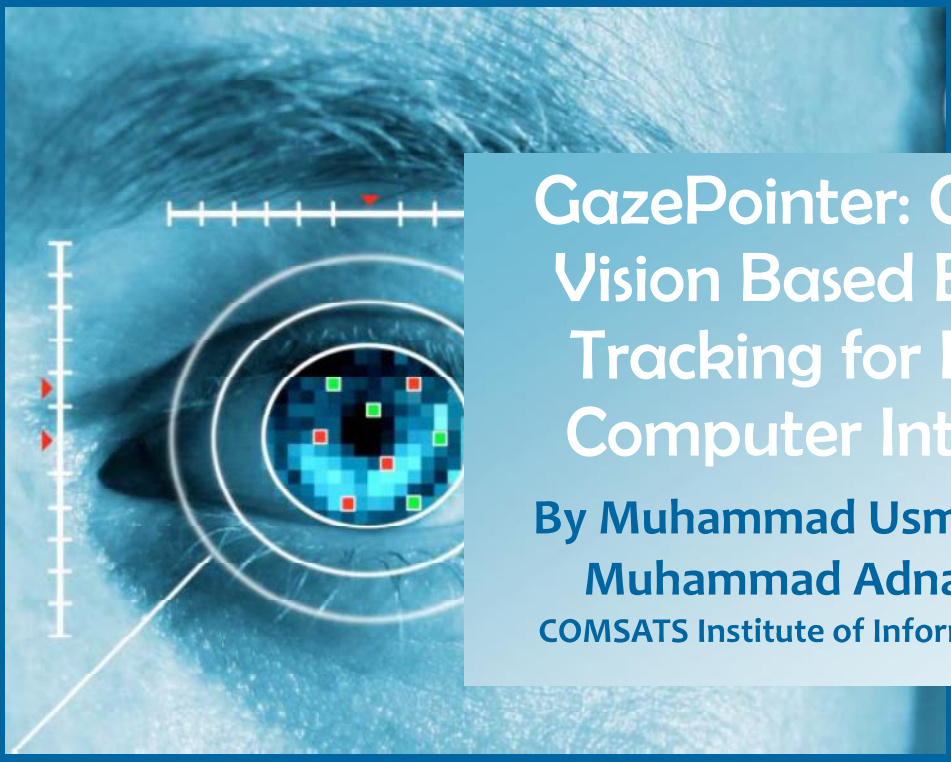

GazePointer: Computer Vision Based Eye Gaze Tracking for Human-Computer Interaction

By Muhammad Usman Ghani and Muhammad Adnan Siddique
COMSATS Institute of Information Technology

Abstract

Human-Computer Interaction (HCI) researchers have been working on the development of new interaction techniques. As an alternative to touch based computer interfaces, HCI research has investigated eye gaze as a means of interaction. It is proposed that eye gaze can offer direct means of interaction, since it avoids the involvement of a hardware mouse, hand gestures or finger touch. This paper endeavors to demonstrate a low-cost real-time computer vision based eye gaze tracking algorithm developed for HCI applications.

Introduction

Researchers are working to develop innovative and efficient interaction techniques for computer users, e.g. Human-Computer Interaction (HCI). One alternative to touch-based interaction is eye gaze tracking [1]. Human eye movements carry a significant amount of information, especially as related to the person's point of interest. This information can be extracted and manipulated for use in applications such as HCI. If the computing device can detect eye gaze of the interacting person, little further intervention is required to interact with the machine. The mouse pointer automatically follows the person's point of interest. An eye gaze tracking based HCI technique could speed up the interaction processes.

Researchers have been working on conducting studies about eye gaze tracking and developing systems capable of eye gaze based interaction. Many companies have developed proprietary solutions and have made them available in the market; but these solutions offer limited usability, involve complex hardware, do not provide user-friendly human-computer interaction, and are very costly; which make them out of reach for domestic users.

GazePointer investigates the potential of eye gaze in HCI applications. A computer vision based tracking algorithm was developed using eye gaze that can provide low-cost, real-time human-computer interaction. The idea behind GazePointer was to develop an interface that can enhance user experience of interaction. A person's point of gaze was tracked using a webcam system and controlled mouse pointer movements accordingly. This paper reports related literature, overviews the system and its implications, describes the Eye Gaze Tracking algorithm, and discusses experimental results and future research.

Literature Review

Several research studies have been conducted on eye gaze tracking technique; applying various approaches to achieve gaze tracking. An eye gaze tracking approach

based on Electro-Oculography (EOG) takes advantage of the electrostatic field that exists around eyes and varies with eye-ball movements. These minor variations can be captured by placing electrodes around eyes. The technique becomes cumbersome due to usage of electrodes and is not suited well for everyday usage. Such an application is presented in [2], and a detailed review of EOG based techniques is demonstrated in [3].

Contact lens tracking based techniques have been developed and offer satisfying performance but they are uncomfortable and invasive. A lens based tracking technique is discussed in [4]. Head-mounted eye gaze tracking techniques are also proposed, but they are not friendly for general purpose use. Head-mounted tracking approaches have been explored in [5, 6].

Video-based techniques can also be applied for eye gaze tracking, but computing devices with low processing capacities restricted video-based solutions. Advent of machines with high processing power has made it possible to provide real-time interaction using video-based techniques. Various video-based tracking approaches has been proposed recently. Purkinje Image Tracking [7] and Corneal Reflections [8] are examples of such systems. This paper presents a low-cost, real-time interaction technique using computer vision algorithms.

System Description and Significance

This research work attempts to address the very complex problem of eye gaze tracking, using basic and simplified algorithms, in order to make it low-cost, user-friendly and real-time. The problem statement of the study was defined as "design and implement a simplified technique using computer vision algorithms to provide low-cost and real-time human-computer interaction based on eye gaze tracking that will only require a computing device and a web-cam". This study also suggests that eye gaze has the potential of offering an efficient means of interaction and could be especially useful for handicapped people.

The approach could have wide applications; eye gaze tracking has potential applications ranging from home appliances control to aviation, from neurosciences to intelligent tutoring systems, and from Psychology to advertising. The proposed technique offers non-invasive gaze-tracking and uses a software only approach. The proposed system requires only a computing device and makes use of built-in web-cam which captures image frames. Images are processed through GazePointer and provide smooth, real-time interaction after detecting point of gaze.

Eye Gaze Tracking Algorithm

The GazePointer tracking algorithm is briefly illustrated in Figure 1. The algorithm has three main components: facial features extraction, eye features computation, and point of gaze calculation. The algorithm processes grayscale images only, therefore, image color space conversion may be performed, if required. The histogram of grayscale images is then equalized [9] for contrast normalization. A face detection technique based on Viola and Jones' object detection algorithm [10] is then applied to detect the face in an acquired image. Eye features are then computed by Viola and Jones' object detection algorithm [10] for eye patch extraction.

Eye patch extraction and face detection require a complex process involving much computation. In order to make GazePointer work in real time, initial frames are firstly down sampled and then face and eye patch co-ordinates are mapped to the original image after face and eye patch extraction. This procedure drastically improved the computational efficiency of the proposed algorithm.

Gaze tracking is a complicated problem, in which a sufficient number of facial and eyes features must be computed and fed to the Point of Gaze (PoG) computation method. After a thorough analysis and study, two most important features were identified, pupil and eye corners. The pupil being the focusing and central part of the eye gives cues about the user's point of interest. Eye corners are also important to identify as they give information about the area of eye being used to trace the Test Area.

The extracted eye patch is further processed before pupil detection with a smoothing filter. This is done to reduce false positives for pupil detection. Hough Circle Transform (HCT) [11] is then applied to detect the pupil. HCT is applied on a digitized version of the extracted eye patch, using a canny edge detector [12]. It was observed that the complete area of eye is not used to trace the Test Area;

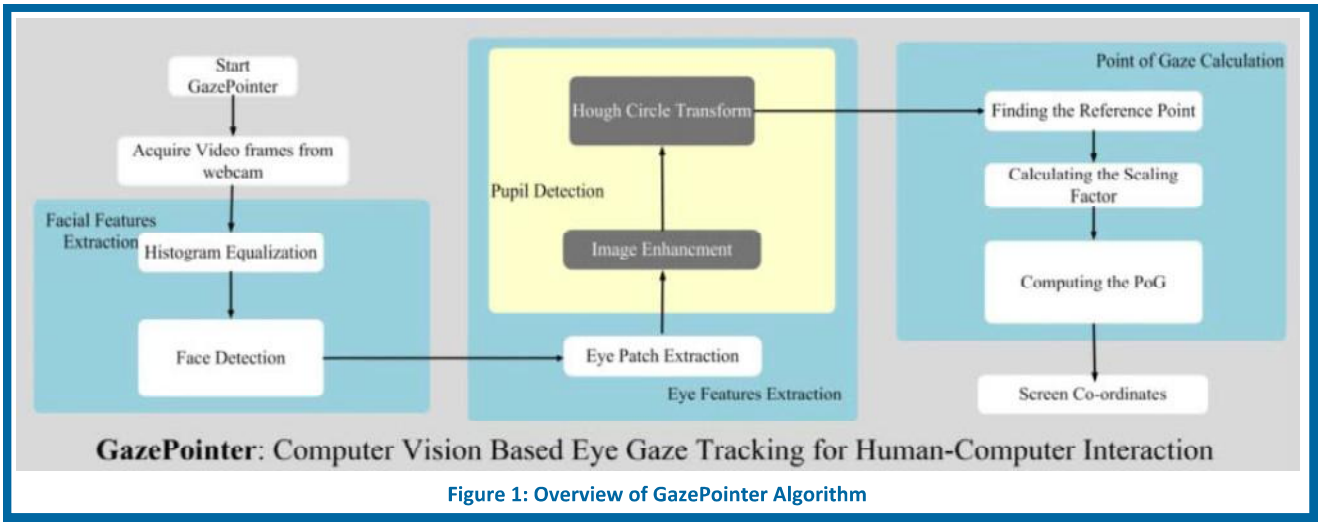

Figure 1: Overview of GazePointer Algorithm

therefore, a simple calibration technique was developed to compute that region of eye.

PoG denotes the user's point of interest in the Test Area. The PoG computation algorithm takes detected features and outputs the user's Point of Gaze in the Test Area with respect to a reference point. Four corner points were already computed during the calibration stage; a simple average of these corner points output 'Center of Eye'. The 'Center of Eye' is computed using Equation 1 and Equation 2 and it is taken as reference point.

$$COE_x = \left(TopRightCorner_x + TopLeftCorner_x\right) \Big/ 2 \quad (1)$$

$$COE_y = \left(TopRightCorner_y + BottomRightCorner_y\right) \Big/ 2 \quad (2)$$

Here, COE_x and COE_y represent x and y co-ordinates of Center of Eye respectively. $TopRightCorner_x$, $TopRightCorner_y$, $TopLeftCorner_x$ and $BottomRightCorner_y$ form movable region of eyes that trace Test Area. The next step requires estimation of scaling factor, which would be applied to pupil movements in order to translate pupil movements into GazePointer movements in Test Area. It needs height (h_{eye}) and width (w_{eye}) of eye's area used to trace Test Area, these can be computed using Equation 3 and Equation 4. X (R_x) and Y (R_y) co-ordinates of Scaling factor can be computed using Equation 5 and Equation 6. Here, h_{screen} and w_{screen} represent height and width of Test Area.

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

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

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

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

PoG computation takes advantage of reference point 'Center of Eye', it computes the displacement of pupil from COE and translates these pupil movements in eyes into GazePointer movements in Test Area. PoG translation can be performed using Equation 7 and Equation 8.

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

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

Here, PoG_x and PoG_y denote x and y co-ordinates of PoG; r_x and r_y represent x and y co-ordinates of pupil displacement in eye (from reference point). r_x and r_y can be calculated by applying Equation 9 and Equation 10.

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

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

Here COI denotes pupil position. This concept is illustrated in Figure 2. A simple four-point calibration mechanism is developed. It is used to estimate the region of eye being used to trace the 'Test Area'. It suggests the user to gaze at all corners of 'Test

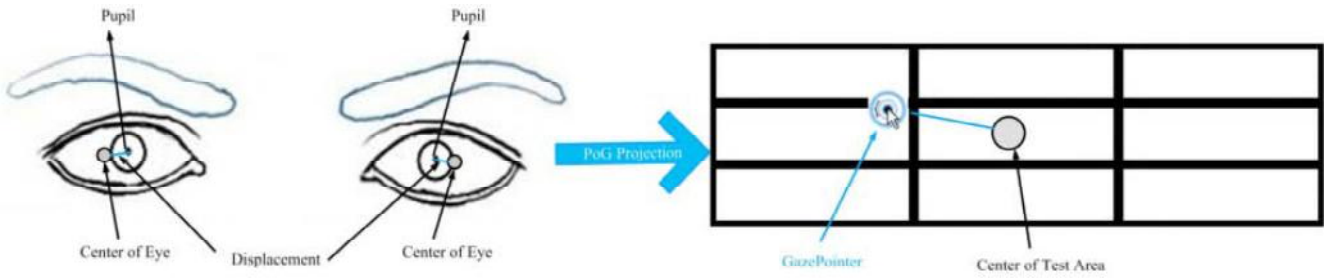

Figure 2: Point of Gaze Projection

Area' one by one, for an arbitrary amount of time. Achieved information is further used in PoG calculation algorithm, as described earlier.

Experiment Results

Two experiments have been designed to test the developed algorithms. The first test uses the algorithm in a live real-world environment and the second test uses a simulated environment. Experiments testing was performed on Intel® Core i3 HP Probook 4530s, it has a built-in web-cam with 720p resolution which was used as the image capturing device.

Experiment 1: GazePointer GUI

Experiment 1 is designed for real-world environment testing of the proposed algorithms. In order to compute accuracy, achieved results are recorded into a video file and later all recorded frames are manually analyzed. If face and eye patch are detected in a frame, rectangular boxes are drawn around detected face and eye patch by system. If the iris and pupil are found, circles are drawn around them by system. During analysis of recorded frames, the detection is considered correct it if the drawn rectangle or circle appears within a distance of 1 pixel from the exact location. Otherwise, it is considered incorrect.

The achieved result for a frame is presented in Figure 3. Detailed accuracy results of experiment 1 are given in Table 1. The algorithm resulted in 87% accuracy for experiment1. Experiment1 results are being reported on 228 frames. There are a few limitations for this experiment, it was only tested for frontal faces and with a single person facing the web-cam. The distance between a user's eyes and interface was not more than 75 cm. The system was tested in good lighting environment and did not allow head movements or use of glasses.

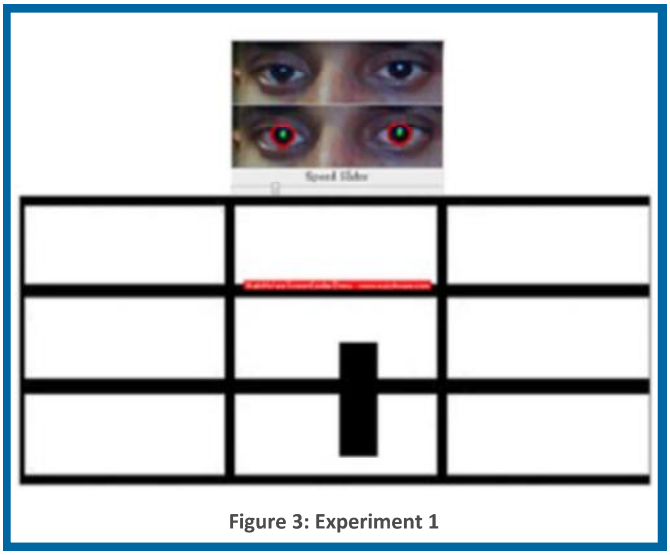

Figure 3: Experiment 1

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

Table 1: Experiment 1 Accuracy Results

Time efficiency results for experiment 1 are given in Table 2. Overall processing time for each frame is being reported to be 31 -43 ms, which provides interaction at rate of 23 fps.

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

Table 2: Experiment 1 Efficiency Results

Experiment 2: Test Bench

A test bench was designed to test developed algorithms in ideal scenarios. Artificial eyes were used in this experiment. These artificial eyes are computer drawings; one drawing contains an eye patch without an iris and other contains an iris. A computer program was setup to simulate eye movements based on eye movements of actual users . The achieved results for a frame are demonstrated in Figure 4. Experiment 2 results show that system accuracy remains 100% for simulated environment, results are given in Table 3.

Extracted Features	Accuracy
Pupil Detection	100%
PoG Computation	100%

Table 3: Experiment 2 Accuracy Results

Table 4 presents processing time for all modules in experiment 2. Achieved results show that system takes 8-13 ms per frames.

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

Table 4: Experiment 2 Efficiency Results

Conclusion and Future Research Directions

This paper presents a low-cost, real implementation of an eye gaze tracking algorithm for HCI. Eye gaze tracking has applications in several fields. The approach applies simple and classical computer vision algorithms and demonstrated the feasibility of the approach.

The system was limited by lighting, equipment, and user variation, e.g. posture and glasses. The performance deteriorated in poor lighting conditions. The GazePointer size was also large, since the web-cam was low resolution. This can be improved with enhancement in web-cam resolution and Test Area size. Image pre-processing procedures can also be employed to improve performance in poor lighting environments. Head-posture descriptor can also be included to allow head movements. Gaze estimation can also improve system performance and particle filers can be applied to achieve gaze estimation.

Acknowledgments

The authors would like to thank Mr. Muhammad Nafees Geelani, Ms. Sarah Chaudhry and Ms. Maryam Sohail for their help in conducting experiments.

How Work Was Done

This research work was carried out as Final Year Project during undergraduate degree. It was a team based project, which included student and a supervisor from University faculty (Mr. Muhammad Adnan Siddique).

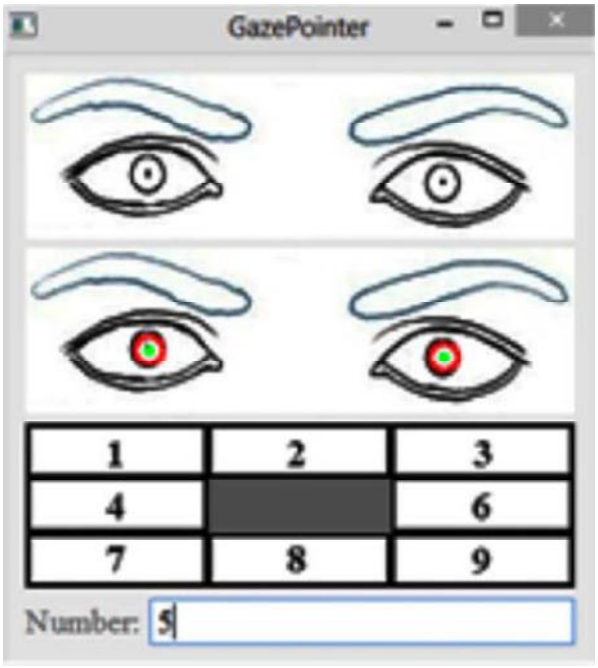

Figure 4: Experiment 2

References

[1] M. U. Ghani, S. Chaudhry, M. Sohail and M. N. Geelani, "GazePointer: A Real Time Mouse Pointer Control Based on Eye Gaze Tracking," 16th IEEE International Multi topic Conference (INMIC) 2013, Lahore, Pakistan, 2013.

[2] R. Barea, L. Boquete, M. Mazo and E. López, "System for assisted mobility using eye movements based on electrooculography," IEEE Transaction on Neural Systems and Rehabilitation Engineering, vol. 10, no. 4, pp. 209-217, DECEMBER 2002.

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

[4] 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.

[5] P. Ballard and G. C. Stockman, "Computer operation via face orientation," Pattern Recognition, 1992. Vol.I. Conference A: Computer Vision and Applications, Proceedings., 11th IAPR International Conference on, 1992.

[6] T. Horprasert, Y. Yacoob and L. S. Davis, "Computing 3D head orientation from a monocular image sequence," 25th Annual AIPR Workshop on Emerging Applications of Computer Vision, 1997.

[7] 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.

[8] D. H. Yoo, J. H. Kim, B. R. Lee and M. J. Chung, "Non-contact Eye Gaze Tracking System by Mapping of Corneal Reflections," Fifth IEEE International Conference on Automatic Face and Gesture Recognition (FGRí02), 2002.

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

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

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

[12] J. F. Canny, "Finding edges and lines in images," M.S. thesis, Dept. Elect. Eng. Comput. Sci. , Massachusetts Institute of Technology, Massachusetts, CA, 1983.

About the Authors:

Muhammad Usman Ghani holds a Bachelor of Science degree in Electrical Engineering from COMSATS Institute of Information Technology (Lahore, Pakistan). His research interests are Computer Vision, Affective Computing and Human-Computer Interaction.

Muhammad Adnan Siddique has been a Lecturer at Department of Electrical Engineering, COMSATS Institute of Information Technology (Lahore, Pakistan). His major research interests include remote sensing, SAR Tomography and SAR Signal Processing.