UACEE International Journal of Advances in Computer Science and its Applications – IJCSIA
Volume 3 : Issue 2 [ISSN 2250 – 3765]
Publication Date : 05 June 2013

Eye Position Estimation using Distance between Eyes Calibration base on Iris Tracking

Naphat Rattanathawornkiti
Department of Electrical Engineering, Kasetsart University, Bangkok, Thailand, 10900
naphat12@hotmail.com

TeerasitKasetkasem
Department of Electrical Engineering, Kasetsart University, Bangkok, Thailand, 10900
fengtsk@ku.ac.th

Thitiporn Chanwimaluang
Knowledge Elicitation and Archiving Laboratory, National Electronics and Computer Technology Center, Pathumthani, Thailand, 12120
thitiporn.chanwimaluang@nectec.or.th

Makoto Sato
Precision and Intelligence Laboratory, Tokyo Institute of Technology, Nagatsuta-cho, Midori-ku, Yokohama, 226-8503, Japan
msato@pi.titech.ac.jp

Abstract—**This paper proposes a new desktop virtual reality system that uses only a single camera to track and calculate the distance between a viewer's eyes and the camera in order to produce proper viewing images. The system estimates a distance between the eyes and the camera by measuring the distance between irises of left and right eyes. The system has the initial step to calibrate distance between left and right irises of the viewer and use these irises as the reference points to find the distance between the eyes and the screen. However, since this system uses only two eyes, it cannot cope with the rotation on the axes parallel to the camera. The FaceAPIlibrary [1] is employed as the orientation estimator since the FaceAPI uses the sophisticating image processing algorithm to detect and measure human face orientation and distance with respect to the camera. However, since not all human faces have equal size, the estimated distance between a human face and camera from the FaceAPI library is usually inaccurate. Nevertheless, we can use the orientation information derived from the FaceAPI to adjust distances between left and right irises of a viewer. Our experiment shows that the distance between two irises from the system is more accurate the system using only the FaceAPI.**

Keywords—Keywords: Desktop Virtual Reality, Iris Tracking, FaceAPI

I. Introduction

The Desktop virtual reality (DVR) is a virtual reality module that involves displaying a 3-dimensional virtual world on a regular desktop display. The DVR can be applied for many applications such as a flight simulator, entertainment, advertisements and medication applications. To create DVR, it is important to find the position of a viewer to create the proper size of an image to interact with the viewer correctly, and appear to be realistic. Therefore, many researchers employed many approaches and used different technologies for finding the position of a viewer's head such as the Kinect [2], 6DOF Logitech head tracker [3], Wii Remote[4][5]. However, these technologies require additional hardware which sometimes is expensive and cumbersome to use. Therefore, it is inconvenient for a viewer to use and install.

The use of a single camera for finding the eye position of the viewer is an interesting approach. This approachallowsthe DVR to be implemented without the need to purchase any additional equipment since the modern computers usually equip with a camera, and hence, a viewer can employ the DVR this with ease.

There are many researchers such as [6]that use only a single camera to track eyes of a subject or estimate face pose of a person of interest, but there is no research work that can show the position of the eyes in a 3D axis. However, the "Seeing Machine" has implemented the FaceAPI [1], a C++ library, that is able to detect and give the estimated rotation angle values of the face in each axis, but the FaceAPI has significant errors when finding the distance between a viewer's face and the camera because the FaceAPI relies on the size of the human head as a distance in distance estimation. There is one researcher that attempted to improve the distance error of the FaceAPI system by integrating information from the Wii Remote to correct [7]. Their results show significant improvement in the term of accuracy. However, a viewer must wear glasses all the time.

This research attempts to find the distance between viewer's eyes and a camera by using a the estimated distance between two irises of a viewer. This approach is similar to thesystem equipped with the Wii Remote and the sensor bar. However, our system does not require any additional hardware and, hence, it is easier to be deployed. Like the Wii remote system, our approach can only correctly determine the position of viewer if the face rotation is known. As a result, the FaceAPI is used to estimate the face rotation.

This paper is organized as follows. Section II presents a new system for finding eye position. In Section III is explained the use of the FaceAPI in calculating eyes position when face rotation occur, In Section IV show the experiment result and in section V is conclusion.

II. New system for finding eye position

Our system is divided into three parts. First, we present an algorithm to track irises of viewer's eyes. Then, we introduce the process in initial step to estimate a real distance between two irises of a viewer and use them as reference points. After that we explain how to use the distance between two irises of a viewer to compute distance between the viewer and camera.

A. *Iris Tracking*

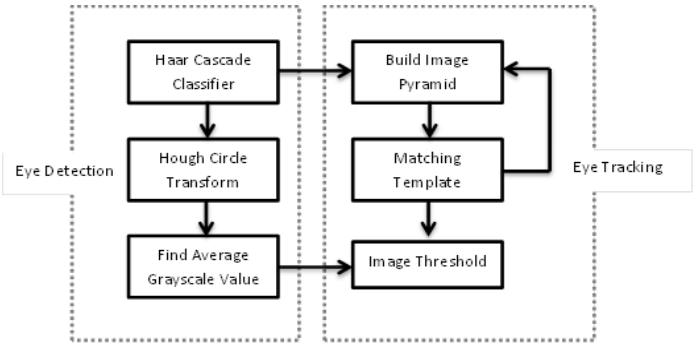


Figure 1. The block diagram of the iris tracking system.

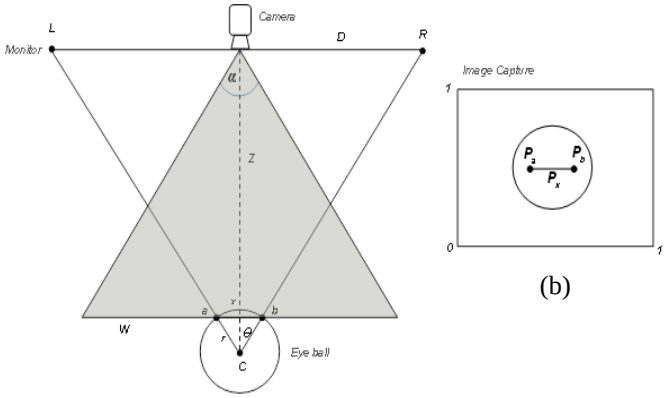
Since the iris of a human eye has a low intensity value (dark area) than surrounding area. Thus, we use the value of intensity to track the iris. The iris tracking starts with identifying the location of an eye by using the Haar cascade classifier [8]. Next, the Hough circle transform [9] is used to find circle in the eyes area and we identify this area as iris. When the system successfully detects eye and locate the iris, the image pyramid technique [10] is employed to track eyes area by using the template matching technique [11], and the iris location is determined by identifying thearea with same intensity value from pervious frame.

B. *Calibration distance between two irises of viewer*

In determining the distance between two irises of viewer, it is necessary to find the distance between viewer's eye and camera first. For example, when viewers move their face closer to the camera, the distance between two irises of viewer's eyes that appear on captured image by camera will be further, vice versa. Therefore, we divided the calibration process into two phases. First, we find distance between viewer's eye and camera. Next, we find distance between two irises of viewer's eyes calculated by distance between viewer's eye and camera.

1) Find distance between viewer's eye and camera

Firstly, we need to calibrate the distance between the viewer's eye and camera by assigning a viewer to look at two points on the screen without the face movement. Here, only an eye closest to center of the camera is used since when an eye placed in the center of the camera, the distance from iris to screen when looking at two points in the opposite end of the screen are the same.Figure 2(a) shows eye movement in real world when looking at the *L* and *R* points in the screen. Here, *a* and *b* are the iris centers when looking at the points *L* and *R*, respectively where *W* is a maximum capture range at eyes position and α is a maximum view angle of the camera.



(a) (b)

Figure 2. The variables in real world that use to calculate position of eyes (a), the image was obtained by camera which has the points of iris center (b)

The positions of iris center that looks at points *a* and*b* is appeared on the captured images as an iris tracking as points P_a and P_b (Figure 2). The ratio between the actual iris movements, *x*, with respect to maximum capture range, *W*, of the camera is the same as the relative distance, P_x(Figure 2(b)) of the iris movement in the captured images with respect to the image width

$$\frac{W}{x} = \frac{1}{P_x} \quad (1)$$

In this system, the camera have the maximum view angle (α) of 75°. Hence, the distance between the eyes and the camera can be written as

$$Z = \frac{\frac{W}{2}}{\tan\left(\frac{\alpha}{2}\right)} \quad (2)$$

Furthermore, for a given, the distance between two points on the screen, *D* and by using the similar triangle principle, we have

$$x = \frac{Dr\cos\theta}{Z} \quad (3)$$

Where*r* is approximately the radius of eyeball of 1.25 cm [12]

From Eq.(1),(2) and (3), we can calculate the distance froman eye to the screen as

$$Z = \sqrt{\frac{\frac{Dr\cos\theta}{P_x}}{\tan\left(\frac{\alpha}{2}\right)}} \quad (4)$$

For our experiment, we found that the maximum movement of human iris inside an eye is less than 1.15 cm (Fig. 3.) Therefore, the maximum angle that an iris moves is less than 24.7024 degree from center which corresponds to $\cos\theta$ of greater than 0.9088. Hence, Eq. (4) can be approximated as

$$Z \approx \sqrt{\frac{\frac{Dr}{P_x}}{tan\left(\frac{\alpha}{2}\right)}} \quad (5)$$

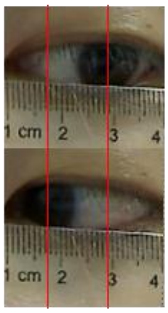


Figure 3. The change of iris when look at defference view points.

2) Find distance between two irises of viewer's eyes

The distance between the eye and camera plays an important role in finding the distance between two irises of a viewer's eyes. Figure 4 displays the camera and a viewer's head. E_L and E_R are the positions of a viewer's eyes (left eye and right eye). P_L and P_R are the corresponding points of the eyes appeared on the captured images as an iris tracking.

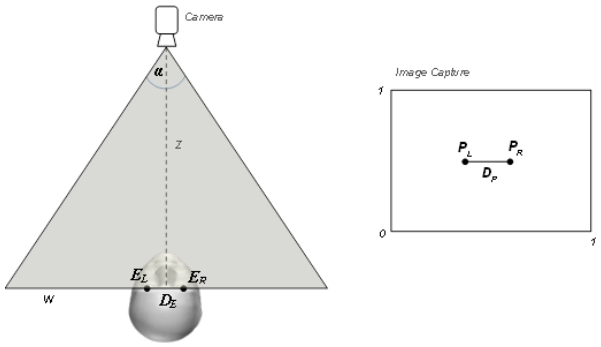


Figure 4. The variables that use to calculate distance between two irises in real world and the image was obtained by camera (b)

From Eq. (2), we can find the view range of camera (W) by using distance between the eye and camera as

$$W = 2Ztan(\frac{\alpha}{2}) \quad (6)$$

Since the ratio between distance from E_L to E_R in Fig.4 (D_E) and view range of camera (W) must be equal to the relative distance in pixels of left and right eye irises with respect to the screen width. Therefore, we can estimate the real distance between two irises as

$$D_E = WP \quad (7)$$

C. *Eyes position estimation*

We can now computethe distances from a viewer at different positions to the camera. In Figure 5, we illustrate the effect of viewer distance to the camera to the captured images. Here, we assume Z_2 is smaller than Z_1. Hence, the relative distance between irises appeared on the screen when a viewer is located at the distance Z_2 is larger than those at the distance Z_1. The relationship between the distance between a viewer to the camera and the relative distance between irises P is given by

$$\boldsymbol{Z} = \frac{\frac{\boldsymbol{D}}{\boldsymbol{P}}}{\boldsymbol{2}} \quad (8)$$

Where α in the maximum view angle of the camera

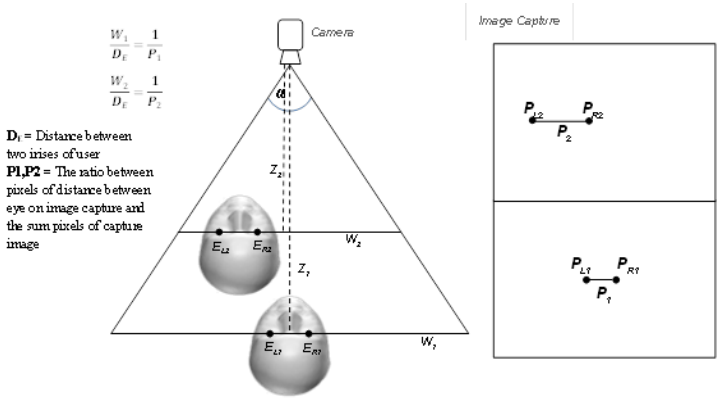


Figure 5. The view range of camera in different distance between eyes and the camera.

Although, we can use the distance between eyes in pixels to find distance between eyes and camera in the real world coordinate, this approach only work under the conditions that human face is parallel to the image plan.When there is a rotation around the *y*-axis (the axes parallel to the camera) the estimated distance computed by Eq. (8) is incorrect. In fact, when the viewer face is rotated, the estimated distance is always further from the actual one. To cope with this problem, we employ the FaceAPI library [1] to estimate orientation of face and use this information to correct the calculated distance computed by the distance between eyes.

The orientation information provided by the FaceAPI library can be incorporated with the relative irises position provided by the iris tracking. Figure 6 displays the effect of human face rotation in *y*-axis to the relative positions of two irises resulting from iris tracking. If the actual distancebetween irises is P and rotation in y-axis inθ_y, the two irises will appears in the camera to have distance of P' where P' is given as

$$P' = P\cos\theta_Y \quad (9)$$

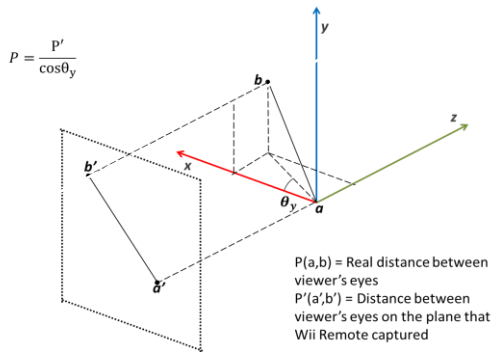


Figure 6. The points of eyes that the camera receive

However, the distance from our system, the distance from FaceAPI and the orientation angle provided by the FaceAPI still have errors. To improve the accuracy of distance between the viewer's eyes and camera, and rotational angle of the viewer's head in y-axis, we propose the energy function given by

$$E = k_1(Z - Z_f)^2 + k_2\left(\frac{Z}{cos\theta} - Z_e\right)^2 + k_3(\theta - \theta_f)^2 \quad (10)$$

whereZ_f is the distance from FaceAPI , $Z_e$is the distance from our system and θ_f is the orientation degree in y-axis from FaceAPI. k_1,k_2 and k_3 are constants that give priority to each variables. In other words, k_1, k_2 and k_3 are the amount of influence of theviewer distance derived from FaceAPI, the distance divided by distance between irises, andthe face orientation derived from FaceAPI, respectively. Because the distance viewer and camera derived from the FaceAPI has low accuracy, the value of k_1 is set to be low. For experiment of 602 samples, we found that the distance from our system has high accuracy and $k_2$places importance on both distance and orientation degree. Hence, we use $k_1 = 1$, $k_2 = 4$ and $k_3 = 1$

From the energy function (10), we can find the answer of distance(Z) and answer of orientation(θ) that give the lowest energy by finding distance(Z) whenthe derivative of the energy with respect to the distance Z and orientation angle are equal to zero, i.e.,

$$k_1Z - k_1Z_f + k_2Zsec^2\theta - k_2Z_esec\theta = 0, \quad (11)$$

and

$$k_2Z^2tan\theta - k_2Z_eZsin\theta + k_3(\theta - \theta_f)\cos^2\theta = 0, \quad (12)$$

respectively.

Here, the Newton method [13] is employed to solve Eq. (11) and (12).

III. Experimental result

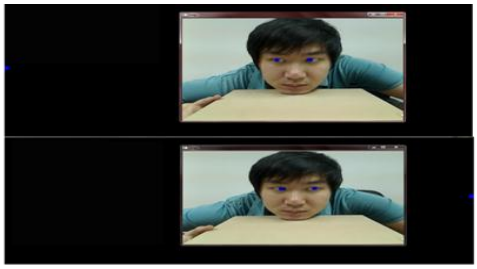


Figure 7. Calibration distance between two irises by look at two points

We divide the experiment into two parts. In this first part, we investigate our proposed distance estimator between the left and right irises. In experiment, the five viewers(3 males and 2 females) will have to look at two points separated by the distance of 30.5 cm on the screen at the distance comfortable to the each individual vieweras shown in Fig.7. The experiment is repeated five times and the resultsare summarized on Table I. Here, we found that the distance computed by our proposed algorithm is closed to the actual distances measured by a ruler with the errors less than 5%. These results demonstrate the effectiveness of our proposed algorithm.

TABLE I. EXPERIMENT RESULT WITH USING DISTANCE BY RULER IN CALCULATION

Distance by ruler(cm)	Average distance from new system(cm)	RMSError(cm)	Error(%)
5.7	5.90085	0.28452	4.99152
6.2	6.14537	0.09998	1.61256
5.8	5.86772	0.25842	4.45556
6.1	6.15878	0.11779	1.93096
6.4	6.51040	0.27389	4.27949

In the second part, we examine the performance of our eyes-to-camera distance estimation algorithm by comparing the results from our proposed algorithm with the FaceAPI library and the system when actual distances between left and right irises are measured by a ruler. The experiments are conducted with difference face orientation around the y-axis. Here, we use 0, 15 and 30 degrees and at distance of 30 and 50 cm between eyes and the camera Table II, Table III and Table IV summarize the results of our experiment. In measuring by ruler, the accuracy is lower than those of our proposed algorithm. We believe that our measurement of the distance between irises by a ruler may not be accurate since the surface between the irises is not flat. However, in some experiments such as the experiment at 15 degree of face rotation at the distance of 50 cm, our proposed algorithm may have some error form little movement of the viewer's head while initial step of iris-to-iris distance estimation. From the errors of both of them, both methods also are more accurate than using FaceAPI only.

TABLE II. EXPERIMENT RESULT WITH USING DISTANCE BETWEEN IRISES MEASURED BY A RULER

Distance (cm)	Rotation in Yaw (degree)	RMSError (cm)	Error(%)	Samples
30	0	1.31549	4.38496	236
	15	1.48102	4.93673	218
	30	0.45335	1.51116	124
50	0	1.28322	2.56644	98
	15	2.53459	5.06918	115
	30	1.54608	3.09216	148

TABLE III. EXPERIMENT RESULT FROM OUR PROPOSED SYSTEM

Distance (cm)	Rotation in Yaw (degree)	RMSError (cm)	Error(%)	Samples
30	0	0.86035	2.86783	236
	15	1.08891	3.62970	218
	30	0.29479	0.98263	124
50	0	0.58855	1.17710	98
	15	1.38974	2.77948	115
	30	1.64682	3.29364	148

TABLE IV. EXPERIMENT RESULT WITH USING ONLY FACEAPI

Distance (cm)	Rotation in Yaw (degree)	RMSError (cm)	Error(%)	Samples
30	0	1.82753	6.09176	236
	15	2.17216	7.24053	218
	30	1.03100	3.43666	124
50	0	2.60357	5.20714	98
	15	2.90862	5.81724	115
	30	2.68201	5.36402	148

For the experiment result of all samples shown in Table V, the eyes-to-camera distance from our proposed has the most accuracywhereas the FaceAPI has the lowest.

TABLE V. EXPERIMENT RESULT OF ALL SAMPLES

Type	RMSError(cm)	Samples
A system with irises distance measured by a ruler	1.42235	939
Our proposed system	0.99915	
The FaceAPI System	2.15043	

IV. Conclusion

This research proposes a new system to track the eyes position by use only a single camera. Here, the irises are used as the reference points and the FaceAPI is used to estimate a viewer head orientation. Our results have shown that our system can greatly improve the accuracy in the estimation of aviewer position to the camera from the system that equipped with the FaceAPI alone.

Acknowledgment

This research is financially supported by Thailand Advanced Institute of Science and Technology (TAIST), National Science and Technology Development Agency (NSTDA) Tokyo Institute of Technology and Kasetsart University (KU).

References

[1] FaceAPI : http://www.seeingmachines.com/product/faceapi

[2] F.A.Kondori, S.Yousefi, H.Li, S.Sonning, S.Sonning, “3D Head Pose Estimation Using the Kinect”, Wireless Communications and Signal Processing (WCSP), pp. 1-4, Nov 2011.

[3] L.Liu, R.Liere, “Modeling Object Pursuit for DesktopVirtual Reality,” IEEE Transactions on Visualization and Computer Graphics, vol. 18, no. 7, pp. 1017-1026, July 2012.

[4] J.C.Lee, “Hacking the Nintendo Wii Remote”, Pervasive Computing, IEEE, vol. 7, pp. 39-45, July 2008.

[5] Bharath L, Shashank S, Nageli V S, SangeetaShrivastava, S Rakshit, “Tracking method for Human Computer Interaction using Wii Remote”, Emerging Trends in Robotics and Communication Technologies (INTERACT), pp. 133-137, Dec 2010.

[6] W.W.Kim, S.Park, J.Hwang, S.Lee, “Automatic Head Pose Estimation from a Single Camera Using Projective Geometry”, IEEE 2011.

[7] N.Rattanathawornkiti, T.Kasetkasem, T.Chanwimaluang, M.Sato, “Fusion between Wii Remote and FaceAPI for Desktop Virtual Reality”, 4th International conference, 2012.

[8] Paul Viola and Michael J. Jones. “Rapid Object Detection using a Boosted Cascade of Simple Features”, Computer vision And Pattern recognition, 2001.

[9] Yuen, H. K., Proncen, Illingworth, J., Kittler, J., “Comparative Study of Hough Transform Methods for Circle Finding”, Image and Vision Computing, Vol.8, No.1, 71-77, 1990.

[10] Ogden, J. M., Adelson, E. H., Bergen, J. R., Burt, P. J., “Pyramid-Based Computer Graphics”, RCA Engineer 30, 4–15, 1985.

[11] Opencv API Reference, “Image Processing: matchTemplate”, http://docs.opencv.org/modules/imgproc/doc/object_detection.html?highlight=matchtemplate#matchtemplate

[12] Encyclopedia Britannica Macropedia: Sensory Reception. USA: Britannica Inc., 1997.

[13] Abramowitz, M. and Stegun, I. A. (Eds.). “Handbook of Mathematical Functions with Formulas, Graphs, and Mathematical Tables”, 9th printing. New York: Dover, p. 18, 1972.