GazePointer: A Real Time Mouse Pointer Control Implementation Based On Eye Gaze Tracking

Muhammad Usman Ghani, Sarah Chaudhry, Maryam Sohail, Muhammad Nafees Geelani
Department of Electrical Engineering
COMSATS Institute of Information Technology, Lahore, Pakistan.
Email:{ch.usman.ghani, sarah.chaudhry71, maryamsohail22, ngeelani48}@gmail.com

***Abstract*—The field of Human-Computer Interaction (HCI) has witnessed a tremendous growth in the past decade. The advent of tablet PCs and cell phones allowing touch-based control has been hailed warmly. The researchers in this field have also explored the potential of 'eye-gaze' as a possible means of interaction. Some commercial solutions have already been launched, but they are as yet expensive and offer limited usability. This paper strives to present a low cost real time system for eye-gaze based human-computer interaction.**

I. Introduction

Innovative and efficient techniques of HCI are being developed rapidly. It is an active research field of many experts. This paper concentrates on a human computer interaction application based on eye-gaze tracking. Human eyes carry much information which can be extracted and can be used in many applications i.e. Computer Interaction. Eye gaze reflects a person's point of interest. Eye gaze tracking is aimed to keep track of human eye-gaze. "Eye movements can be captured and used as control signals to enable people to interact with interfaces directly without the need for mouse or keyboard input" [1]. This can be achieved by employing computer vision and image processing algorithms.

Technique explained in the paper is non-invasive and user-friendly, as it does not require a complex hardware or wires. Moreover, it does not have any physical interaction with the user. A cheap solution is provided for gaze-tracking. A built-in web-cam in laptop is used as a capturing device. A software based solution is proposed for controlling mouse pointer using 'eye gaze'. It is a natural and efficient way of interaction with the computer. Mostly the methods of interaction available are complex and cumbersome. Using this method, for controlling mouse pointer increases the interaction efficiency and reduces complexity. This technique is a special boon for disabled persons, such as spinal cord injured, or paralyzed patients. These patients are entirely dependent on assistance. Currently, disabled people usually type on the computer keyboard with long sticks that they hold in their mouth [2], but the technique being presented is a benefaction for handicaps to help them be independent in their lives. Giving them a chance to work, socialize, and entertain in their lives.

The remainder of this paper is structured as follows. Related research work is presented in Section II. Section III presents an overview of proposed system. Proposed Eye Gaze Tracking algorithm is described in Section IV. Section V includes experimental results. Section VI presents conclusion of this research work and future research directions.

II. Related Research

A number of eye-gaze tracking techniques are already available. Some researchers performed eye gaze tracking using the Electro-Oculography tracking technique. It takes advantage of the fact that an electrostatic field exists around the eyes which changes with eye ball movement and these small differences can be recorded with help of electrodes placed on the skin around eye. The use of electrodes makes this technique troublesome and not well-suited for everyday use, an application can be found in [3]. A detailed review of Electro-Oculography tracking technique is presented in [4].

Various methods have been developed based on tracking contact lenses. These systems perform very well, but they are invasive, uncomfortable, and often require a topical anesthetic. "Matin and Pearce (1964) developed a scleral contact lens system that uses a pair of noncoplanar 4-mm-diameter mirrors embedded in the surface of the lens on opposite sides of the pupil, their system has a resolution of 0.00028 within a range of 10 for all three dimensions" [5]. A laser-based eye-tracking system is proposed in [5], it falls under the category of head-mounted eye tracking systems, which is not favorable for everyday use. Other example of head mounted trackers are [6], [7].

Video-based systems have also been reported in literature. In past, low processing power of computing devices limited the use of video-based techniques for Eye Gaze Tracking as computing devices did not had the potential to provide real time eye gaze tracking operation. In last few decades, high processing power computing devices have been made available which motivated the researchers to develop video-based solutions for Eye Gaze Tracking. Several video-based system methods have been reported in literature, a few of them are Corneal Reflections [2], Purkinje Image Tracking [8]. This paper also presents a video-based eye gaze tracking system and attempts to take advantage of built-in web-cam in laptop for eye gaze tracking. It presents a solution using computer vision and image processing algorithms. This is an attempt to report a low cost eye gaze tracking system for Human-Computer Interaction.

III. System Description

An illustration of setup for GazePointer is presented in Figure 1. The system consists of a laptop built-in web-cam which takes live image frames and GazePointer application processes the frames to extract user's Point of Gaze (PoG).

978-1-4799-3043-2/13/$31.00 2013 IEEE

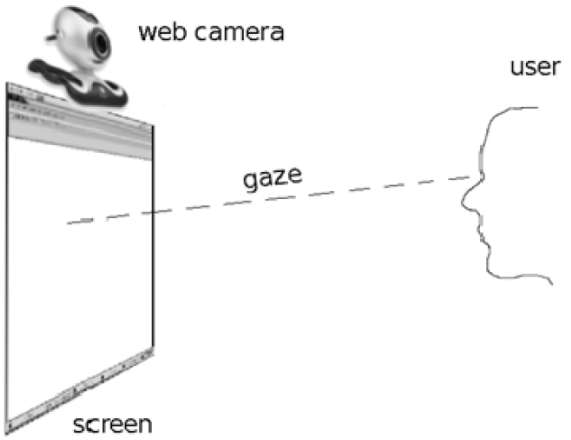

Fig. 1. Setup for Eye Gaze Tracking [9]

Proposed system performance was analyzed in different scenarios and some limitations were defined, which are as follows. User head should be at the same altitude as the web-cam. Distance between user's eyes and web-cam should be in the range of 20-75 cm. This system can't be used with glasses on, lighting conditions should be good and head movements are not allowed. Currently system is only tested for frontal faces.

IV. Eye Gaze Tracking Algorithm

Overview of Eye gaze tracking algorithm is presented in Figure 2. It consists of three major modules: (i) Facial features extraction; (ii) Eyes features detection and (iii) Point of Gaze calculation.

Algorithm presented in this paper performs operations on grayscale images. Camera captures BGR or YCbCr color space images, depending upon default settings. As a first step BGR $\rightarrow$ grayscale color space conversion is performed. Basic image pre-processing procedures are performed at each stage of algorithm. Histogram equalization is applied on grayscale images to normalize contrast in acquired image. It attempts to equalize the image histogram by adjusting pixel intensities in accordance with histogram [10]. For face detection, a machine learning based approach is used, Object detection algorithm proposed in [11]. This technique employs a Haar-features based approach for object detection, which makes the rapid and accurate object detection possible. Eye patch extraction can also be performed using same object detection algorithm described in [11]. For pupil detection, extracted eye patch must be smoothed to avoid false detections. Pupil detection technique being used is Hough Circle Transform (HCT) [12]. For image binarization, edge detection approach is used. Eye region being used to trace the Test Area is to be detected, for this purpose a simple calibration technique is designed, which is explained later in this section. After features detection, a simple Point of Gaze calculation algorithm is designed which systematically interrelates the detected feature points to result in a precise PoG calculation.

A. Facial Features Extraction

This paper intends to present an eye gaze tracking algorithm and facial features detection i.e. face and eyes extraction is an important task in this regard.

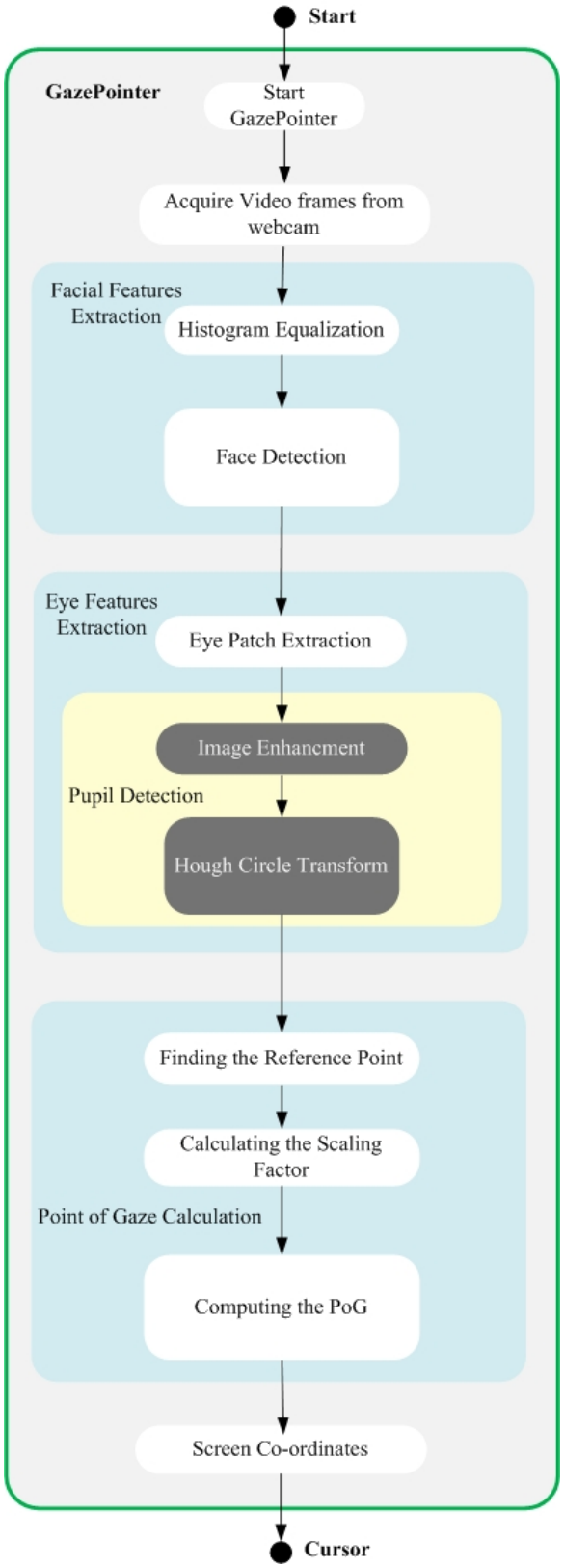

Fig. 2. Overview of GazePointer algorithm

1) Histogram Equalization: Histogram equalization is a technique for contrast normalization by adjusting image intensities in accordance with image histogram [10]. It is a contrast enhancement procedure which takes benefit of histogram data and attempts to equalize the histogram. For performing facial features detection, histogram equalization gives benefit of better performance in terms of accuracy.

2) Face and eye patch extraction: Face detection is a classification problem i.e. classify acquired image into face and non-face regions. A rapid and robust object detection algorithm is employed to perform face and eyes extraction, proposed in [11]. It follows a machine learning based approach. Haar-features based classifiers can be trained for several objects detection i.e. face, eyes. These classifiers are trained for false positive and false negative samples as well. A set of simple features are obtained from training data. Haar- features are calculated by the difference between dark-region and light-region pixel values. A threshold value is fixed at learning stage i.e. feature is said to be present if difference value comes out to be greater than the value fixed as threshold.

Face and eyes detection is a complex procedure, require much computations to solve this classification problem. For this purpose, acquired images are down-sampled, face is detected and face co-ordinates are mapped to original image using simple calculations. Same practice is exercised for eye patch extraction to reduce the computation time; this approach has proved to be effective in order to achieve real-time processing of the frame.

B. Eye Features Detection

Eye gaze tracking is a complex problem; it needs to acquire a number of facial and eye features to compute Point of Gaze (PoG). In this regard, first problem is to identify necessary and sufficient eyes features which can result in an accurate PoG calculation. Two important eye features necessary to compute PoG were identified, which are (i) Pupil and (ii) Eye Corners. This section presents the techniques utilized for these eye features extraction.

1) Pupil Detection: Pupil is the central and focusing part of eye, located at center of iris. Light enters into eye through pupil, and finding the position of pupil is principal point of interest in proposed technique. Eye gaze projection is based upon the relative displacement of pupil from center of eye. Pupil needs to be detected to project user's eye gaze in the 'Test Area'. The first step in this regard, is to detect the iris from the frames captured with web-cam, and then pupil can be found, as it is situated at center of iris. Iris is a circular region and can be detected using the very commonly used Hough circle transform technique [12]. Hough Circle Transform takes a binary image as an input and detects circle in it. The quality of the image needs to be good to extract every possible information from it. First, the input image is enhanced for good quality and then "Hough Circular transform" is applied on it.

To enhance image, smoothing is applied which in-effect reduces noise. For this purpose, grayscale image is passed through Gaussian blurring filter. The technique used for iris detection is 'Hough Circle Transform'. Canny edge detection filter [10] is applied on enhanced grayscale image to compute a binary image. HCT is then applied on binarized image to compute Pupil points in both eyes.

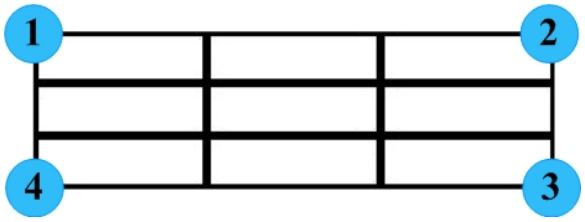

Fig. 3. 4-Point Simple Calibration

2) Eye Corners Extraction: Eye corners are such an important eye feature that a lot of information about eyes can be extracted using eye corner locations. Once eye corner locations are known, they can be used to estimate eye width, eye height and most importantly this information can be used to locate center of eye.

Another fact is established through several experiments that eye corners extraction is not suitable, to get to know about movable area for iris in eye to trace the 'Test Area'. For this purpose, after analysis it was concluded that computer is not that intelligent to discover those boundary points in eye, user should feed some information at start and some procedures must be implemented to use this information in intelligent way to extract features i.e. movable area for iris, center of eye etc. To meet this requirement, a simple 4-point calibration technique is designed.

A 4-point calibration algorithm is designed for this system, shown in Figure 3. Idea behind designing such calibration algorithm is, it can be helpful to calculate eye region which will be used to scan the 'Test Area'. This simple algorithm allows the user to look at all corner points in free mode. Here 'free mode' suggests that user is allowed to stay at a corner point for an arbitrary time duration. This idea helps the system to reduce errors occurred due to erroneous pupil detections during calibration.

C. Point of Gaze Calculation Algorithm

Point of gaze can be referred to as the point of interest of user in 'Test Area'. User's point of interest i.e. PoG can be calculated by extracting eye patch and some important eye features. Important eye features sufficient to calculate PoG has been identified and discussed in earlier sections.

It is the most important and critical stage of algorithm as it involves using already found feature points to calculate PoG. This stage must effort to compensate the errors occurred at detection stage.

1) Finding the Reference Point: It is absolutely in-feasible to perform PoG calculations without a reference point. It will be required to translate pupil movements in eyes into cursor movements on screen. 'Center of Eye' can act as a desired candidate for reference point. Eye's movable region has already been computed during calibration stage, a simple averaging of x and y co-ordinates can result in Center of Eye calculation. This concept is illustrated in Equation 1 and Equation 2.

$$COE_x = \frac{TopRightCorner_x + TopLeftCorner_x}{2} \quad (1)$$

$$COE_y = \frac{TopRightCorner_y + BottomRightCorner_y}{2} \quad (2)$$

Where, COE_x and COE_y denote x and y coordinates of center point of eye's movable region respectively. $TopRightCorner$, $TopLeftCorner$, $BottomRightCorner$ and $BottomLeftCorner$ construct a rectangular region which represent eye's movable region.

2) Calculating the Scaling Factor: In this step the cursor movements and pupil movements were interrelated i.e. it was to be found that how many pixels a cursor will traverse for a single pixel movement of the pupil. For this calculation, width and height of eyes were associated with the width and height of the screen. Screen width and height is constant, but eye's movable region width and height is subject to change in different scenarios. Eye's movable region width and height can be computed using Equation 3 and Equation 4.

$$w_{eye} = TopLeftCorner_x - TopRightCorner_x \quad (3)$$

$$h_{eye} = TopRightCorner_y - BottomRightCorner_y \quad (4)$$

Where, w_{eye} and h_{eye} represent width and height of eye's movable region respectively. Now scaling factor is to be computed for x and y coordinates with help of Equation 5 and Equation 6.

$$R_x = \frac{w_{screen}}{w_{eye}} \quad (5)$$

$$R_y = \frac{h_{screen}}{h_{eye}} \quad (6)$$

Where, w_{screen} and h_{screen} denote width and height of 'Test Area'. R_x and R_y represent scaling factor for x and y coordinates.

3) Computing the PoG: This is the final step of PoG calculation as well as GazePointer algorithm. This stage will realize the significance of reference point. It translates the pupil movements in eyes into cursor movements in 'Test Area'. Taking assumption that reference point in eye corresponds to center point in 'Test Area', pupil movements can be simply translated into cursor movements using Equation 7 and Equation 8.

$$PoG_x = \frac{w_{screen}}{2} + R_x \times r_x \quad (7)$$

$$PoG_x = \frac{h_{screen}}{2} + R_y \times r_y \quad (8)$$

Where, PoG_x and PoG_y represent x and y coordinates of Point of Gaze respectively and r_x denotes pupil distance in x direction from reference point and r_y denotes pupil distance in y direction from reference point and they can be computed by using Equation 9 and Equation 10.

$$r_x = COI_x - COE_x \quad (9)$$

$$r_y = COI_y - COE_y \quad (10)$$

Where, COI represents pupil location. Figure 4 illustrates this procedure.

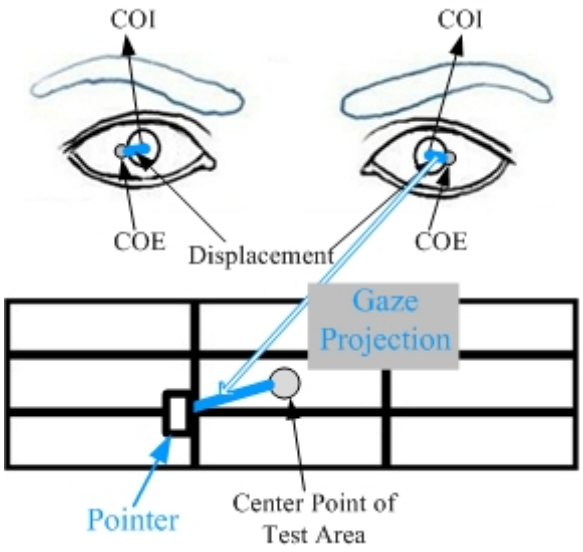

Fig. 4. Computing Point of Gaze using Reference Point

TABLE I. EXPERIMENT 1 ACCURACY RESULTS

Extracted Features	Accuracy
Face Detection	100%
Eye Patch Extraction	100%
Pupil Detection	87%

V. EXPERIMENT RESULTS

Two experiment setups were designed to test the developed system.

A. Experiment 1: GazePointer GUI

A 'Graphical User Interface' is designed to demonstrate the results. First area show the extracted eye-patch from the captured frames and second area indicates HCT results. The portion under it represents the 'Test Area' for movement of mouse pointer. A small GazePointer is shown in the Figure 4 which shows the projection of eye movements.

1) Experiment 1 Results: In this section, results of experiment 1 are presented. Accuracy results of experiment 1 are presented in Table I. Overall accuracy of 87% is being reported for experiment 1. Experiment 1 efficiency results are presented in Table II. Overall time for processing of 43 mS per frame is being reported, which means GazePointer system works at 23 fps.

2) Experiment 1 Results Discussion: Face detection provided satisfactory results. Accuracy did not drop with changing lighting conditions, backgrounds, and distance. This implementation detects frontal faces only. Its accuracy remained 100% when tested on a limited dataset of 223 frames. A few

TABLE II. EXPERIMENT 1 ACCURACY RESULTS

Extracted Features	Processing Time (mS)
Face Detection	13-15
Eye Patch Extraction	10-15
Pupil Detection	6-10
PoG Computation	2-3

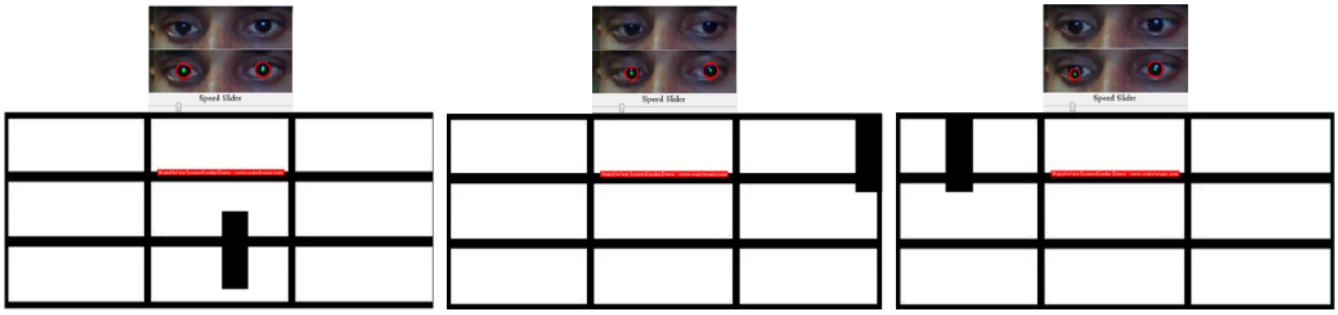

Fig. 5. PoG calculation results in different frames

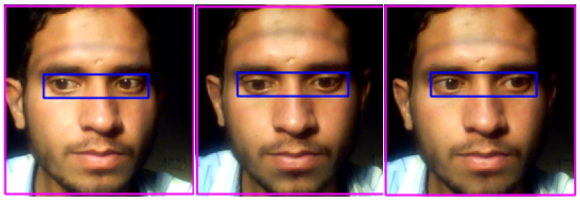

Fig. 6. Face and Eyes Detection Results

Fig. 7. Extracted Eye Patch in different frames

samples of face detection results are presented in Figure 6. GazePointer is assumed to be a system working with real-time constraints. Captured frames were down sampled 5 times to reduce the computation time. Before re-sizing, it took 500-600 mS per frame and it reduced to only 13-15 mS per frame.

Eyes detection implementation was done accurately in almost every case when face was detected. This implementation was also tested on a limited dataset of 223 frames and it resulted in 100% accuracy. Eyes detection results are given in Figure 7. This implementation resulted in a large computation time. The computation time was reduced by limiting the region of interest to detected face only. Then this face region was down sampled 2 times. Computation time was reduced to 10-15 mS per frame after modifying the implementation.

HCT implementation resulted in a few mS processing time. Initial algorithm resulted in lots of false detections. It resulted in accuracy of 80% when tested on a limited dataset of 223 frames. Pupil detection results are given in Figure 8. The rate of false detections was decreased by applying a threshold between previous frame pupil location and present frame pupil location. Thus, accuracy improved to 87%. Point of Gaze Calculation resulted in satisfactory results when tested on a limited dataset PoG calculation results are presented in Figure 5. Projections followed user's eye-gaze. False detections were involved because pupil detection results were not 100% accurate.

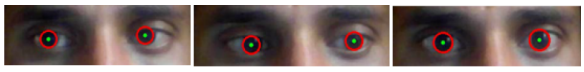

Fig. 8. Pupil detection in different frames

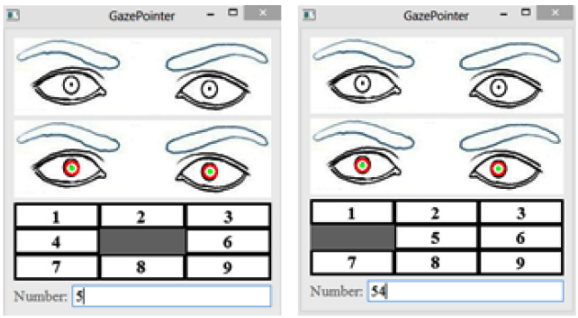

Fig. 9. Experiment 2 Results in different frames

TABLE III. EXPERIMENT 2 ACCURACY RESULTS

Extracted Features	Accuracy
Pupil Detection	100%
PoG Computation	100%

B. *Experiment 2: Test Bench*

A simple Test Bench model was designed to test the accuracy of this system. Artificial eyes instead of human eyes were used. Application prompts user to input a number. Iris moves in accordance with the number entered. Algorithm processes these iris movements and projects the eye gaze in 'Test Area'.

1) Experiment 2 Results: Accuracy results of experiment 2 are presented in Table III. Overall accuracy of 100% is being reported for experiment 2. A few results from experiment 2 are shown in Figure 9. Time efficiency results of experiment 2 are shown in Table IV. Overall processing time of 13 mS per frame is being reported for experiment 2.

VI. CONCLUSION

In this paper a computer vision algorithms based solution is implemented. An attempt has been made towards development of low cost, real-time solution for eye gaze tracking. There are many applications of eye gaze tracking, for instance in HCI, appliances control, usability studies and in advertising

TABLE IV. EXPERIMENT 2 EFFICIENCY RESULTS

Extracted Features	Processing Time (mS)
Pupil Detection	6-10
PoG Computation	2-3

effectiveness. Accuracy for features extraction algorithms depends upon image quality and lighting conditions. Algorithm performance drops down in poor lighting environment. Computer Vision algorithms are employed for features detection and they don't perform well in bad lighting. PoG is accurately calculated provided detections are correct. Pointer size is large due to low web-cam resolution and small 'Test Area' size.

To improve the projection results, image quality must be enhanced. Better image quality would improve accuracy of computer vision algorithms. Sophisticated Pre-Processing algorithms should be introduced to compensate lighting variations and web-cam resolution should also be increased to decrease the pointer size. A feature describing head-posture must also be introduced, it will allow the user to move-freely while interacting with system. Introducing the concept of gaze estimation along with gaze projection will be beneficial because it will improve gaze projections drastically. The idea of gaze estimation promises to learn from usage statistics and infer gaze projections. Particle Filters can be used to implement gaze estimation because they are quite simple and has resemblance with problem of gaze estimation.

ACKNOWLEDGMENT

The authors would like to thank Mr. Muhammad Adnan Siddique for his continuous guidance during this research work.

REFERENCES

[1] Alex Poole and Linden J. Ball, "Eye Tracking in Human-Computer Interaction and Usability Research: Current Status and Future Prospects," in *Encyclopedia of Human Computer Interaction (30 December 2005) Key: citeulike:3431568*, 2006, pp. 211-219.

[2] D. H. Yoo, J. H. Kim, B. R. Lee, and M. J. Chung, "Non-contact Eye Gaze Tracking System by Mapping of Corneal Reflections," in *Fifth IEEE International Conference on Automatic Face and Gesture Recognition (FGR02)*, 2002, pp. 94-99.

[3] Rafael Barea, Luciano Boquete, Manuel Mazo, and Elena Lpez, "System for assisted mobility using eye movements based on electrooculography," *IEEE TRANSACTIONS ON NEURAL SYSTEMS AND REHABILITATION ENGINEERING*, vol. 10, no. 4, pp. 209-217, DECEMBER 2002.

[4] H. Singh and J. Singh, "A Review on Electrooculography," *International Journal of Advanced Engineering Technology*, vol. III, no. IV, 2012.

[5] K. Irie, B. A. Wilson, and R. D. Jones, "A laser-based eye-tracking system," *Behavior Research Methods, Instruments, & Computers*, vol. 34, no. 4, pp. 561-572, 2002.

[6] P Ballard and George C. Stockman, "Computer operation via face orientation," in *Pattern Recognition, 1992. Vol.I. Conference A: Computer Vision and Applications, Proceedings., 11th IAPR International Conference on*, 1992, pp. 407-410.

[7] T. Horprasert, Y. Yacoob, and L.S. Davis, "Computing 3-D head orientation from a monocular image sequence," in *Second International Conference on Automatic Face and Gesture Recognition*, 1996, pp. 242-247.

[8] K. Arai and M. Yamaura, "Computer Input with Human Eyes-Only Using Two Purkinje Images Which Works in a Real-Time Basis without Calibration," *CSC Journals*, vol. 1, no. 3, pp. 71-82, 2010.

[9] D. Back, "Neural Network Gaze Tracking using Web Camera.," Linkping University, MS Thesis 2005.

[10] R. Gonzalez and R. Woods, *Digital Image Processing*, 3rd ed.: Pearson Education, 2009.

[11] P. Viola and M. Jones, "Rapid Object Detection using a Boosted Cascade of Simple Features," in *COMPUTER VISION AND PATTERN RECOGNITION*, 2001.

[12] C. Kimme, D. Ballard, and J. Sklansky, "Finding circles by an array of accumulators," in *Communications of the Association for Computing Machinery*, 1975, pp. 120122.

[13] H. Hua, P. Krishnaswamy, and J. P. Rolland, "Video-based eyetracking methods and algorithms in head-mounted displays," *Optics Express*, vol. 1, no. 10, pp. 4328-4350, 2006.