International Journal of Scientific Research in Information Systems and Engineering Volume 1, Issue 2, December - 2015
ISSN 2380-8128

Computer Game Controlled by Eye Movements

Yusuf Sait Erdem[1], Ibrahim Furkan Ince[2*], Huseyin Kusetogullari[3], Md. Haidar Sharif[4]

Department of Computer Engineering
Gediz University
Seyrek, Izmir 35665 – TURKEY

E-mail[1]: yusuf.erdem2011@ogr.gediz.edu.tr
E-mail[2]: furkan.ince@gediz.edu.tr
E-mail[3]: huseyin.kusetogullari@gediz.edu.tr
E-mail[4]: md-haidar.sharif@gediz.edu.tr

*Corresponding author: Ibrahim Furkan Ince

Abstract— sundry years ago people played video games for fun merely. Nowadays, video games are correlated to education, medicine, and researches. In this paper, we have addressed a computer game which takes input from a video camera by detecting user looking direction as well as eye gestures. Since webcam is easily accessible, we have carefully weighed it as video input device. We have tried to use the eye movements as the human computer interaction (HCI) tool, which would be used instead of a mouse. In general, this furnishes much easier and faster interaction with computer for everyone especially elders, children, and disabled people who cannot use mice. We have also evinced that eye tracking and eye gaze estimation can be very easy and fast way to obtain HCI. Experimental results show the potential usages of our proposed game.

Keywords— Eye Tracking, 2D Eye Gaze Estimation, Computer Games Controlled by Eye Movements, Human-Computer Interaction.

1 Introduction

BESides pleasures, at present, video games are playing a vital role in education, medicine, and researches. For example, video games are taking important role in the development of children. Some 15 years ago every parent was trying to educate their children apart from the school by teaching themselves or buying books for them, but now technology and video games makes it much easier. By playing games children can learn languages, differentiate shapes and colors from the childhood without forcing and having a fun [1]. There is fast development on HCI technologies for controlling video games. Big companies find very different ways to get user input for better game interactions in earlier days. There are three popular different approaches for this purpose: (i) Analyzing data from accelerometer sensing of hardware; (ii) Speech recognition; and (iii) Analyzing optical input. Nevertheless, most of these approaches require specific hardware. Some of them also requires extra physical effort that users are not used to. Gamer would feel more comfortable if there would exist better tools for gaming.

Eye gaze estimation is the one of the most important HCI tool that is getting developed, instead of just getting user's input by keyboard and mouse [2, 3]. We have aimed to show that the eye movements may be used as a HCI tool, which may be used instead of mouse even in our daily lives without extensive hardware. This provides much easier and fast interaction with computer for everyone including old people, children and disabled people who cannot use mouse. There are many algorithms about eye gaze tracking system and eye gesture detection, but many of them are not effectively used in our daily lives. Our aim is to design the software that optimizes computer vision techniques on gaze tracking and eye gestures detection for a computer game. Consequently, these techniques would be efficiently used in our daily lives more often by providing much easier and faster interaction with computer for everyone including old people, children, and disabled people who are not able to use a mouse easily.

In this paper, we have addressed a computer game which can be controlled by eye gaze estimation and gestures. Our proposed system is fast enough to run on an average PC and applicable to work with low-resolution webcam video stream. Eye gaze and gestures are detected on the visual input of the user. Detection of the head and pupil positions of a user is important to estimate user's eye gaze. For this anticipation, area of eyes and pupil position of the user are detected. Area of eyes provides opportunity for the rough position of user's head. The cursor on the screen is moved according to the relative position of the pupil on the area of eyes of the user. For detecting the pupil position a low cost pupil detection algorithm [4] has been taken into account.

The remaining part of this paper has been organized as follows: Section 2 accords with the most fashionable techniques. Section 3 describes in vivid detail of our proposed framework; Section 4 reports some experimental results; finally, Section 5 presents the conclusion of the work with few glimpses for further study.

2 State-of-the-art Methods

Accuracy, tracking frequency, and hardware dependencies are important keys for eye gaze tracking systems. Since accuracy determines the feasibility of selection of targets e.g., images and buttons, this is used for benchmarking of the eye gaze detection systems. In addition to, speed of the systems is another important reference of such systems [5]. There are miscellaneous techniques described in literature for eye gaze estimation. Nevertheless, a few of them do not require additional special hardware and require manual initialization of pupils. For instances, the algorithm of Hansen et al. [6] the iris is modeled as an ellipse, but that technique requires high quality image taken from very close of user's eyes. For instances, Hansen et al. [6] modeled the iris as an ellipse, but that technique requires high quality image taken from very close of user's eyes. The systems described by Noureddin [7] and Park [8] required visuals from two cameras with different angles. As we are interested in developing a game that is able to run with easily accessible hardware, we have researched more about algorithms which only need video input without special lighting tools.

Snake algorithm proposed by Kass et al. [9] is also used for pupil detection. But it requires improvement on its very low accuracy. Williams [10] proposed an improvement to minimize energy functional of algorithm by using greedy algorithm technique. Additional to this method, Choi [11] hinted an improved method for segmentation. Abe et al. [12] used multiple snakes to increase completeness of that method but since none of versions of snakes are not efficient in case to detect only pupils as those method are not specifically aims to find circles. Ince et al. [13] introduced an algorithm for fast and sub-pixel precise detection of eye blobs for extracting eye features. A low-cost system for two-dimensional eye gaze estimation with low-resolution webcam images was presented by Ince et al. [4].

A light-reflection-based method proposed by Yoo et al. [14]. It requires 4-light-emitting-diode sources placed around the screen, 1-light-emitting-diode source on the camera and 2 cameras; one for wide vision of user to detect user's face and one for zooming to one pupil of user. Corneal reflection of 5-light-emitting-diode sources on users one of the pupils detected on camera input. Eye gaze estimated by projection calculation of 5 reflection points of light-emitting-diode sources. This technique has good accuracy, lets large heads movements of the user and it does not require to detect head pose additionally. In spite of this, it needs additional excessive hardware such as 5-light-emitting-diode sources and higher quality cameras than average webcams.

Sewell et al. [15] proposed a method which does not require any additional hardware except an average webcam for eye gaze estimation on an average PC. Haar-like object detection is applied to video input. User's face and eye areas are detected. Artificial neural network techniques applied on one detected eye area to detect iris. Iris detection is only applied to one eye area rather than both eye areas for simplicity. By projection on iris location and head location that is detected by Haar-like object detection, eye gaze is estimated as x and y coordinates on the screed and cursor is placed accordingly. Nevertheless, this method troubled with jumped movements on cursor because of noisy input from average webcam.

3 Implementation Steps

3.1 Frontal Eye Areas Detection

Visual of user whose entire face can be seen clearly by the camera is streamed from the camera. Tolerably plain background is required for robust frontal area of eyes detection. Frontal area of eyes is detected by using Haar-like object detectors which was proposed by Viola et al. [16] and extended by Lienhart et al. [17] and is available in the OpenCV library [18]. By applying this method to the input frame, for example Figure 1, search area is reduced for the pupils and user's head position calculated in purpose of calculating pupil position relatively to user's head position.

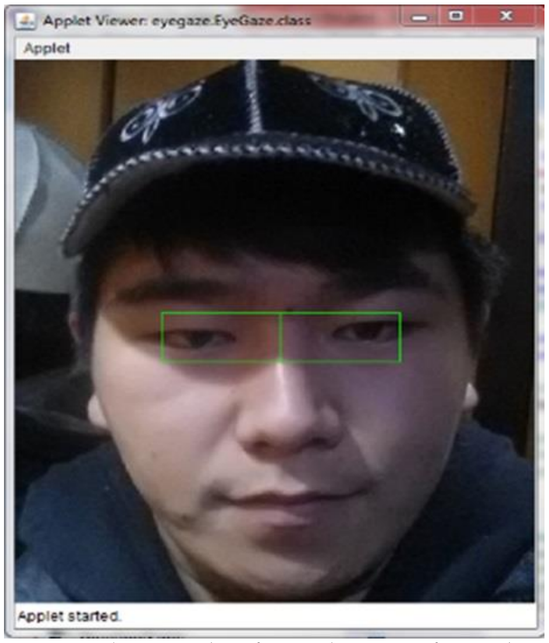

Fig. 1. Sample Snapshot for Reduction of Search Area

This information used on estimation of eye gaze. A frontal area of eyes cascade is used by Haar-like object detector to reduce false pupil detections and increase speed, instead of detecting eyes one-by-one. The found area is divided to half in length to apply pupil detection for left and right eye separately. Since frontal area of eyes detection process only returns horizontal result, head rotations cannot be detected by this method as well as area of eyes cannot be detected at all on extensively rotated head visuals. Consequently, user is asked to hold his/her head fairly strait. As gamer usually plays computer games in such position, it will not cause an important issue.

3.2 Pupil Detection

One elliptic model with varying radius is traversed on two parts of search area (frontal area of eyes). Since this method requires radius of elliptic model as input, range of radius has to be determined for the search. This range is determined by the height of the search area. In purpose of traversing rectangular search area (e.g., Figure 2, a rectangular iteration method is purposed which starts traversing from given point and traversing continues by the closest neighbor according to Manhattan distance measure [19]. Manhattan distance measure is selected because of its simplicity. After detecting first pupil, starting point of iteration for search is set to center of detected pupil on next frames of video input for increasing speed and accuracy. Closer candidate pupils to the starting point have more change to be the correct pupil detection after first iteration.

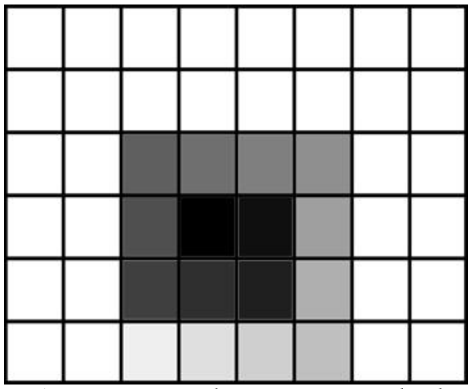
Fig. 2. Rectangular Iteration Method

On each step of traversing the search area, current pixel is counted as center of ellipse and coordinates of some number of points are calculated according to $x^2 + y^2 = r^2$ equation where x and y are the coordinates of current pixel and r is the current radius on iteration. Calculated coordinates of points on the edge of ellipse are checked by its outer neighbor and next neighbor on the ellipse. Distance from each point on the edge of ellipse to their outer and next neighbor pixels are calculated with the Euclidean distance as: $\sqrt{(r_1 - r_2)^2 + (g_1 - g_2)^2 + (b_1 - b_2)^2}$ where r_1, g_1, b_1 are red, green, and blue values of edge pixel, respectively as well as r_2, g_2, b_2 are red, green, and blue values of outer pixel, respectively. If the distance to outer pixel is higher than the edge threshold and distance to next neighbor is smaller than the line threshold, that pixel is considered on the arc. Figure 3 depicts the model of elliptic hollow kernel as follows:

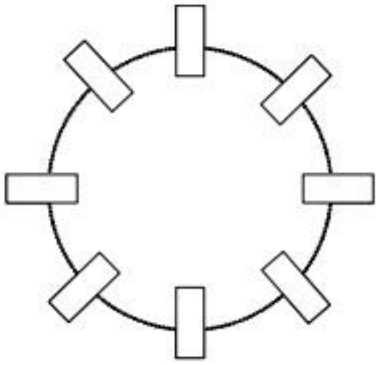
Fig. 3. Window-based Elliptic Hollow Kernel [4]

On getting results of each pixel on the edge of the ellipse, consequential arc points are clustered. Longest two clusters considered as true arcs of the ellipse, the rest is ignored. Sum of the number of edge points on arcs is divided to total number of points on the ellipse to find percentage of the completeness. Then average of Euclidean distance of all inner pixel's RGB values to RGB value of black color (Red: 0, Green: 0, Blue: 0) are calculated to find average intensity with $\frac{1}{N}\sum_{i=0}^{N-1} x_i$, where N be the number of pixels and x_i be the intensity of current pixel inside the elliptic hollow kernel. Figure 4 depicts a sample snapshot for the pixels fit to the elliptic hollow kernel as follows:

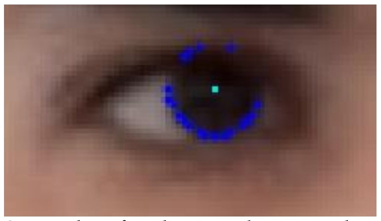
Fig. 4. Sample Snapshot for the Pixels Fit to the Elliptic Kernel

Second iteration through inner pixels is done to find covariance of intensity values of the pixels considering $\frac{1}{N}\sum_{i=0}^{N-1}(x_i - avg(X))$, where x_i be the intensity of current pixel, $avg(X)$ be the average intensity, and N be the total number of pixels of elliptic kernel area. This value divided to average intensity to calculate rate of variance. Fitness of the ellipse is determined by smallness of the rate of variance value. Thresholding is applied to candidate pupils. This thresholding reduces false pupil detections those are detected by elliptic edge detection. By this new feature of pupil, effect of corneal reflection is reduced since pupil is usually dark. Yet sometimes corneal reflection makes it brighter. If this rate of variance is 0, it means that it is a complete black filled ellipse. The ellipse with highest ellipse completeness of the remained pupil candidates, is resulted as the pupil.

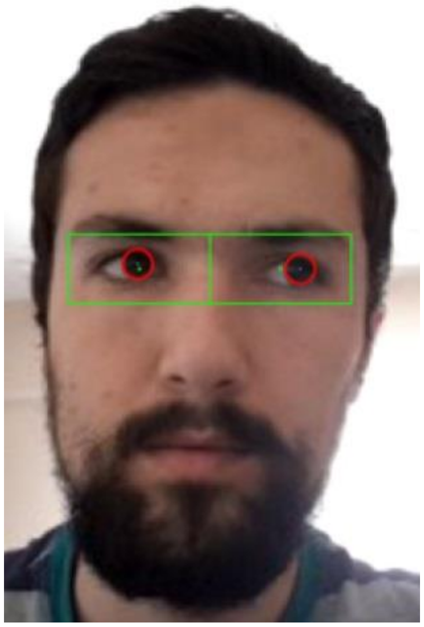
Fig. 5. Sample Snapshot for Pupils Detection

Figure 5 above demonstrates the pupils detected by low cost iterative pupil detection with elliptic hollow kernel algorithm [4].

3.3 Eye Gaze Estimation

For eye gaze estimation, head direction and pupil position are needed [4]. In this paper, user's head pose estimated as the face straightly towards to the video input device and the location of head determined by frontal both eye area detection. By reference of area of eyes, eye gaze estimated by the position of one pupil. Consequently, only one pupil of user is detected to increase system's performance. To calibrate eye gaze direction, user is asked to look middle of the screen for a while when system starts and user's current pupil position is saved as middle position. Pupil position is calculated as relatively to the frontal area of eyes, because by the small head movements or changing distances to the camera may change the position and size of frontal area of eyes thus by head movements pupil position is calculated correctly according to user's head position.

After center location is set game is started. On next frames pupil positions compared with center location of pupil and eye gaze direction is founded. To increase robustness of the system, directions are grouped to 8 as shown in Figure 6. Cursor on the screen is moved with constant speed in the specified direction or stops if pupil is close enough to center. Figure 7 demonstrates the model of shooting game controlled by eye gaze detection where cursor direction indicates movement direction of cursor as well as the model of *8* number of examples for pupil positions and corresponding cursor directions.

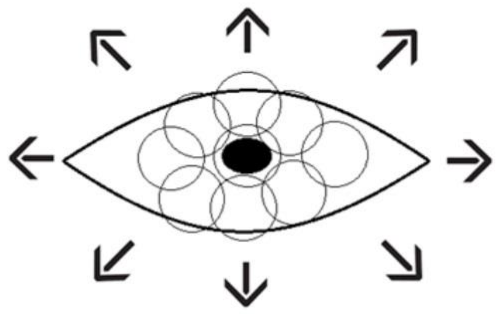
Fig. 6. The Model of 8 Number of Directions

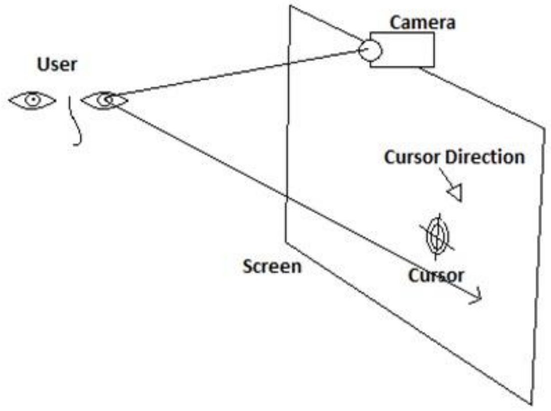

Fig. 7. Framework for Controlling the Cursor Movements

Normally, in the game, there is no traditional eye gaze estimation but eye gaze direction estimation by *8+1* directions. Shooting action is needed to get as input from the user additional to directions input. With this aim, user blinks to apply that action. A blink is detected if there is no pupil detection on frontal both eye area throughout the threshold time.

4 Experimental Results

4.1 Experimental Setup

We have tested our game using standard laptop with its built-in camera. Figure 8 depicts a screenshot from the shooting game. On the game, randomly placed circular target board images appear on the screen. Users moves the cursor image on the screen by his/her eye gaze direction and blinks strongly for applying shooting action (e.g., Figure 8). The target disappears under the cursor if there is any. Number of targets change according to the difficulty level. After given time is expired for the stage or every target get hit by player, new stage is loaded. After all stages completed final points are calculated and displayed on the screen.

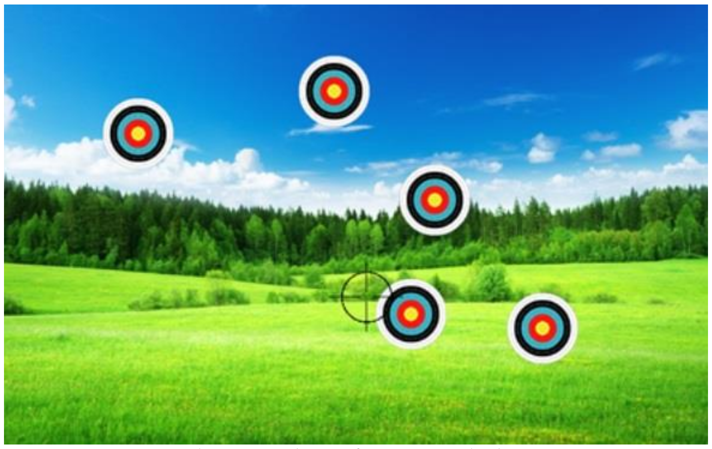
Fig. 8. Sample Snapshot of Designed Shooting Game

The designed shooting game framework is employed as the reference testing module for proposed 2D gaze estimation algorithm with no calibration. The size of the shooting targets determine the precision level. According to the users' eye movements, users are asked to shoot each selected target in a given duration. In our experimental setup, targets with diameter of 50 pixels are distributed to the screen in random positions. Duration of shooting for each target item is given as 5 seconds. The system was tested on 10 number of users whom were trained about the system usage before the actual experiment. The ratio of successful shootings to the overall shootings were observed as 90% as the measure of accuracy. As a future work, it is planned to add calibration module to the proposed system for performance enhancement.

4.2 Findings

We have implemented our proposed game algorithm using a small scale setup. Since eye movements are used as the human computer interaction tool, we do not need any kind of mouse. The proposed game is suitable for not only children but also elders and disabled people who cannot or reluctant to use mice. Outcome of our system shows the potential use of it in daily life applications. Our proposed system is fast enough to run on an average personal computer or laptop and applicable to work with low-resolution webcam video stream.

4.3 Shortcomings

The proposed system was tested with the detection of the head and pupil positions of a user. Area of eyes provided the rough position of user's head. There are still some errors on distinguishing adjacent directions of user's eye gaze. Accuracy should be increased to increase playability. But using only 8+1 eye gaze directions, false cursor movements are decreased largely. Nevertheless, the accurate head position was not estimated correctly and consequently shooting game would suffer from this shortcoming.

5 Conclusion

We suggested a low cost computer game using user's looking direction and eye gestures as well as easily accessible webcam. Low cost iterative pupil detection using elliptic hollow kernel algorithm as well as eye gaze direction estimation technique, a shooting game would be implemented to run on an average personal computer having an average webcam and the system may be robust to environmental changes such as tilt of camera and illumination changes. Experimental results demonstrated the potential usages of our proposed game. Yet the detection of accurate head position would be further studied.

Acknowledgment

This paper was extracted from an undergraduate course entitled: "COM497 - Senior Design Project" held in 2014-2015 academic period at Gediz University, Izmir, Turkey. Previous version of this study was presented in the "Third International Symposium on Engineering, Artificial Intelligence and Applications (ISEAIA 2015)" held in between 4 to 6 of November, 2015, in Girne American University, North Cyprus. This is the extended full version of the paper already published in the conference proceedings and selected to be published in this journal by the conference committee.

References

[1] Moursund, D., "Introduction to Using Games in Education: A Guide for Teachers and Parents", *Second Edition*, 2011.

[2] Zhai, S., "What's in the eyes for attentive input", *Communiations of the ACM*, vol.46, no.3, pp. 34-39, 2003.

[3] Kumar, M., Klingner, J., Puranik, R., Winograd, T., Paepcke, A., "Improving the accuracy of gaze input for interaction", *In Proceedings of the 2008 symposium on Eye tracking research and appliations (ETRA'08)*, pp. 65-68, 2008.

[4] Ince, I., Kim, J., "A 2d eye gaze estimation system with low-resolution webcam images", *EURASIP Journal on Advances in Signal Processing*, 2011.

[5] Bulling, A., Gellersen, H., "Toward mobile eye-based human-computer interaction", *IEEE Pervasive Computing*, vol.9, no.4, pp. 8-12, 2010.

[6] Hansen, D.W., Pece, A.E.C., "Eye tracking in the wild", *Computer Vision and Image Understanding*, vol.98, no.1, pp. 155-181, 2005.

[7] Noureddin, B., Lawrence, P.D., Man, C.F., "A non-contact device for tracking gaze in a human computer interface", *Computer Vision and Image Understanding*, vol.98, no.1, pp. 52-82, 2005.

[8] Park, K., Chang, J., Whang, M., Lim, J., Rhee, D.W., Park, H., Cho, Y., "Practical gaze point detecting system", *Lecture Notes in Computer Science (LNCS)*, vol.3175, pp. 512-519, 2004.

[9] Kass, M., Witkin, A., Terzopoulos, D., Snakes, "Active contour models", *International Journal of Computer Vision*, vol.1, no.4, pp. 321-331, 1988.

[10] Williams, D.J., Shah, M., "A fast algorithm for active contours and curvature estimation", *CVGIP: Image Understanding*, vol.55, no.1, pp. 14-26, 1992.

[11] Choi, W.P., Lam, K.M., Siu, W.C., "An adaptive active contour model for highly irregular boundaries", *Pattern Recognition*, vol.34, no.2, pp. 323-331, 2001.

[12] Abe, T., Matsuzawa, Y., "A region extraction method using multiple active contour models", *In Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, vol.1, pp. 64-69, 2000.

[13] Ince, I., Yang, T.C., "A new low-cost eye tracking and blink detection approach: Extracting eye features with blob extraction", *Lecture Notes in Computer Science (LNCS)*, vol.5754, pp. 526-533, 2009.

[14] Yoo, D.H., Chung, M.J., Ju, D.B., Choi, I.H., "Non-intrusive eye gaze estimation using a projective invariant under head movement", *In Proceedings of IEEE International Conference on Robotics and Automation (ICRA'06)*, pp. 3443-3448, 2006.

[15] Sewell, W., Komogortsev, O., "Real-time eye gaze tracking with an unmodified commodity webcam employing a neural network", *In Extended Abstracts on Human Factors in Computing Systems (CHI'10)*, pp. 3739-3744, 2010.

[16] Viola, P., Jones, M.J., "Robust real-time face detection", *International Journal of Computer Vision*, vol.57, no.2, pp. 137-154, 2004.

[17] Lienhart, R., Maydt, J., "An extended set of haar-like features for rapid object detection", *In Proceedings of the International Conference on Image Processing*, vol.1, pp. I-900-I-903, 2002.

[18] "OpenCV: Open Source Computer Vision", *Opencv.org*, 2015.

[19] Black, P.E., "Manhattan distance in Dictionary of Algorithms and Data Structures", *http://www.nist.gov/dads/HTML/manhattanDistance.html*, 2015.