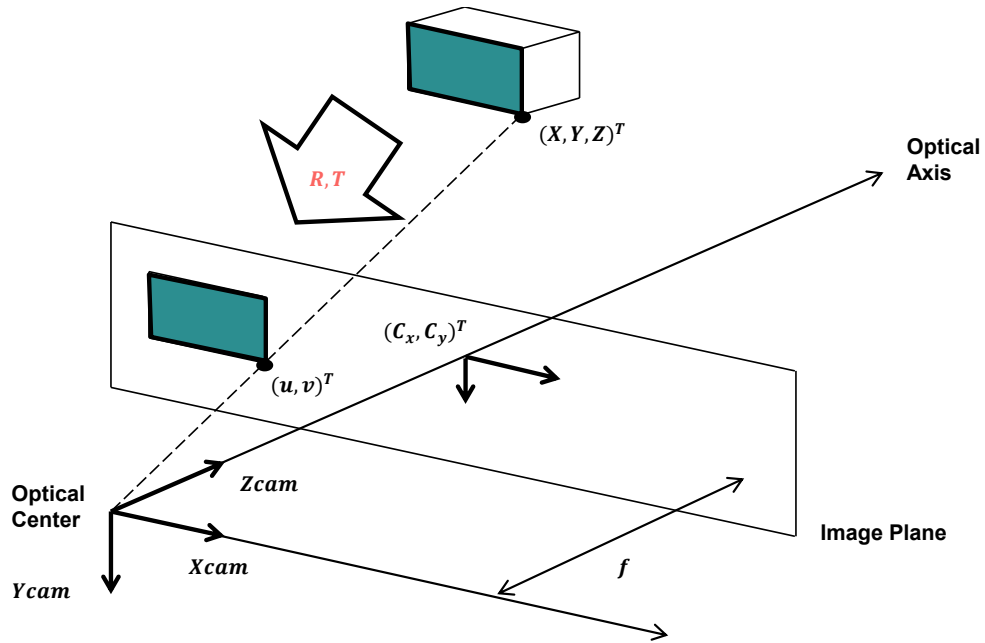


Figure 10: Relationship between a 3D point $(X, Y, Z)^T$ and its corresponding 2D projection $(u, v)^T$ onto the image plane

The POSIT algorithm is used to estimate the position in three dimensions of a known object. The 3D pose of an object includes rotation and translation. The POSIT algorithm requires image coordinates of at least four object's points. It is crucial that these points are not coplanar; the points must not belong to the same plan. The 3D model coordinates of these points must be known as well. Finally, the algorithm necessitates the focal length of the camera used to picture the object. In our case, the image coordinates of the object are the coordinates of the detected features on the face and the 3D model coordinates of these points are their corresponding points on the 3D head model. The focal length is estimated using the face width.

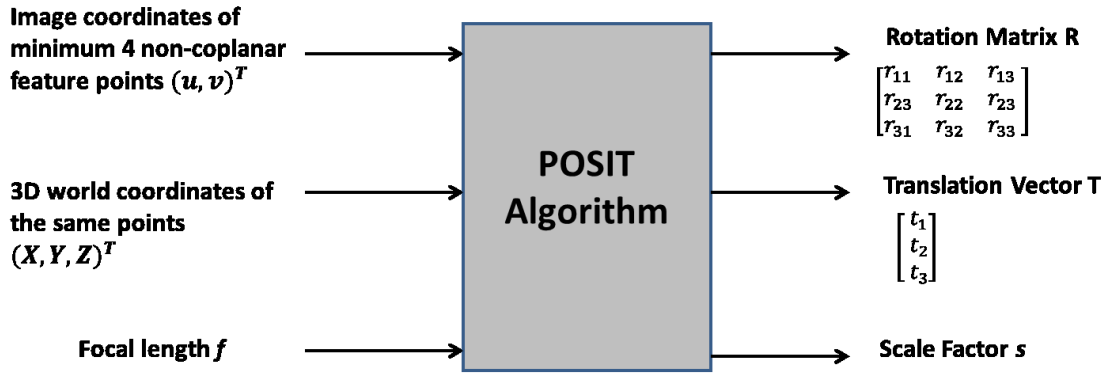


Figure 11: POSIT Algorithm Inputs/Outputs

First let's understand some notations about the pinhole camera model. All M_x belong to the 3D model and all m_x belong to the image plane G . The plane K is parallel to the plane G at a distance Z_0 . M_i projects perspectively on N_i (on K) and m_i (on G). M_i projects orthographically on P_i and P_i projects perspectively on p_i .

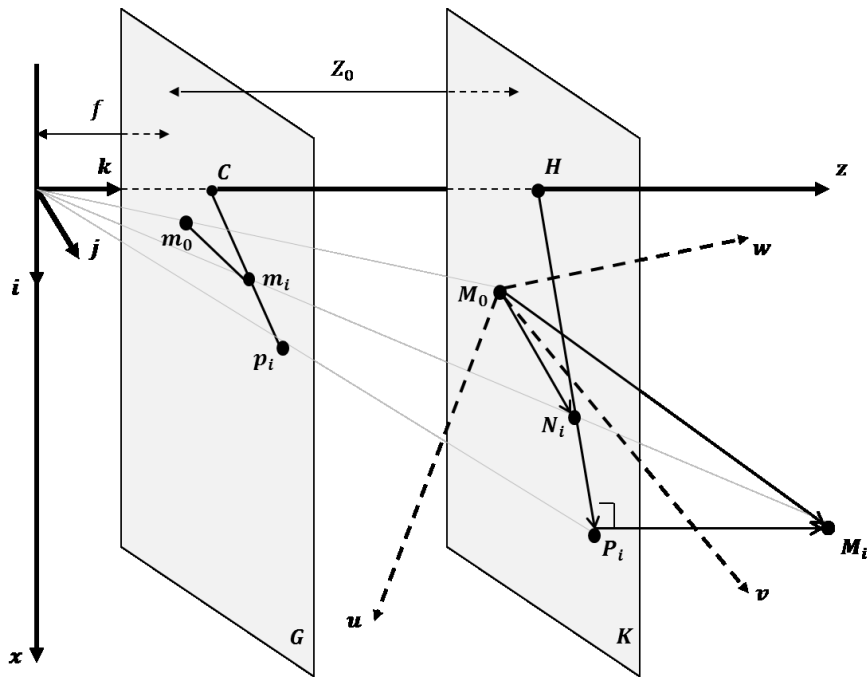


Figure 12: Perspective projection (m_i) and scaled orthographic projection (p_i) for an object point M_i and a reference point M_0

Let's assume the rotation matrix R defined by the unit vectors i, j, k of the camera coordinate system expressed in the object coordinate (M_0u, M_0v, M_0w) such as;

$$R = \begin{bmatrix} i_u & i_v & i_w \\ j_u & j_v & j_w \\ k_u & k_v & k_w \end{bmatrix} \quad (8)$$

Once i and j are computed, k is obtained by taking the cross product of i and j . If Z_0 (depth of M_0) is found, it is possible then to compute M_0 because of the alignment between T and Om_0 such as $T = \frac{Z_0}{f} * Om_0$ with f being the focal length.

In scaled orthographic position, the image of a point M_i of an object is a point p_i of the image plane G which has coordinates;

$$x'_i = \frac{fX_i}{Z_0} \quad (9)$$

$$y'_i = \frac{fY_i}{Z_0} \quad (10)$$

while for perspective projection an image point m_i would be obtained instead of p_i with coordinates

$$x_i = \frac{fX_i}{Z_i} \quad (11)$$

$$y_i = \frac{fY_i}{Z_i} \quad (12)$$

These can be combined to:

$$x'_i = \frac{fX_0}{Z_0} + \frac{f(X_i - X_0)}{Z_0} = x_0 + s(X_i - X_0) \quad (13)$$

$$y'_i = y_0 + s(Y_i - Y_0) \quad (14)$$

The ratio $s = f/Z_0$ is the scaling factor.

Now i and j from the rotation matrix and Z_0 from M_0 are combined with M_0M_i and coordinates x_i and y_i from image points m_0 to m_i such as:

$$M_0M_i * \left(\frac{f}{Z_0}\right) i = x_i(1 + \varepsilon_i) - x_0 \quad (15)$$

$$M_0M_i * \left(\frac{f}{Z_0}\right) j = y_i(1 + \varepsilon_i) - y_0 \quad (16)$$

The terms ε_i are defined as $\varepsilon_i = \frac{1}{Z_0} M_0M_i \cdot k$. Next, we solve the equations above and we get i and j . We also get Z_0 as f , the focal length, is known. We start setting $\varepsilon_i = 0$ assuming scaled orthographic image points and perspective image points coincide. Then once we know i, j and Z_0 we can estimate a better ε_i and this leads to better values for i, j and Z_0 in the next iteration.

3.4 Head Pose with Euler Angles

The head pose is updated after each movement and given in the form of a matrix that can be viewed as a multiplication of three rotations, one about each principle axis. Matrix multiplication does not commute so the order of the axes is important. In this paper, we rotate first around the x-axis, then the y-axis and finally the z-axis such as;

$$R = R_z(\emptyset)R_y(\theta)R_x(\gamma) \quad (18)$$

A rotation of γ radians about the x -axis is defined as:

$$R_x(\gamma) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos \gamma & -\sin \gamma \\ 0 & \sin \gamma & \cos \gamma \end{bmatrix} \quad (19)$$

Similarly, a rotation of θ radians about the y -axis is defined as:

$$R_y(\theta) = \begin{bmatrix} \cos \theta & 0 & \sin \theta \\ 0 & 1 & 0 \\ -\sin \theta & 0 & \cos \theta \end{bmatrix} \quad (20)$$

Finally, a rotation of \emptyset radians about z -axis is defined as:

$$R_z(\emptyset) = \begin{bmatrix} \cos \emptyset & -\sin \emptyset & 0 \\ \sin \emptyset & \cos \emptyset & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad (21)$$

Combining these three, the rotation R is expressed as:

$$R = \begin{bmatrix} \cos \theta \cos \emptyset & \sin \gamma \sin \theta \cos \emptyset - \cos \gamma \sin \emptyset & \cos \gamma \sin \theta \cos \emptyset + \sin \gamma \sin \emptyset \\ \cos \theta \sin \emptyset & \sin \gamma \sin \theta \sin \emptyset + \cos \gamma \cos \emptyset & \cos \gamma \sin \theta \sin \emptyset - \sin \gamma \cos \emptyset \\ -\sin \theta & \sin \gamma \cos \theta & \cos \gamma \cos \theta \end{bmatrix} \quad (22)$$

Given the rotation matrix R , the Euler angles γ , θ and \emptyset are then computed by equating each element in R with its corresponding element from the POSIT rotation matrix.

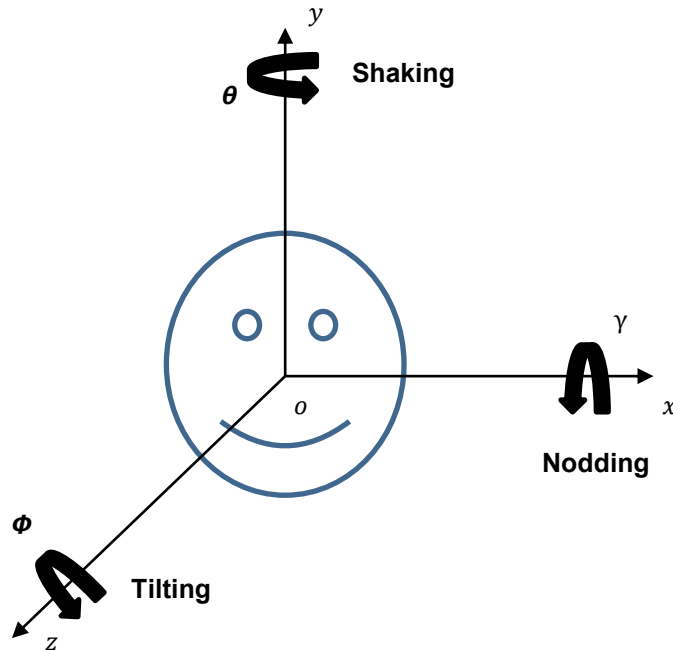


Figure 13: Head Pose with Euler Angles

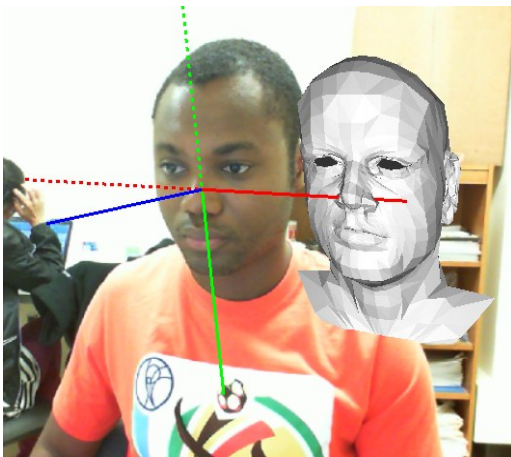
3.5 POSIT Algorithm Tracking Failure and Correction

During initialization, the POSIT algorithm maps the corner feature points onto the 3D head model and the user is required to front at the camera for a few frames at the beginning of the tracking process. When graphing the performance of POSIT over time, one significant observation is made. The original POSIT algorithm fails after an average of 80 frames. The error accumulates over time, especially when the algorithm deals with large head movements. In order to correct such a limitation by POSIT algorithm, the proposed method extends the original POSIT algorithm by implementing an auto reset of the algorithm when three checks are verified. The first check is that the current position of the face must be within 20% of the previous position at initialization. The reset will then only occur when the head is close its initial position.

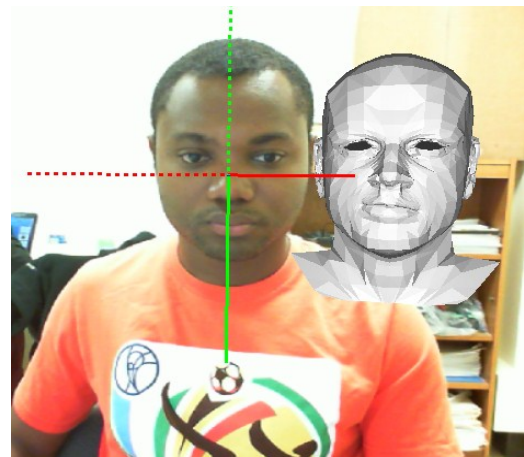
The second check states that the Euclidean distance of the head Euler angles must be less than a desired threshold estimated during calibration such as:

$$\sqrt{\gamma^2 + \theta^2 + \phi^2} < \text{threshold} \quad (23)$$

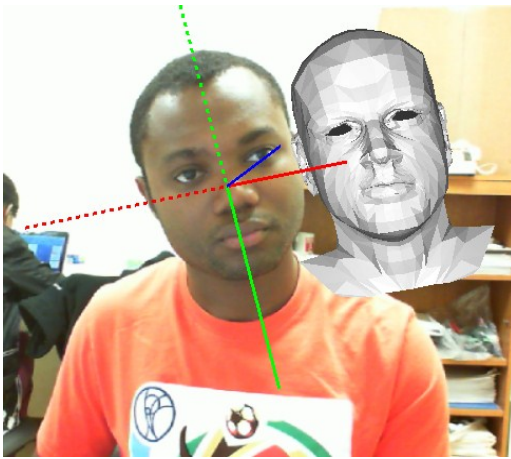
If the head is back to its initial position, it is only natural to expect that the distance between the Euler angles and zero should be minimal. The last and third check is to only do the re-initialization process after a minimum of 40 frames. This gives some time to the program to re-adjust itself. The auto reset only occurs when the three checks have been confirmed.



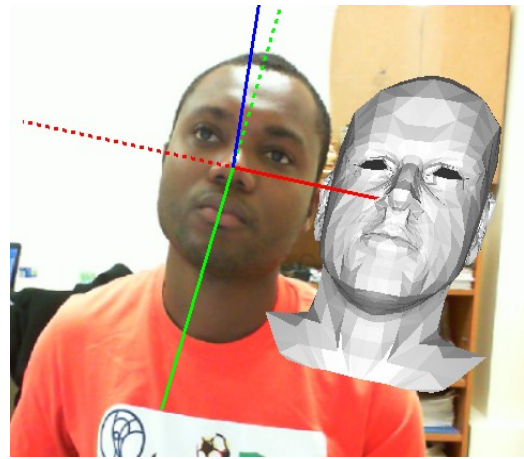
(a) $(2.37^\circ, -4.34^\circ, -5.07^\circ)$



(b) $(0.13^\circ, -0.94^\circ, -0.08^\circ)$



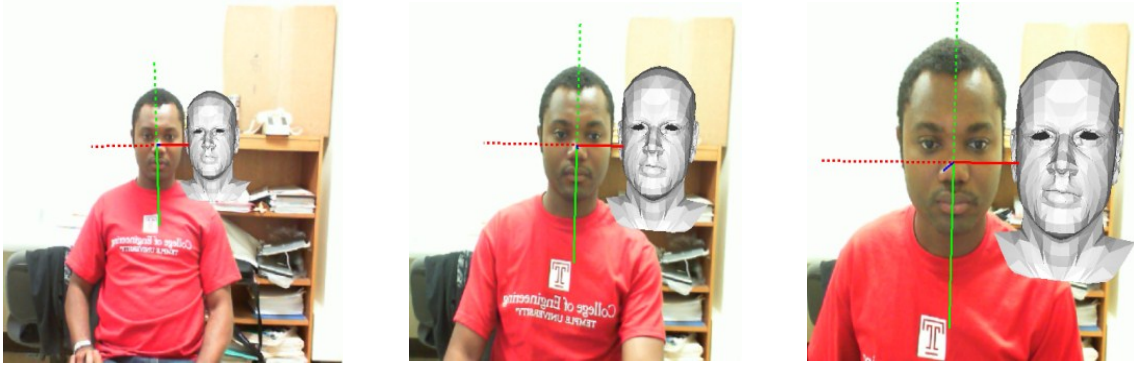
(c) $(-10.84^\circ, 6.26^\circ, 3.99^\circ)$



(d) $(8.42^\circ, -2.13^\circ, 7.12^\circ)$

Figure 14: 3D Head Model next to the face image in different poses. The angle for head tilting is represented by ϕ . The angle for head nodding sideways (shaking) is represented by θ . The angle for head nodding up and down is represented by γ . The head pose is then represented by (ϕ, θ, γ) .

Furthermore, it is necessary to adjust the scale of the pose accurately to obtain valid gaze results. The distance between the two eyes is used as a measure of how far away the user is from the camera (focal length). For example, if the detected head width is small, that means the user is further from the camera. On the contrary, if the head's width is large, that means the user is closer to the camera. As a result, it is possible to adjust the head model scale.



(a) $d = 85\text{cm}$

(b) $d = 43\text{cm}$

(c) $d = 12\text{cm}$

Figure 15: Head pose at different distances d . The distance d is the distance in meters between the user and the computer screen

The POSIT method allows us to compute the 3D position of an object using only one webcam. The described algorithm is a good choice for real time application because its computational complexity is low and it is robust to lighting changes.

CHAPTER 4

EYE PUPIL CENTER DETECTION

The proposed eye tracking method is a hybrid method that combines the strengths of appearance and feature based algorithms while limiting their weaknesses. The proposed pupil center detection system takes advantage of the precision of appearance technique. To make sure the proposed program runs in real time, the amount of training data is reduced to only detecting the face and rough position of the eyes in the image. Then, we exploit feature-based technique processing steps to locate the pupil center by using an adaptive thresholding that is robust against illumination changes. Figure 16 shows the overall procedures of the eye pupil center detection. The face and then the eyes within the face region are detected using the Viola-Jones appearance technique. Because the Viola-Jones method does not detect the eyes accurately, some pre-processing steps including color space conversion and Otsu thresholding are applied to each eye image in order to remove the skin and eyebrow parts around the eyes. Once the eyes are localized more precisely, an adaptive thresholding technique that uses the physical properties of the eye is utilized to localize the pupil center.

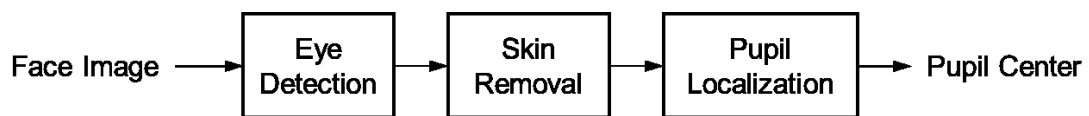


Figure 16: Pupil Center Detection Block Diagram.

4.1 Eye Detection

The detected face is divided horizontally into three parts. The top two parts are selected to be the region of interest that contains the eyes. By reducing the search area, the computation time is reduced. Using the same Viola-Jones detector, both eyes are detected. The face is then

divided into two parts vertically. The left eye is chosen to be the eye on the left side of the face and the right eye on the right side.

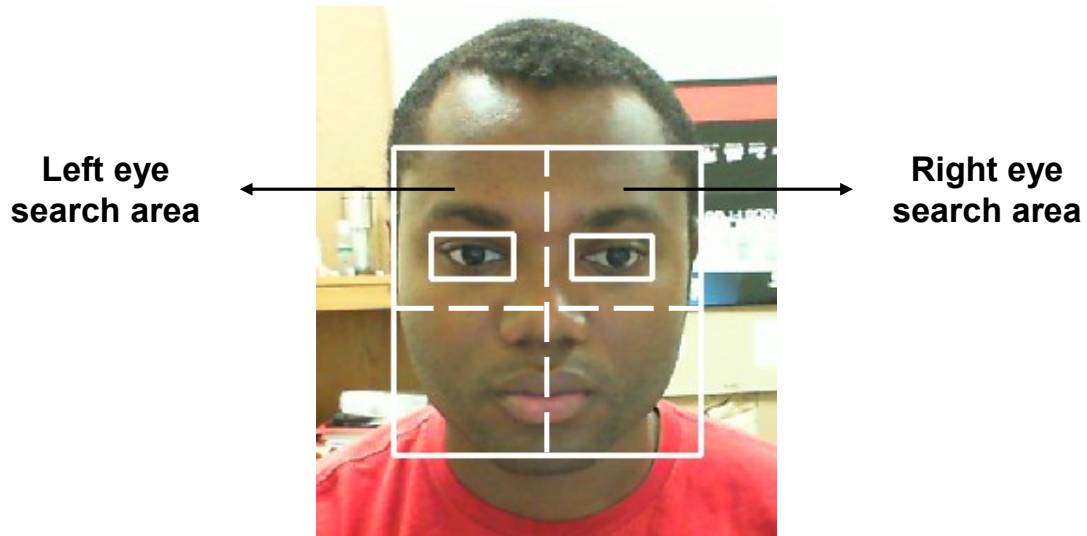


Figure 17: Eye detection from face detection

4.2 Pre-Emphasis: Skin Removal

In the previous section, we successfully detected the face and eyes using the Viola-Jones technique. However, at this stage the eye image detected by the Haar Classifier still contains a lot of useless information such as the skin and eyebrows. Because the skin is illuminated intensively, it influences the thresholding when detecting the pupil center. To remove the skin, the eye image is converted into HSV (Hue Saturation Value) color space. The HSV coordinate system is cylindrical.

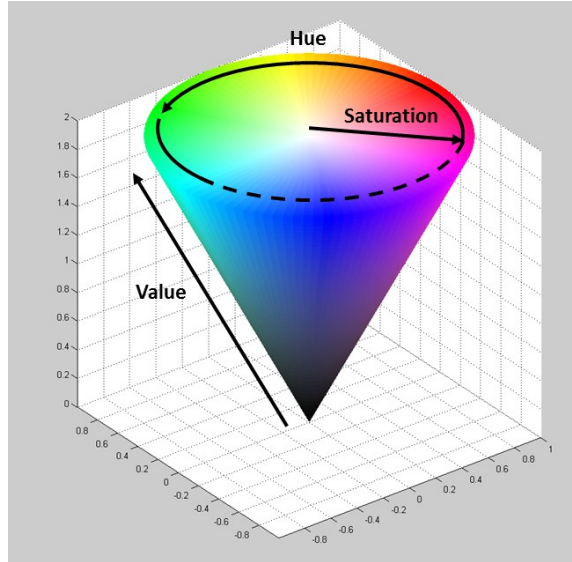


Figure 18: HSV Color space

The angle around the central vertical axis represents “hue”, the distance from the axis is the “saturation”, and finally the vertical position defines the “value” also referred the “brightness” (Fig. 18). Hue is measured in degrees from 0° to 360°. The equations below described how hue H is computed from RGB color space;

$$M = \max(R, G, B)$$

$$m = \min(R, G, B)$$

$$C = M - m$$

$$H' = \begin{cases} \text{undefined}, & \text{if } C = 0 \\ \frac{G - B}{C} \text{ mod } 6, & \text{if } M = R \\ \frac{B - R}{C} + 2, & \text{if } M = G \\ \frac{R - G}{C} + 4, & \text{if } M = B \end{cases}$$

$$H = 60^\circ \times H' \tag{24}$$

Knowing that almost all humans have the same hue, the skin is removed by Otsu automatic thresholding the hue component of the image. As a result, the eye image only contains a little amount of skin and no eyebrows. The image is then cropped to the nearly dimensions of the eye for further processing.

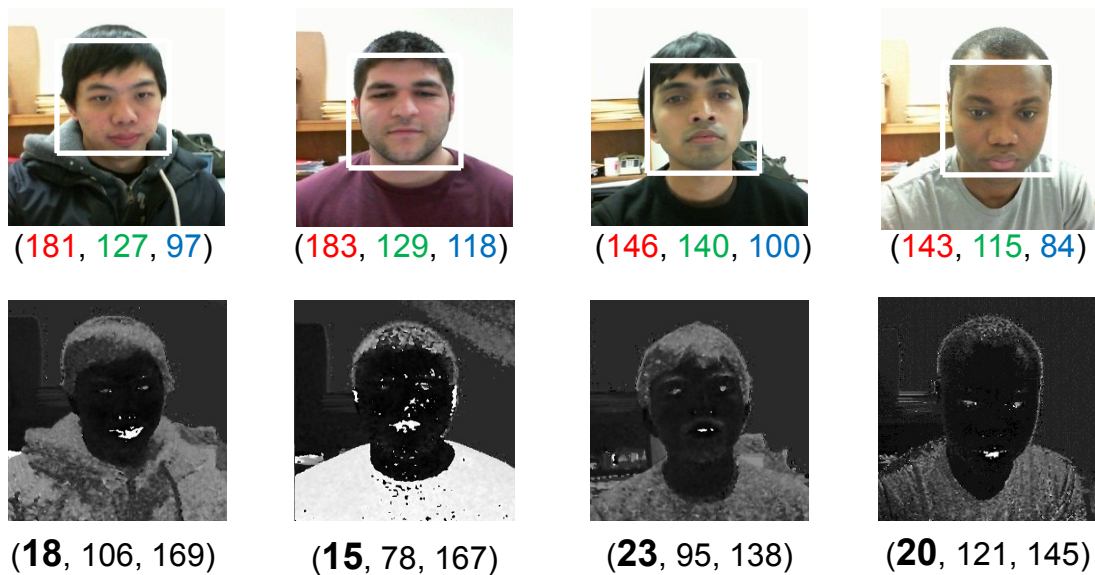


Figure 19: HSV to RGB intensity (0-255) component comparison

4.3 Pupil Center Detection

The pupil center is detected on each eye by the following method. First, the eye image without skin is up-sampled to twice its original size. Then, the resulting image is eroded and normalized to reduce the effect of the variations in illumination. At that point, the eye image is thresholded using an adaptive approach. The pupil is assumed to be darker than the background which is mainly the white of the eye. Therefore any pixel below a certain threshold value is labeled as the pupil. Because the environment especially illumination changes all the time, a fixed threshold

cannot be selected. Consequently, an initial threshold value t_i is selected such as t_i is large enough to contain at least the pupil and unfortunately some of the background as well. Then the image is iteratively thresholded until eye geometric constraints are satisfied and the desired threshold t_f is chosen.

The eye geometric constraints are created based on physical anthropological characteristics of human eyes:

- Two eyes region must belong to the same line
- If w_e and h_e are respectively the width and the height of an eye, then in average;

$$\frac{w_e}{h_e} = 2 \quad (25)$$

- If w_p and h_p are respectively the width and the height of a pupil, we have:

$$1 < \frac{w_p}{h_p} < 3 \quad (26)$$

- If the distance between the left and right eye is noted by d_{lr} , we have the relation:

$$d_{lr} = 2w_e \quad (27)$$

After applying the geometric constraints to the eye image, only the pupil remains. The pupil center is then estimated by computing the center of mass also called the center of gravity. Knowing $I(x, y)$ is the pupil image, the coordinates of the center of gravity x_c and y_c are defined as the spatial moments of first order $m_{1,0}$ and $m_{0,1}$ divided by the area A ($m_{0,0}$) of the object such as;

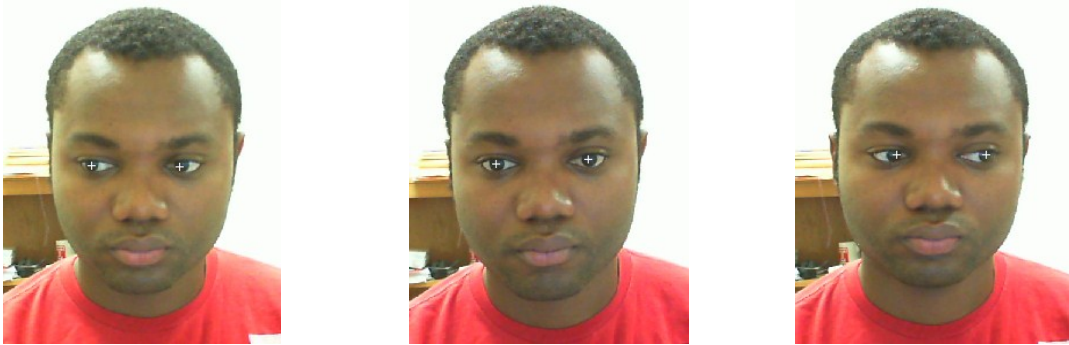
$$m_{p,q} = \sum_{i=1}^n I(x,y)x^p y^q \quad (28)$$

$$Area = m_{0,0} \quad (29)$$

$$x_c = \frac{m_{1,0}}{m_{0,0}} \quad (30)$$

$$y_c = \frac{m_{0,1}}{m_{0,0}} \quad (31)$$

The center of gravity has shown better results in determining the center of the pupil than computing the center of the contour area.



(a) $\theta' = -3.09^\circ, \gamma' = 0.05^\circ$ (b) $\theta' = 1.21^\circ, \gamma' = 0.36^\circ$ (c) $\theta' = 4.04^\circ, \gamma' = 0.83^\circ$

Figure 20: Pupil Center Detection. The angle in which the eye moves sideways is expressed by θ' and the angle in which the eye moves up and down is expressed by γ' .

To prevent the program to search for a pupil when the eye is closed, an eye blink detection method is implemented. The eye geometric constraints are also used to detect a closed eye.

Let's the pupil coordinates be (x_c, y_c) and the eye coordinates be (x_e, y_e) . Then, the pupil region that satisfies those two conditions is considered as *a closed eye*:

$$\begin{cases} \frac{w_p}{h_p} > 3 \\ y_c \approx y_e (10\%) \end{cases} \quad (32)$$

If the eye is closed for more than 3 frames, it is considered a closed eye instead of a blink.



Figure 21: Eye Blink. The right eye is closed while the left eye is open and the left pupil center is detected. Notice that no pupil cross is drawn on the right eye since it is closed.

4.4 2D Eye Model

Once we know the position of the pupil, it is possible to find their coordinate projections θ' and γ' . The eye projections are derived from the position of the pupil relative to the position of the eye corners. In order for the system to work in real time, a limited amount of computation is required which makes a 2D eye model appropriate for our purposes. The point $p(x, y)$ is the center of the pupil on the eye image. The point $C(C_x, C_y)$ is chosen to be the origin of the 2D eye model as the center of E_1 and E_2 as well as L_1 and L_2 . Knowing that $C_x = \frac{E_{1x} + E_{2x}}{2}$ and $C_y = \frac{L_{1y} + L_{2y}}{2}$, it is possible then to express the projections θ' and γ' onto the axis centered at C (fig.22) such as;

$$\theta' = x - C_x \tag{33}$$

$$\gamma' = y - C_y \tag{34}$$

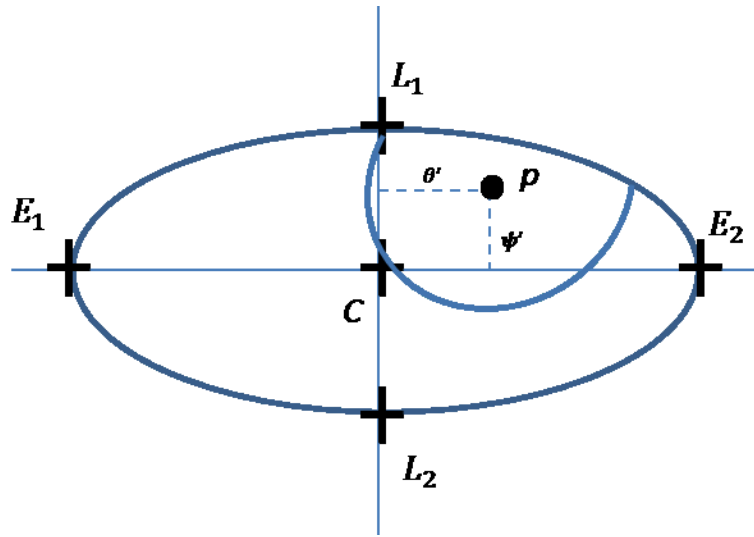


Figure 22: 2D Eye Model

CHAPTER 5

GAZE POINT COMPUTATION

Knowing the head pose and the positions of the eyes and pupil centers, we are now able to determine where the user is looking at on the monitor. The gaze point is the intersection of the line of sight with the screen.

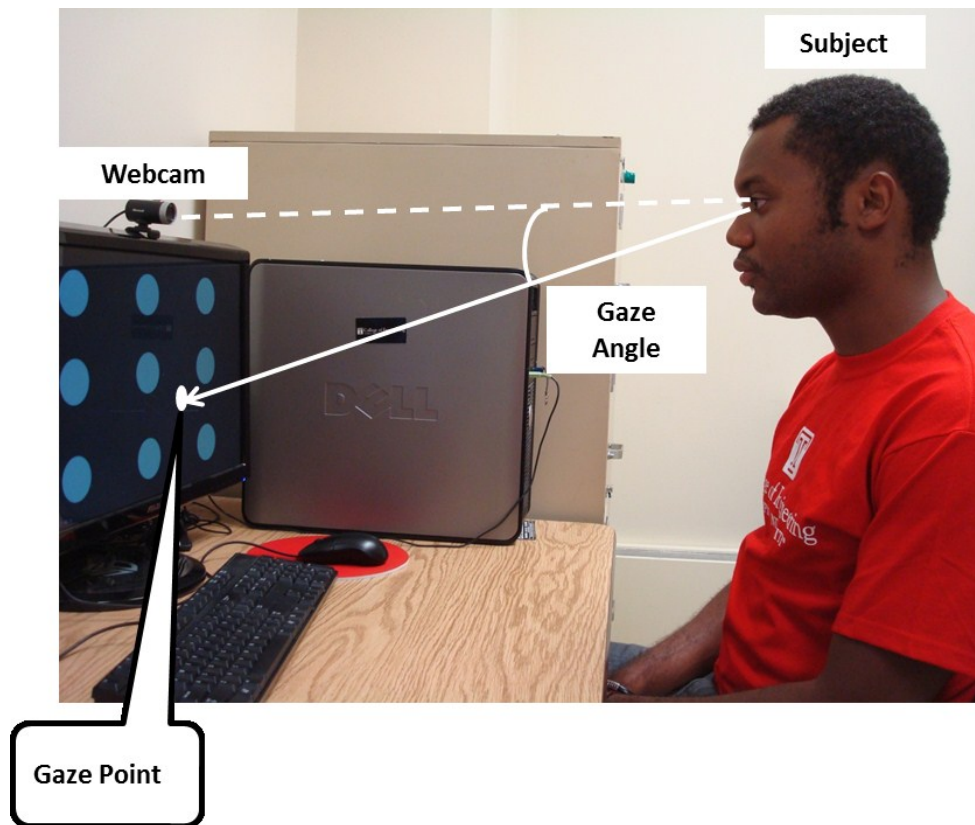


Figure 23: System Setup.

5.1 Fusion of Head Pose and 2D Eye Model

Let $P(X,Y)$ be the gaze point onto the computer screen. The gaze point in real time is computed using the linear combination of the head pose Euler angles (θ, γ) and the pupil coordinate projections (θ', γ') such as:

$$X = \alpha\theta + \alpha'\theta' \quad (35)$$

$$Y = \beta\gamma + \beta'\gamma' \quad (36)$$

The constants α , α' , β and β' are estimated during calibration.

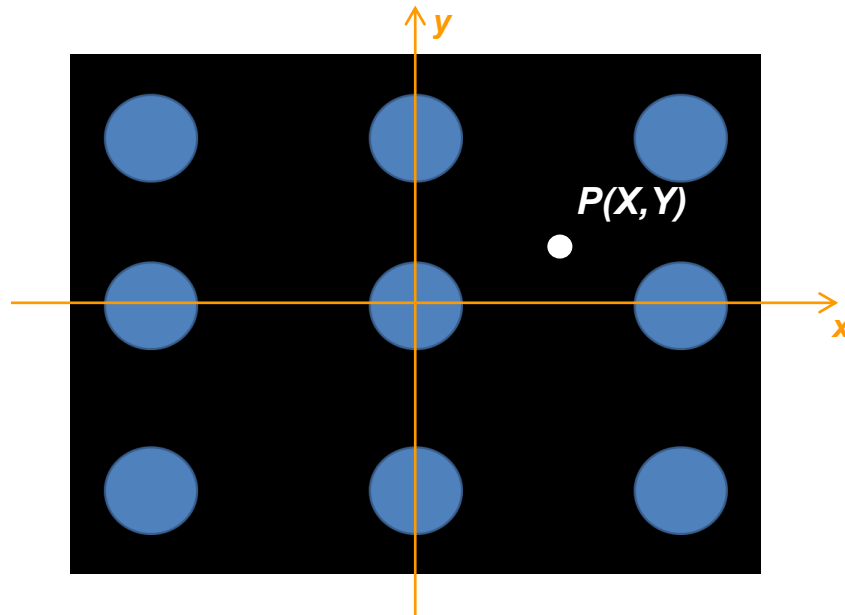


Figure 24: Gaze Point Estimation

5.2 Calibration

During calibration, the estimation of the parameters α , α' , β and β' is done. The user is asked to look at the 4 corners of the computer screen and the values of α , α' , β and β' are recorded. The parameters are estimated using the average method. The user first looks at the center of the screen. Then, the user looks at the four calibration points which correspond to the four corners of the screen. That calibration process facilitates computation of the parameters by allowing us to calculate α , α' , β and β' by simply averaging them such as;

$$\alpha = \frac{1}{5} \sum_{n=0}^4 \frac{X_n}{\theta_n} \quad (37)$$

$$\alpha' = \frac{1}{5} \sum_{n=0}^4 \frac{X_n}{\theta'_n} \quad (38)$$

$$\beta = \frac{1}{5} \sum_{n=0}^4 \frac{Y_n}{\gamma_n} \quad (39)$$

$$\beta' = \frac{1}{5} \sum_{n=0}^4 \frac{Y_n}{\gamma'_n} \quad (40)$$

By recording multiple sequences of different head and eye positions, the parameters α , α' , β and β' are computed and can be used to determine the gaze in real time.

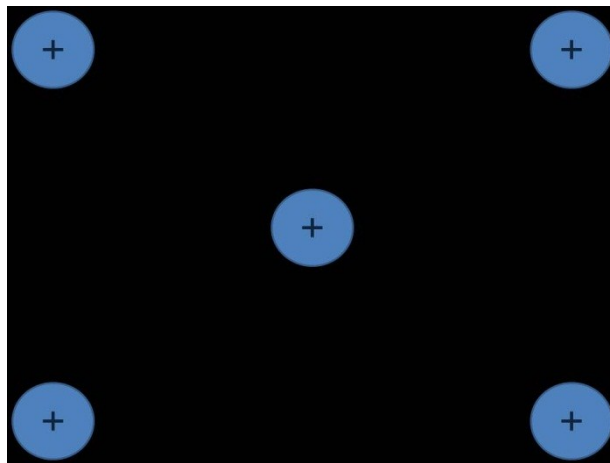


Figure 25: Calibration Markers. The calibration markers are the five blue crossed disks

CHAPTER 6

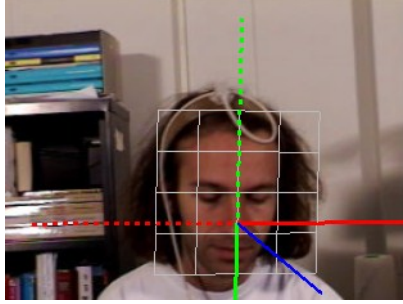
EXPERIMENTAL RESULTS

6.1 Head Pose Tracking Results

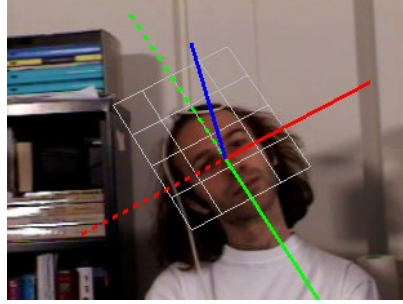
To evaluate the performance of the proposed head pose estimation and tracking system, the Boston University (BU) public database was used (<http://csr.bu.edu/headtracking/>). The BU database had been used by many head tracking papers. The BU database contains 72 videos and the ground truth for position and orientation of the head was obtained using a magnetic sensor. The BU database contains 2 set of video sequences of free head movements from different subjects. The first class of sequences includes 45 videos under uniform lighting conditions. The second class includes 27 videos under varying lighting conditions. The proposed algorithm was tested on all the 72 videos which are 200 frames long (approximately seven seconds) each.

The proposed algorithm is first tested on uniform lighting. The initialization is done in the first frames when the subject stares at the camera for a short period of time before moving freely at different distances from the camera. Then the proposed method tracks the user's head efficiently for different subjects and poses for a MAE (mean absolute error) of (3.779°, 3.943°, 4.834°).

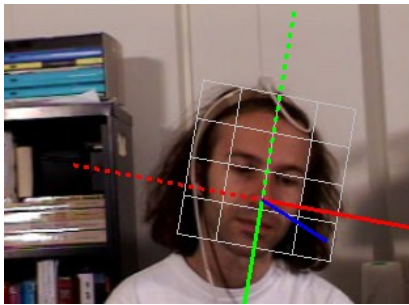
Next the proposed method is evaluated under varying lighting conditions in 27 videos for diverse subjects and poses at different distances from the camera. Although the performance decreases to (5.054°, 6.334°, 5.315°), the proposed method still manages to track the user's head under varying illumination.



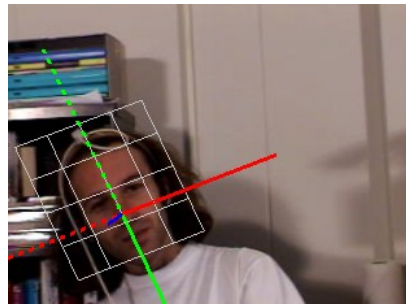
(a)



(b)



(c)



(d)

Figure 26: Uniform lighting. The figure shows results of the proposed head tracker in an example of the BU database under uniform lighting conditions for different poses.

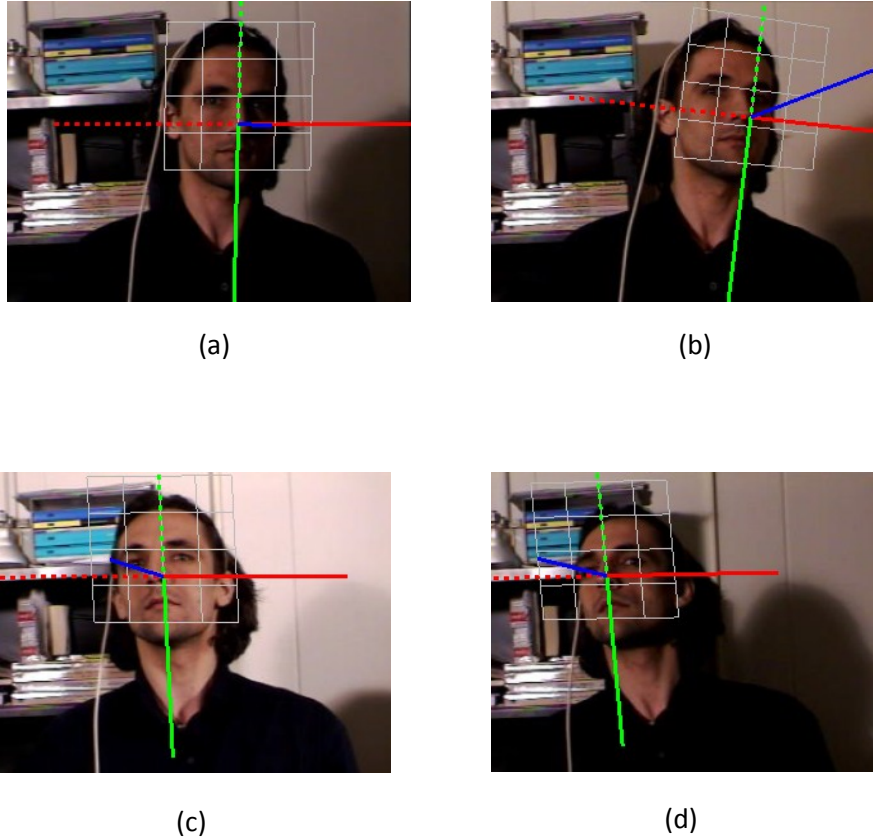


Figure 27: Varying lighting. The figure shows results of the proposed head tracker in an example of the BU database under varying lighting conditions for different poses.

The estimated angles and ground truth values are compared in all the 72 videos under uniform and varying lighting conditions by computing the average MAE. The average MAE for uniform illumination videos is $(3.779^\circ, 3.943^\circ, 4.834^\circ)$ while the average MAE for varying illumination videos is $(5.054^\circ, 6.334^\circ, 5.315^\circ)$. Because of illumination variations, the MAE dropped by an average of 1.3° which is negligible. The overall performance of the head pose estimation and tracking system is similar and in some cases better than other algorithms using uniform and varying illumination videos.

Table 1. Head Pose Tracking MAE Results

Illumination Condition	Tilting(°)	Shaking(°)	Nodding(°)
Uniform	3.779	3.943	4.834
Varying	5.054	6.334	5.315
Overall	3.940	5.161	5.312

To better evaluate the head pose, we use two graphical representations. The first set of graphs represents each head Euler angle in a video. As a result, we have 3 graphs per video with the angle value on the y-axis and the frame number on the x-axis.

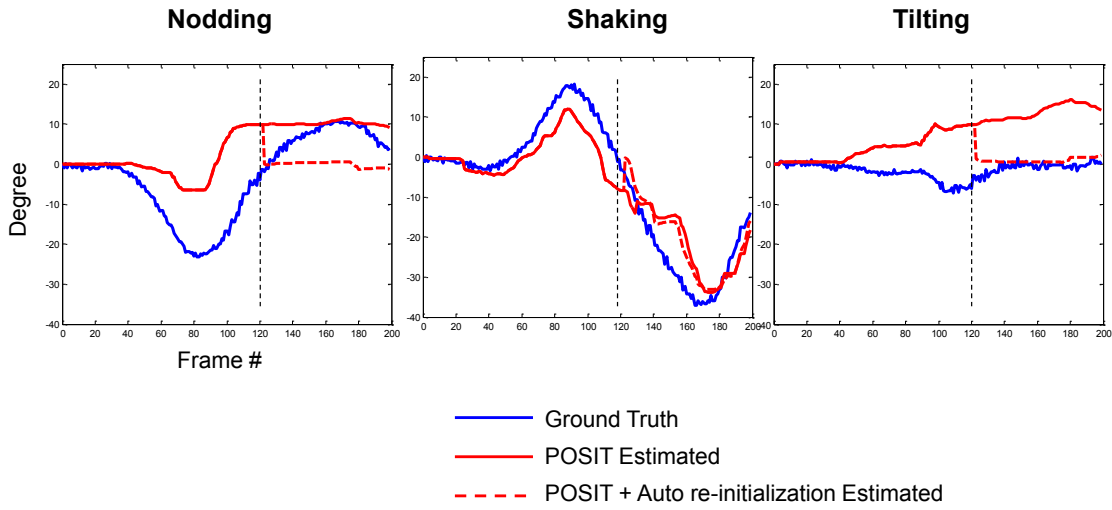


Figure 28: Uniform lighting graph. The auto re-initialization occurs around the 120th frame as shown by the black dotted line.

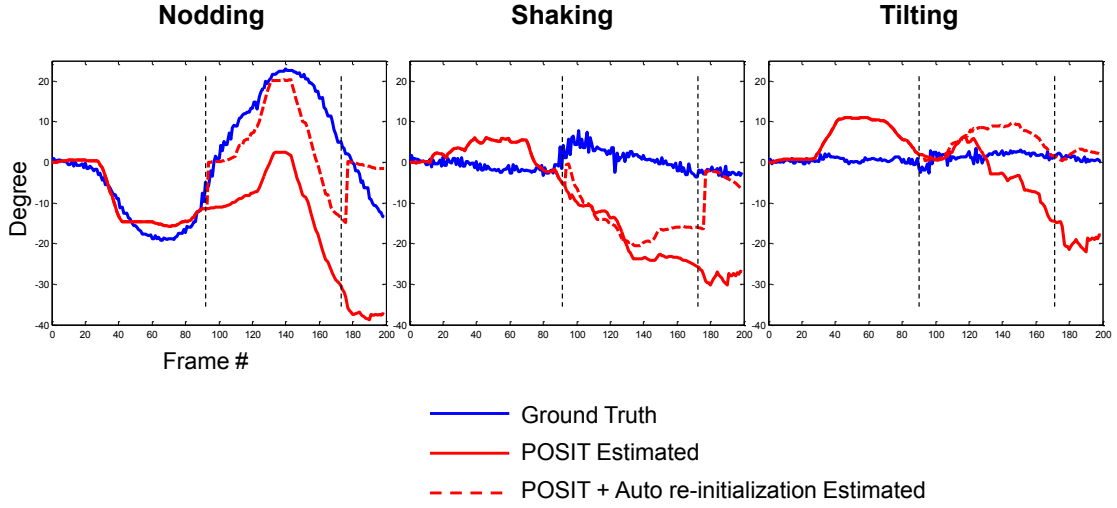


Figure 29: Varying lighting graph. The auto re-initialization occurs twice; around the 90th and 170th frame as shown by the black dotted lines.

The last graph represents Euclidean distance errors of the three head Euler angles with respect to the frame number. The estimated Euler angles are defined as ϕ , θ and γ . The ground truth Euler angles are ϕ_0 , θ_0 and γ_0 . Then, the Euclidean distances are computed using the following equation;

$$E_{dist} = \sqrt{(\phi - \phi_0)^2 + (\theta - \theta_0)^2 + (\gamma - \gamma_0)^2} \quad (41)$$

The blue curve displays the Euclidean distance errors of the original POSIT algorithm, while the red curve displays the errors of the modified POSIT algorithm with auto re-initialization. The left graph is for a uniform video and the right for a varying video. In those graphs, the x-axis is the ground truth.

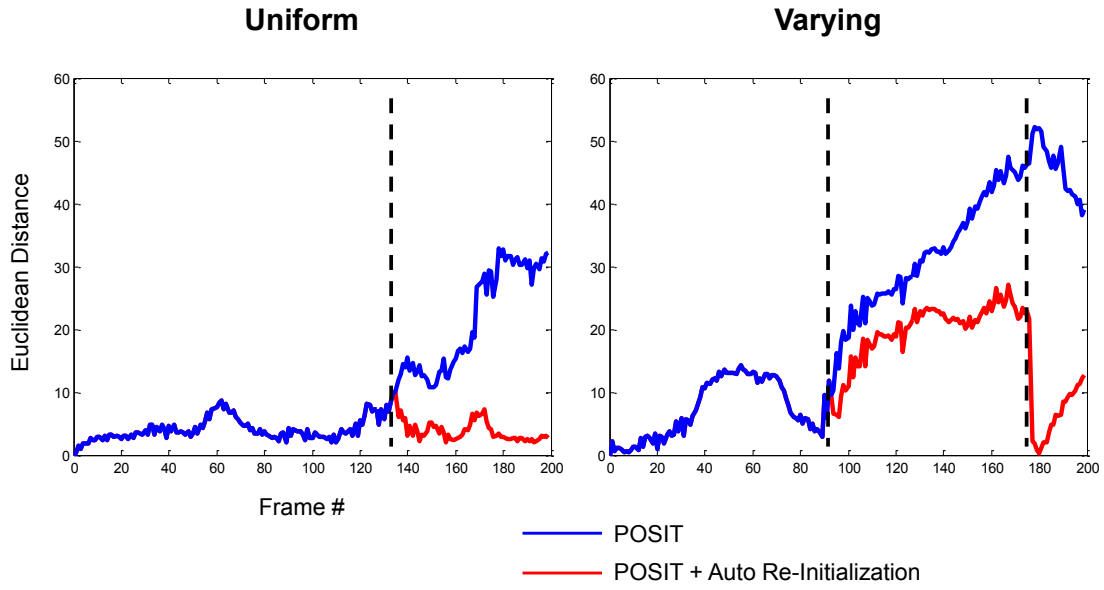


Figure 30: Euclidean distance errors. The auto re-initialization occurs around frame 130 for uniform and around frame 90 and 175 for varying lighting conditions.

By observing those graphs and the behavior of those curves, we deduced some of the system's limitations that needed to be overcome. The first observation is that in figure 30, the curves from uniform illumination are closer to the ground truth than the curves from varying illumination. Therefore, uniform illumination videos give a better performance than varying illumination videos. The second observation is that after an average of 80 frames, the curves go further apart from each other and stop following each other when using the original POSIT algorithm. Also in figure 30, the curves increase exponentially away from the ground truth x-axis after 80 frames when using the original POSIT algorithm. However, when using the proposed method, POSIT + auto re-initialization, the accumulation of such error is prevented.

Table 2. Head Pose Tracking MAE of Different Algorithms

Author	Initialization	Tilting(°)	Shaking(°)	Nodding(°)
La Cascia et al, 2000	AUTO	3.3	6.1	9.8
Choi et al, 2008	MANUAL	3.92	4.04	6.71
Prasad and Aravind, 2010	MANUAL	2.5	3.8	3.6
POSIT	AUTO	5.268	6.001	6.229
POSIT + Auto Re-Initialization	AUTO	3.940	5.161	5.312

The proposed method outperforms the original POSIT algorithm by 1% in average. Also in average, it outperforms other algorithms. The only algorithm that has better results than the proposed algorithm is the algorithm implemented by Prasad and Aravind. However, Prasad and Aravind map features to a 3D model manually while our proposed method does it automatically.

6.2 Pupil Detection Results

The proposed pupil center detection algorithm is robust to illumination changes. As shown in figure 31, hue is less affected by illumination than RGB. Next, the iterative thresholding guarantees the pupil center is within the region of interest despite bright areas over the eye region. The center of gravity is then computed instead of taking simply the center of the final thresholded eye image to localize the true pupil center. In that final stage, we avoid chosen a false center due to illumination variations.

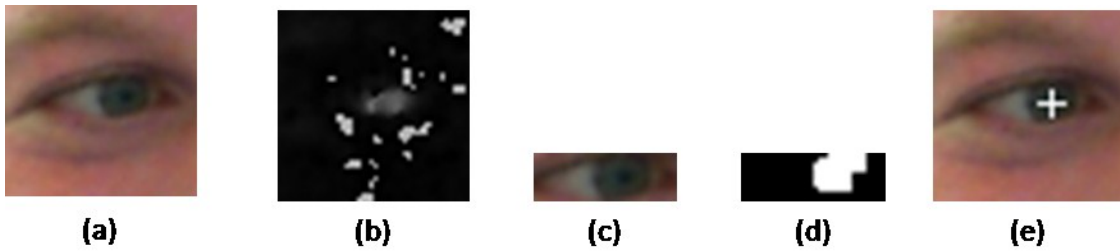


Figure 31: Eye Processing Results: (a) Eye detection, (b) Hue channel conversion, (c) Skin and eyebrow removal, (d) Iterative thresholding, (e) Pupil center localization

The experiments are made using the BioID Face Database. The BioID database was chosen because it provides with images with a variety of illumination, background and face size. The database contains 1521 gray scale images with a 384*286 resolution. The images in the database were created using 23 different test persons in real time conditions. Subjects wearing eyeglasses were not tested.

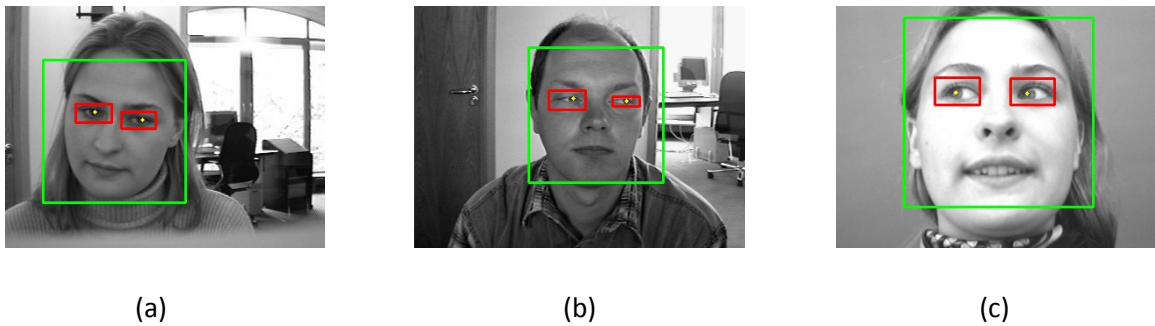


Figure 32: BioID sample images

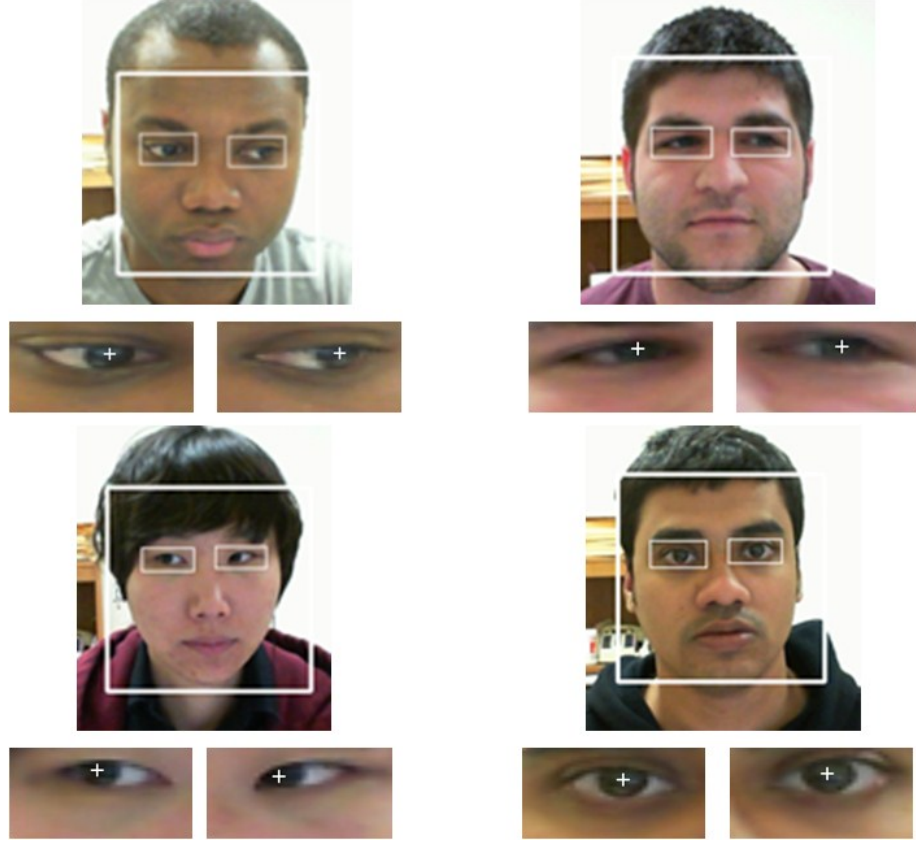


Figure 33: Real Time Detection Results for Different Subjects

The relative error d_{eye} between the expected and the estimated pupil center positions is used to generate the results;

$$d_{eye} = \frac{\max(d_l, d_r)}{\|C_l - C_r\|} \quad (42)$$

where $d_i = \|C_i - \bar{C}_i\|$ and $C_i = \sqrt{x_i^2 + y_i^2}$.

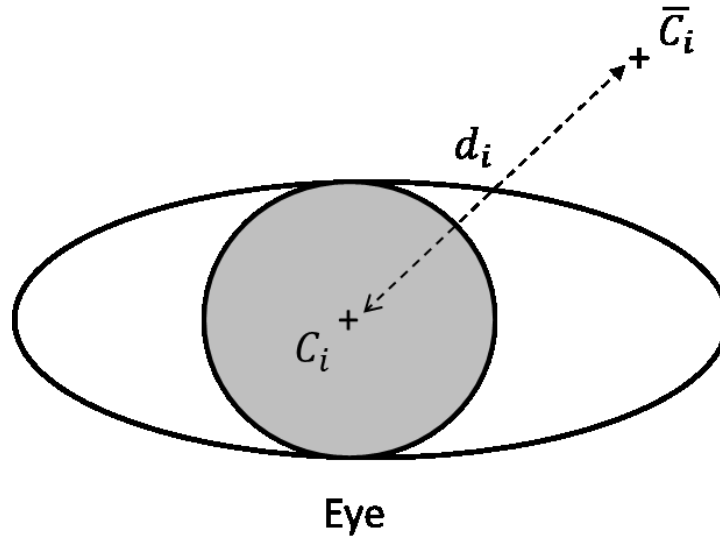


Figure 34: Pupil Center Detection Experiment

The distances d_l and d_r are the left and right distances between the true pupil center positions C_l, C_r and the estimated pupil center positions \bar{C}_l, \bar{C}_r . The subscript i represents the left l or right r side (fig. 34). Note that a relative error of less than 0.25cm is considered as a successful detection so that the detection rate is computed as followed;

$$Detection\ Rate = \frac{Number\ of\ success}{Total} * 100 \quad (43)$$

To validate our results, the algorithm was also tested in real time. It is important for the pupil center detection algorithm to be fast since it only represents the first step of many of the human-computer interaction applications. Our pupil detection speed was about 12 frames per sec using a PC Intel Core 2 Duo 3GHz.

Table 3 shows the results obtained using the proposed algorithm compared with state of art methods that have been applied to the BioID images too.

Table 3. Pupil Center Detection Rate

Author	Algorithm	Detection Rate	Remarks
Jesorsky et al, 2001	AdaBoost	91.8%	
Zhou and Geng, 2004	Generalized Projection Function (GPF)	94.8%	
Asadifard and Shanbezadeh, 2010	Cumulative Distribution Function (CDF)	96.0%	*
Oyini Mbouna and Kong, 2011	Hybrid	97.2%	*

(*) Images with eyeglasses are not considered

6.3 Gaze Estimation Results

The gaze estimation method is tested using a regular webcam which is the Microsoft LifeCam Web camera. The camera is mounted on top of the computer screen and the subject is located at approximately 60cm away. The screen width is 25cm and the screen height is 19cm. The 9 markers are set as shown in figure 35 for testing the system accuracy.

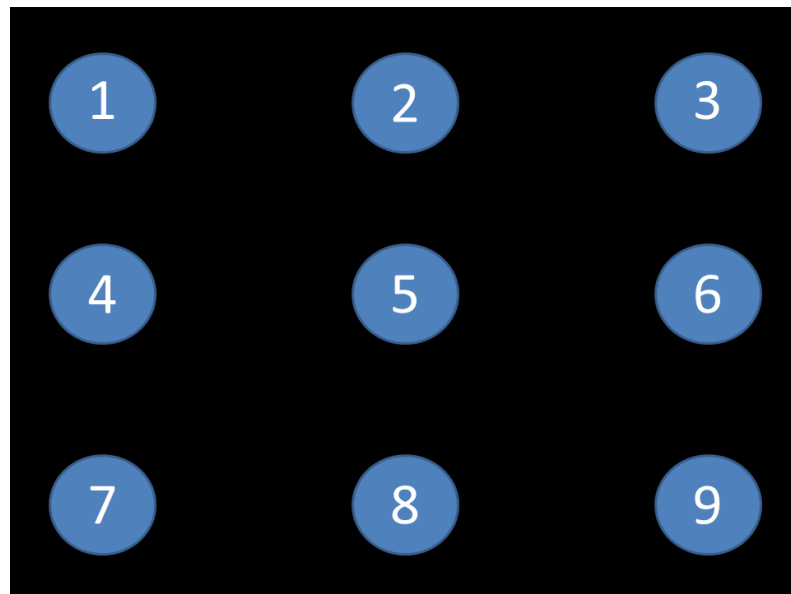


Figure 35: Test Markers. The test markers range from 1 to 9.

Figure 36 shows the different gaze estimation stages. The first stage is face detection. The second phase can be divided into two parts; eyes and corners tracking within the face region. Finally the last phase uses all that information for computing the overall gaze vector.

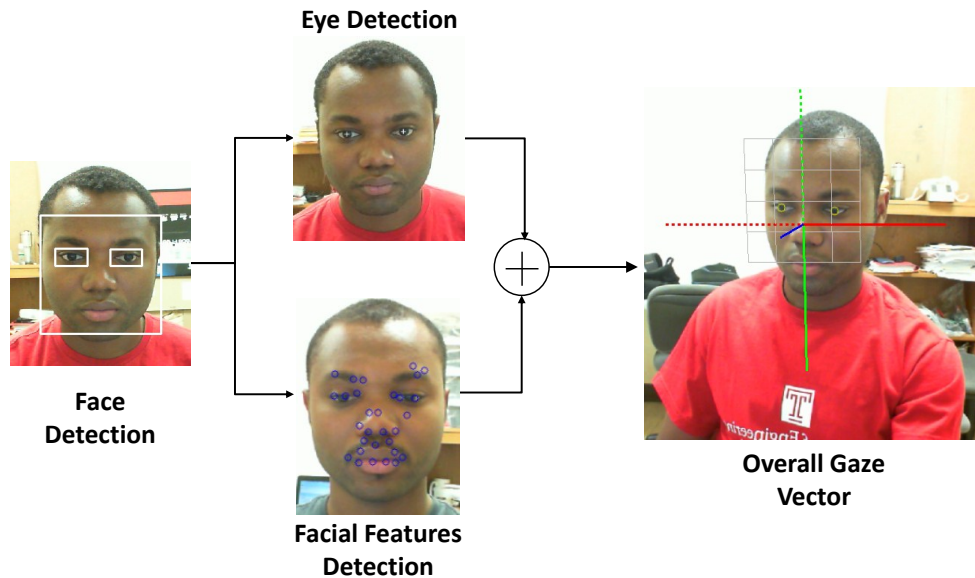


Figure 36: Gaze Estimation Stages. The red and green axes represent the x and y axes respectively. The blue axis is the gaze vector.

The gaze estimation system is tested at different distances away from the camera. From figure 37, we see that the system operates at its best when the user is at 60cm away from the camera. However figure 37 also shows that even though the system's performance decreases, it still works at distances nearer and further from the camera.

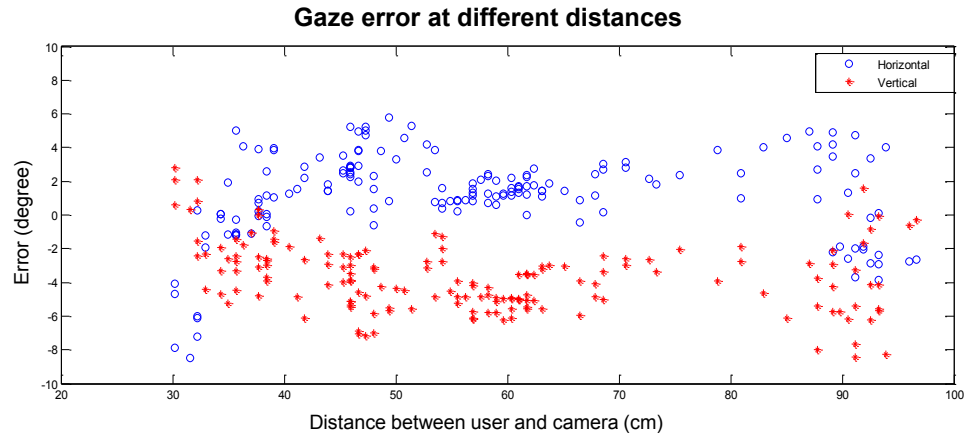


Figure 37: Gaze estimation errors at different distances away from the camera. The user is asked to stare at a random marker (1.3° , 5.8°) on the screen at different distances from the camera.

The gaze estimation system is tested using 10 subjects from different ethnicities staring at different combination of 5 markers starting at the center. Every test was done in real time with pupil and head movements. The gaze estimation database contains 2 sets of videos. The first 10 videos are calibration videos. The second set of 10 videos regroups unseen videos for testing. The accuracy results are shown in Table 4 for ten different users. Table 5 shows how the proposed gaze system performs compared to the state of art.

Table 4. Gaze Estimation Error (Average)

User #	Combination	Horizontal	Vertical
1	5-2-3-6-9	1.9°	2.8°
2	5-4-7-8-5	3.3°	3.8°
3	5-2-4-7-8	2.5°	3.2°
4	5-7-4-2-6	3.1°	3.6°
5	5-4-1-2-3	2.1°	2.5°
6	5-3-7-8-1	4.8°	6.1°
7	5-2-3-6-9	2.4°	2.7°
8	5-6-5-4-1	1.6°	1.9°
9	5-1-9-3-8	4.1°	6.3°
10	5-7-4-2-6	3.3°	4.4°
Average		2.9°	3.7°

Table 5. Gaze Estimation Current State of Art

Author	Method	Gaze Error Average (Horizontal , Vertical)	Distance between user and camera in mm
Yamazoe, 2008	High- Resolution Camera	(5.3°, 7.7°)	2200
Zhu and Ji, 2005	Infrared	(2.0°, 2.45°)	440
Proposed Method	Webcam	(2.9°, 3.7°)	600

The proposed method successfully detects the gaze direction independent of the ethnicity of the subject which implies various eye size (scalable) and skin color (illumination reflected differently) . Additionally, its accuracy is similar to other gaze estimation methods that use special equipment as shown in Table 5.

CHAPTER 7

CONCLUSION

Gaze estimation is a complicated task. Many methods were developed to solve the problem of determining where a person is looking and as a result several solutions were proposed. Selecting the optimal method really depends on the purpose for which the test is performed. The purpose of the system determines the required robustness to illumination changes, accuracy, image resolution, speed, calibration's complexity, convenience and price.

The aim of this thesis is to develop and implement a gaze tracking system that determines where a person, who is seated in front of a computer, is looking at on the screen in real time using a regular webcam. The most important aspect of this project is to keep the system as simple as possible by not using any additional hardware (sensors, infrared LEDs, radar etc...) so that it could work for every personal computer equipped with a single regular webcam in any type of lighting condition environments. The implementation of this system is sectioned in three stages.

The first stage is head pose estimation and tracking. The head is detected in the image using Viola-Jones face detector. The feature corners are extracted within the face region and tracked across successive frames with optical flow. The 2D features are mapped onto a 3D head model and the pose is estimated using the POSIT algorithm with auto re-unitization. The changes in head position relative to the preliminary position during initialization are assessed and the pose is updated across frames. The implemented head pose tracking method is fully automatic and prevents errors due to illumination variations by auto re-initializing. The public BU head tracking database is used to evaluate the proposed head pose tracker. The test results show that the proposed method leads to satisfactory results compared to other popular algorithms.

In the second stage, a pupil center detection method is implemented. A hybrid method that combines appearance and model based characteristics has been used. The Viola-Jones algorithm is used to detect eyes within the face region. The eye is not detected accurately at this point and contains useless information such as skin and eyebrow. Further processing including color space conversion is applied to the image to get rid of extra skin and the image is reduced to the pupil and white of the eye. Then the resulting image is iteratively thresholded by geometric constraints to avoid further errors caused by illumination. The center of the pupil is finally estimated as the center of gravity. The eye pupil center detection method has been tested rigorously with the BioID public database. The proposed method outperforms other methods by 1.2%.

The last stage is estimating the gaze point from the previous two stages. The gaze point is computed as a linear combination of the head pose's Euler angles and pupil center positions. The gaze system is tested on ten different users from different ethnicities looking at 9 different markers on the computer screen. Even though the gaze results cannot be conclusive since the tests are done using a personal gaze database, the experiments conducted have shown that the proposed system is as accurate as other systems. In addition the eye pupil center detection and the head pose tracking, which are the two major parts of the gaze estimation system, have both been tested using public databases.

In conclusion, we can state that a complete eye and head gaze estimation system that delivers accurate results from using only a single normal webcam was implemented. The gaze system had proven to work for persons coming from different ethnicities, at different distances away from camera and for various lighting conditions.

CHAPTER 8

FUTURE WORK

From the results obtained, some extensions to the developed algorithm are proposed in this section.

In the calibration process, the user is asked to look at five points and the values of the parameters α , α' , β and β' are simply averaged. Then those averaged values are used for real time gaze estimation afterwards. An improvement would be to use more calibration points and some type of parameter estimation beforehand such as least squares method to approximate those parameters during calibration.

The gaze estimation results gave some positive feedback but we definitely need to perform additional tests to explicitly identify the problems of the system. We need more statistical significance results to compare the proposed algorithm to others as well.

Finally, the most important problem that needs to be improved is the accuracy of the proposed method. It was expected that the accuracy would not be as high as the commercial methods but that is the price to pay for not using any markers or special equipment. However, there are still ways to make the proposed system more accurate while keeping computation complexity low. One solution is to design a simple 3D eye model instead of using a 2D eye model as we have done so far. The eye is similar to a sphere, so it is only logical to believe a 3D eye model would give overall better results than 2D eye model.

REFERENCES

- [1] Oyini Mbouna Ralph, Seong Kong. Pupil Center Detection with a Single Webcam for Gaze Tracking. In 3rd International Conference on Smart IT Applications, 2011
- [2] Asadifard M. and Shanbezadeh J., "Automatic adaptive center of pupil detection using face detection and CDF analysis", In Proceedings of the IMECS, vol. 1, pp. 130-133, 2010.
- [3] Hansen D. and Ji Q., "In the eye of the beholder: A survey of models for eyes and gaze", IEEE Trans. On PAMI, 32(3): 478-500, 2010.
- [4] Ono Y., Okabe T., and Sato Y., "Gaze estimation from low resolution images," In Proc. IEEE Pacific-Rim Symp. On Image and Video Technology(PSIVT'06), pp. 178-188, 2006.
- [5] Grover D., "Progress on an eye tracking system using multiple near-infrared emitter/detector pairs with special application to efficient eye-gaze communication", Proceedings of the 2nd COGAIN Conference, pp. 21-25, 2006.
- [6] 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, 98:4-24, 2005
- [7] Zhou Z. and Geng X., "Projection functions for eye detection", Pattern Recognition, 37(5): 1049-1056, 2004.
- [8] Ishikawa T., Baker S., Matthews I., and Kanade T., "Passive driver gaze tracking with active appearance models," In Proc. 11th World Congress on Intelligent Transportation Systems, 2004.
- [9] Paul Viola and Michael Jones, "Robust Real-time Object Detection", International Journal of Computer Vision, 57(2): 137-154, 2004.

- [10]Jesorsky O., Kirchberg K., and Frischholz R., "Robust face detection using the Hausdorff distance", In Proceedings of the 3rd AVBPA, LNCS, pp. 90-95, 2001.
- [11]K. Grauman, M. Betke, J. Gips and G.R. Bradski, "Communication via Eye Blinks: Detection and Duration Analysis in Real Time", Proc. IEEE Conf. Computer Vision and Pattern Recognition, vol.1, pp. 1010-1017, 2001.
- [12]J. G. Wang and E. Sung, "Gaze Determination via Images of Irises," Image and Vision Computing, vol.19, no.12, pp. 891-911, 2001.
- [13]Matsumoto Y., and Zelinsky A., "An algorithm for real-time stereo vision implementation of head pose and gaze direction measurement," In Proc. Int. Conf. Automatic Face and Gesture Recognition, pp. 499-504, 2000.
- [14]Baluja S., and Pomerleau D., "Non-intrusive gaze tracking using artificial neural networks," Tech. Rep. CMU-CS, pp. 94-102, 1994.
- [15]Otsu N., "A threshold selection method from gray-level histogram", IEEE Trans. Syst. Man Cybern., 9:62-66, 1979.
- [16]B. H. Pawan Prasad and R. Aravind. 2010. A robust head pose estimation system for uncalibrated monocular videos. In *Proceedings of the Seventh Indian Conference on Computer Vision, Graphics and Image Processing (ICVGIP '10)*. ACM, New York, NY, USA, 162-169.
- [17]Jimenez, P.; Nuevo, J.; Bergasa, L.M.; , "Face pose estimation and tracking using automatic 3D model construction," *Computer Vision and Pattern Recognition Workshops, 2008. CVPRW '08. IEEE Computer Society Conference on* , vol., no., pp.1-7, 23-28 June 2008

- [18]Valenti, R.; Lablack, A.; Sebe, N.; Djeraba, C.; Gevers, T.; , "Visual Gaze Estimation by Joint Head and Eye Information," *Pattern Recognition (ICPR), 2010 20th International Conference on* , vol., no., pp.3870-3873, 23-26 Aug. 2010
- [19]Daniel DeMenthon and Larry S. Davis. 1992. Model-Based Object Pose in 25 Lines of Code. In *Proceedings of the Second European Conference on Computer Vision (ECCV '92)*, Giulio Sandini (Ed.). Springer-Verlag, London, UK, 335-343.
- [20]Murphy-Chutorian, E.; Doshi, A.; Trivedi, M.M.; , "Head Pose Estimation for Driver Assistance Systems: A Robust Algorithm and Experimental Evaluation," *Intelligent Transportation Systems Conference, 2007. ITSC 2007. IEEE* , vol., no., pp.709-714, Sept. 30 2007-Oct. 3 2007
- [21]Vacchetti, L.; Lepetit, V.; Fua, P.; , "Fusing online and offline information for stable 3D tracking in real-time," *Computer Vision and Pattern Recognition, 2003. Proceedings. 2003 IEEE Computer Society Conference on* , vol.2, no., pp. II- 241-8 vol.2, 18-20 June 2003
- [22]M. La Cascia, S. Sclaro, and V. Athitsos, "Fast, reliable head tracking under varying illumination: an approach based on registration of texture-mapped 3D models," *IEEE Transactions on PAMI*, vol. 22, no. 4, pp. 322-336, 2000
- [23]S. Choi and D. Kim, "Robust head tracking using 3D ellipsoidal head model in particle

- Iter," *Pattern Recognition*, vol. 41, no. 9, pp. 2901-2915, 2008
- [24]Hirotake Yamazoe, Akira Utsumi, Tomoko Yonezawa, and Shinji Abe. 2008. Remote gaze estimation with a single camera based on facial-feature tracking without special calibration actions. In *Proceedings of the 2008 symposium on Eye tracking research & applications* (ETRA '08). ACM, New York, NY, USA, 245-250
- [25]Zhiwei Zhu; Qiang Ji; , "Eye gaze tracking under natural head movements," *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on* , vol.1, no., pp. 918- 923 vol. 1, 20-25 June 2005