International Journal of Engineering Applied Sciences and Technology, 2019
Vol. 4, Issue 6, ISSN No. 2455-2143, Pages 144-149
Published Online October 2019 in IJEAST (http://www.ijeast.com)

WEBCAM BASED EYE-GAZE ESTIMATION

Surya Govind,
Dept. of Computer Science and Engineering,
Vel Tech Rangarajan Dr. Sagunthala R & D Institute, of Science and Technology.
Chennai, Tamil Nadu, INDIA

***Abstract*— This research addresses to eye gaze tracking problem using low cost and more convenient web camera in a desktop environment, as opposed to gaze tracking techniques requiring specific hardware, e.g. infrared high resolution camera and infrared light sources, as well as a cumbersome calibration process. In the proposed method, we first track the human face in a real-time video sequence to extract the eye region. Then virtual reference point is done with key points to provide the real-time reference for gaze estimation. The eye-gaze tracking is accomplished by integration of the eye vector and the head movement information. Experiments are performed to estimate the eye movement and head pose. Usually, eye gaze systems are really systematic and high in cost, so a question arises that is it possible to get this system at low cost. I have tried here to provide a scriptural based algorithm that will fulfil the same purpose for tracking eyes.**

Keywords- **Gaze estimation, Reference points, RANSAC, Pupil centre, Pygame, numpy**

I. INTRODUCTION

Eye gaze tracking technology over the past few decades have led to the development of promising gaze estimation techniques and applications for human-computer interaction. Historically, research on gaze tracking dates back to the early 1900s, starting with invasive eye-tracking techniques. These included electro-occulography using pairs of electrodes placed around the eyes or the scleral search methods that include coils embedded into a contact lens adhering to the eyes. The application focus towards general-purpose human-computer interaction was sparse. This changed in the 1990s as eye gaze found applications in computer input and control. Post-2000, rapid advancements in computing speed, digital video processing and low-cost hardware brought gaze tracking equipment closer to users, with applications in gaming, virtual reality and web-advertisements. In the current scenario, eye gaze is used in many aspects of artificial intelligence, machine learning, computer vision etc.

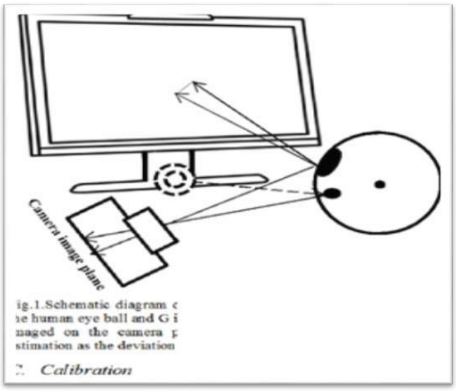


Figure 1

Eye gaze brings human and machine interaction more closer by applying few specific approaches and algorithms. The Script uses a laptop webcam to track a user's gaze location on the screen, to use as a user interface. Starting from Haar Cascade face detection, eye detection and so on, then use least-squares to train a transformation from pupil position to screen coordinates. This algorithm can estimate gaze location in real-time, within approximately 5-25 cm in favourable conditions.

Script uses known face geometry to remove spurious eye and face detections, producing very reliable bounding boxes around the user's eye. Script will be performing the following

Specific Activities:

(1) Starting from formal face and eye detection of user from a favourable distance.
(2) Detecting the reference points for gaze.
(3) Applying virtual reference point algorithm for estimation.
(4) Training the script using pygame UI.
(5) Pupil detection and tracking.

II. EXISTING SYSTEM

- Video-based gaze approaches commonly use two types of imaging techniques: *infrared imaging and visible imaging*. The former needs infrared cameras and infrared light sources to capture the infrared images, while the latter usually utilizes high-resolution cameras for images.
- Compared with the infrared-imaging approaches, visible imaging methods circumvent the aforementioned problems without the need for specific infrared devices and infrared light sources. They are not sensitive to the utilization of

International Journal of Engineering Applied Sciences and Technology, 2019
Vol. 4, Issue 6, ISSN No. 2455-2143, Pages 144-149
Published Online October 2019 in IJEAST (http://www.ijeast.com)

glasses and the infrared sources in the environment. Visible-imaging methods should work in a natural environment, where the ambient light is uncontrolled and usually results in lower contrast images.

- Sugano et al. (2011, 2012, 2013, 2014), has presented an online learning algorithm within the incremental learning framework for gaze estimation, which utilized the user's operations (i.e., mouse click) on the PC monitor.
- Nguyen (2009, 2010), first utilized a new training model to detect and track the eye, and employed the cropped image of the eye to train Gaussian process functions for gaze estimation. In their applications, a user has to stabilize the position of head in front of the camera after the training procedure.
- Williams et al. (2005, 2006, 2007), proposed a sparse and semi-supervised A gaussian process model to infer the gaze, which simplified the process of collecting training data. Generally, it takes a huge amount of data (images) from various subjects to train a system for an accurate and efficient result.

III. DISADVANTAGES OF EXISTING SYSTEM

- The iris centre detection will become more difficult than the pupil centre detection because the iris is usually partially occluded by the upper eyelid
- The construction of the classifier needs a large number of training samples, which consist of the eye images from subjects looking at different positions on the screen under different conditions.
- They are sensitive to head motion and light changes, as well as the number of training samples.
- They are not tolerant to head movement.

IV. PROPOSED SYSTEM

- In this paper, we concentrate on visible-imaging and present an approach to the eye gaze tracking using a web camera in a desktop environment. First, we track the human face in a real-time video sequence to extract the eye region.

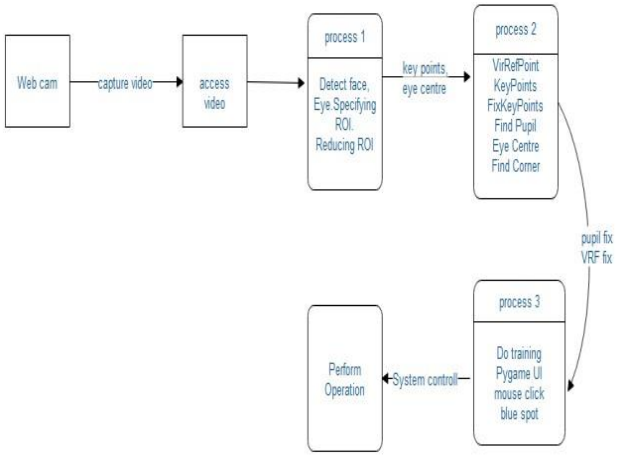
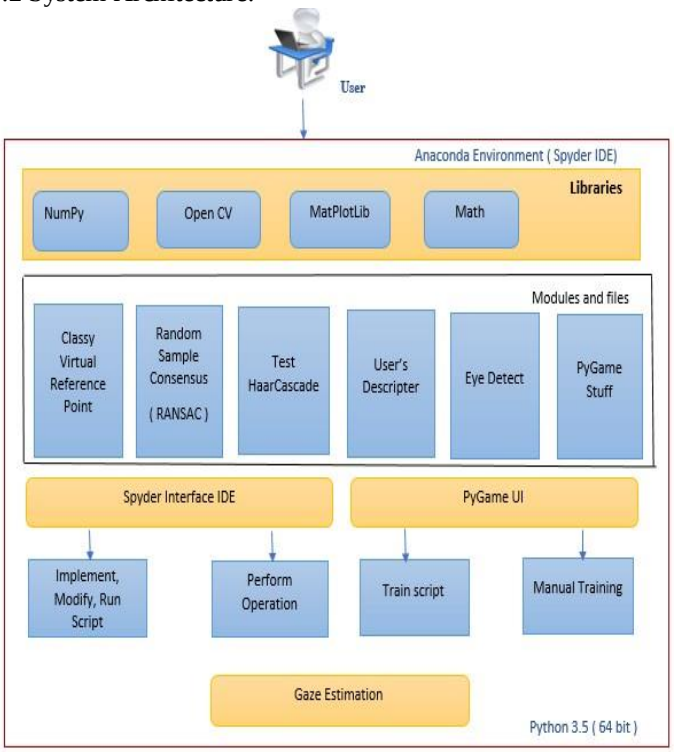


Figure 2: Process of System

4.1 Identification of face and eye region (R.O.I.):

- Algorithm will identify the area of the face where eyes are located. And then reduces the region specified for the purpose of more accurate detection of eyes. And then dram small boxes over ROI for more detailed detection.

4.2 System Architecture:



4.3 Finding Pupil Centres:

- To estimate gaze direction, we must find the pupil centre within the bounding box for each eye. For this, we have implemented a gradient-based method which uses the fact that image gradients at the border of the iris and pupil tend

International Journal of Engineering Applied Sciences and Technology, 2019
Vol. 4, Issue 6, ISSN No. 2455-2143, Pages 144-149
Published Online October 2019 in IJEAST (http://www.ijeast.com)

to point directly away from the centre of the pupil. It works as follows:

- If we make a unit vector di which points from a given image location to a gradient location, then its squared dot product with the normalized gradient vector gi, expressed as (di T gi) 2 , will be maximized if we are at the pupil centre. We do this calculation for each image location and each gradient. The pupil centre c* is the location that maximizes the sum of these dot products:

$$\mathbf{c}^* = \arg\max_{\mathbf{c}} \left\{ \frac{1}{N} \sum_{i=1}^{N} \left(\mathbf{d}_i^T \mathbf{g}_i\right)^2 \right\} ,$$

$$\mathbf{d}_i = \frac{\mathbf{x}_i - \mathbf{c}}{\|\mathbf{x}_i - \mathbf{c}\|_2} , \quad \forall i : \|\mathbf{g}_i\|_2 = 1$$

We only calculate pupil-centre-probability in the dark parts of the image, because the pupil is known to be dark. (To do this, we threshold on pixel brightness, then dilate the image to expand the allowable areas slightly. We must dilate so that our search region includes parts of the pupil where there are bright reflections.) Eyelashes and shadows produce unwanted gradients, which can produce spurious peaks in the pupil-centre probability image. Our desired gradients are fairly strong, so we discard all gradients whose magnitude is below the mean.

This method provides centre of the iris with fairly high reliability, and acceptable accuracy (usually within 3 pixels). However, it is vulnerable to regions which look similar to a pupil in low-resolution images, such as the shadowed blob on the left of the eye.

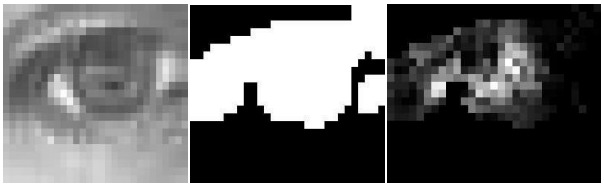


Figure 3: Left to right: A blurred eye image; the areas considered dark enough to be pupil candidates; our estimates of centre probability in the regions that were dark (white=highest probability). Note the spurious peak on the left.

4.4 General flow for Eye Pupil Detection:

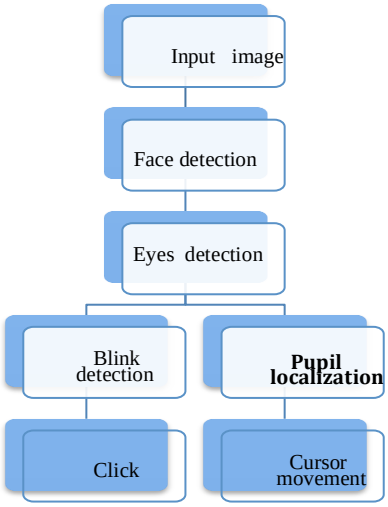


Figure: 4

4.5 Reference Point for Eye Direction:

- To detect the pupil coordinates in terms of an offset from some reference point on the face. The Haar Cascade bounding boxes for the face and eyes are not steady enough to serve as reference points, even when averaged together. So I experimented heavily with several other possible reference point methods: draw a blue dot between the eyes with a dry-erase marker, and apply our single pupil-finding code to find the centre of the blue dot.

4.6 Pupil Probability From Two Eyes:

- We improved the reliability of our estimation by combining multiple probability estimates. Specifically, we designed a method to combine the estimates from the two eyes: we overlay the probability image from the right eye onto the left eye and multiply the two probabilities.
- Eye pupils move together and therefore appear at the same place in the two images. But other (noise producing) parts of the eye are mirrored about the centre-line of the face, and therefore appear at different places in the two images.

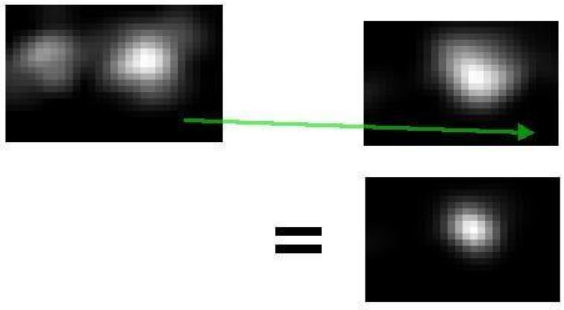


Figure 4: Camera-left eye probabilities are overlaid on camera-right probabilities using the average pupil-topupil vector from previous frames. Their product has less noise

International Journal of Engineering Applied Sciences and Technology, 2019
Vol. 4, Issue 6, ISSN No. 2455-2143, Pages 144-149
Published Online October 2019 in IJEAST (http://www.ijeast.com)

- Combining the two eye estimates was not Straight forward. Because of the eye bounding boxes are inaccurate, they cannot be used to align the probability images. We used the insight that, if the head position is steady, the vector from the left eye's pupil to the right eye's pupil is almost constant as the pupils move around. (Especially when the eyes are focused on a plane, such as a computer monitor.) Therefore, if **any** image region that includes the left eye is shifted by that vector, the left pupil will end up on top of the right pupil.
- Thus, we can combine the eye probability estimates without requiring any absolute reference points on the face. (We blur both images slightly before the multiplication so that an error of a few pixels in the pupil-pupil vector still yields a strong peak near the true pupil.) The pupil-to-pupil vector can be obtained from the noisy estimates of individual pupil positions. We simply applying a very slow moving-average filter to reject noise in the estimate of the vector.
- This method proved very effective. If the pupils are visible to the camera, the joint pupil estimate is within the iris nearly 100% of the time.

4.7 Reference Point:

A perfect reference point should be as stable as possible, but a key insight (which has not yet appeared in any paper I'm aware of) is that the facial reference point does not need to be based on any specific facial feature. With that in mind, I have implemented a "virtual reference point" tracking algorithm, based on many image keypoint descriptors.

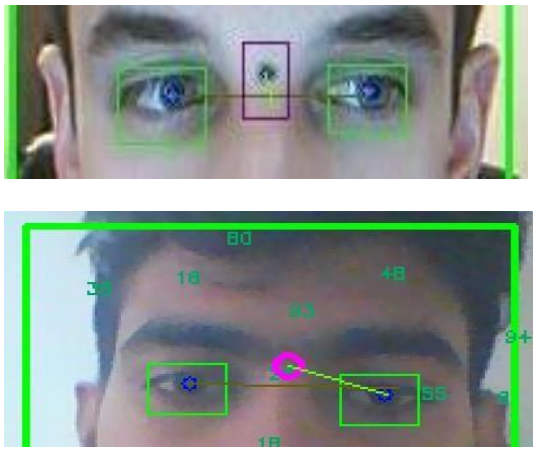


Figure 5: The magenta circle is the virtual reference point.

V. TRAINING THE SCRIPT

- we use the numpy library's least-squares solver to produce a transformation matrix which relates pupil offset (x, y) to screen position (pixelX, pixelY). We also input quadratic features, because the relationship between eye offset and screen location is not entirely linear (as the eye looks more to the side, small movements of the eye correspond to larger movements on the screen).

- So our input feature vector is [x, y, 1, x 2 , y 2 , xy]. We use least squares to train a 2 x 6 matrix H such that: **feature * H = (pixelX, pixelY)**
  Unfortunately, the least-squares fit is strongly degraded if there are outliers in training data (such as when the user's eyes were closed and the pupil position is incorrect). Therefore, we use the **RANSAC (Random Sample Consensus)** algorithm to reject outliers.
- RANSAC works by choosing a random small subset of training data, training H matrix, and using only the training data that is well-described by H to refine H. Many such H candidates are computed (800 in our case), and the best-performing one is chosen. We retain the algorithm after each click by the user. After approximately 10 to 20 clicks, our algorithm can estimate gaze along the top of the screen very well.

Figure 6: Pygame UI to train the script manually

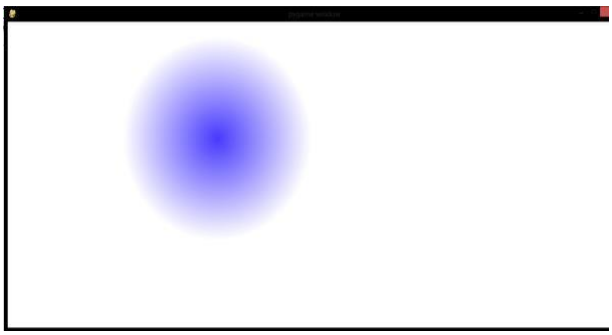


Figure 7: Blue mark indicates the mouse click

- Measurement of Results:

Inadequate lighting conditions, the system performs quite well. In a test of the system with 43 clicks using a webcam with resolution 640 x 480, the median error between predicted and actual click position was 69 pixels, and the mean was 79 pixels. Most of the error was in the vertical direction. This is because the system still has trouble locating the exact centre of the pupil vertically. The horizontal median error was only 28 pixels. With our test setup, the overall mean error of 79 pixels corresponds to 2.8 degrees of angular error in estimating gaze direction. Commercial services which use webcams claim accuracies

International Journal of Engineering Applied Sciences and Technology, 2019
Vol. 4, Issue 6, ISSN No. 2455-2143, Pages 144-149
Published Online October 2019 in IJEAST (http://www.ijeast.com)

of 2 to 5 degrees. So, our system seems to be comparable in quality to commercial eye-tracking systems. The plot is shown below:

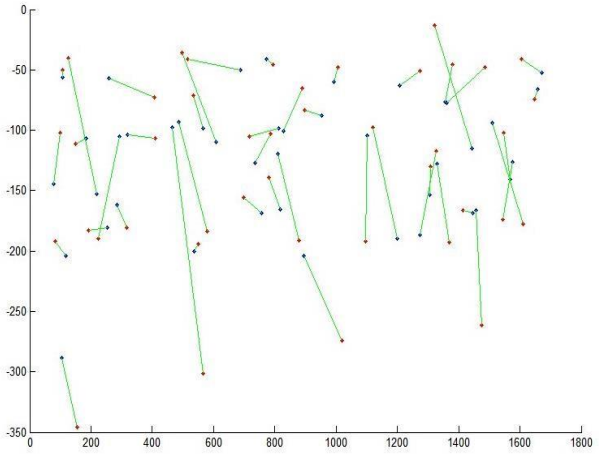
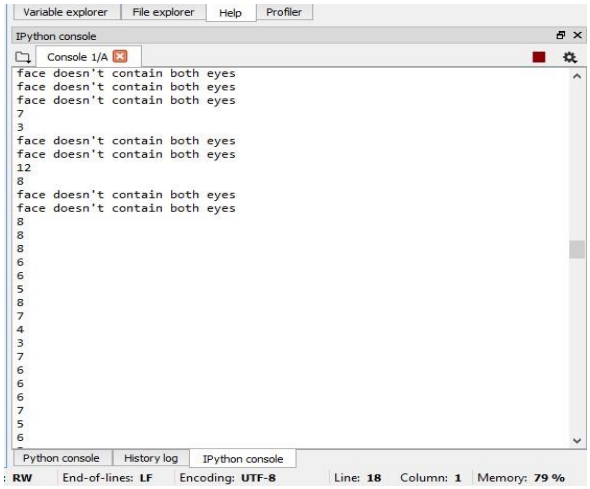
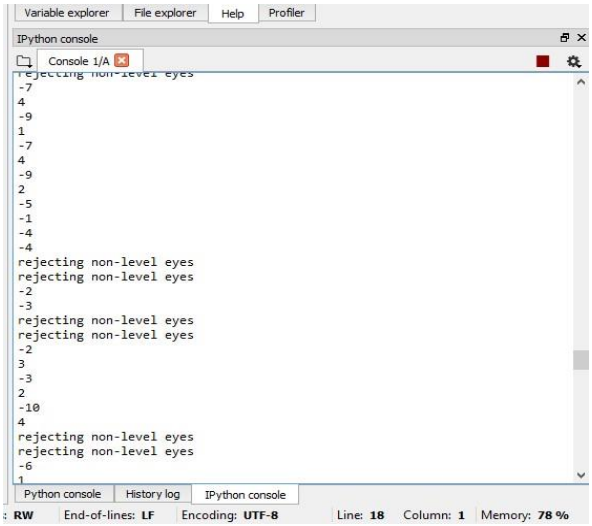


Figure 8: *Red indicates true click position, and blue indicates the system's predicted eye gaze location at the time of that click. The prediction is made by transforming pupil offset with the H matrix discussed previously.*



(a)



(b)

(a) Rejecting frames that does not contain both eyes. (b) Detecting both eyes and rejecting non- level frames.

VI. CONCLUSION & FUTURE WORK

The biggest opportunity for improvement is to allow head movement without retraining. (Neither our algorithm nor commercial systems can handle head movement.) One possibility for this is to estimate the head position and apply geometry. Side-to-side head movement is detected by the current system and could be used to modify the model in real-time. Distance is not directly sensed by webcams, but we believe the length of our pupil-to-pupil vector is a very promising signal for estimating distance changes. Head orientation changes would be harder to correct for, as head rotation produces non-linear changes in the reference point's position relative to the pupils. One option might be to use a haar cascade to detect the user's nose and infer head orientation and then apply either machine learning or facial geometry to correct for head orientation's effect on the reference point.

Another promising possibility for user interfaces which could be used in combination with geometry, is to continuously train the system by assuming the eye is likely to look at salient features on the screen or at mouse click positions during normal computer use. Such a method could provide very large amounts of (noisy) calibration data without requiring the user to manually train the system.

VII. ACKNOWLEDGEMENT

I would like to acknowledge and convey my best regards to Prof. Ching- Ting Tu for guiding me throughout the research at Tamkang University, Taiwan And also to Dean and prof. Hui-Huang Hsu for providing the positive conditions and guide-lines about my desired work.

VIII. REFERENCES

[1] B. L. Nguyen, Y. Chahir, M. Molina, C. Tijus, and F. Jouen,(August 2010), "Eye gaze tracking with free head movements using a single camera," in Proceedings of the Symposium on Information and Communication Technology (SoICT '10), N. T. Giang, N. T. Hai, and H. Q. Thang, Eds., vol. 449 of ACM International Conference Proceeding Series, , ACM, Hanoi, Vietnam, (pp. 108–113).

[2] O. Williams, A. Blake, and R. Cipolla, (June 2006), "Sparse and semi-supervised visual mapping with the S3GP," in Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR '06), (pp. 230–237).

[3] C.-C. Lai, Y.-T. Chen, K.-W. Chen, S.-C. Chen, S.W. Shih, and Y.-P. Hung, (August 2014), "Appearance based gaze tracking with free head movement," in Proceedings of the 22nd International Conference on Pattern Recognition (ICPR '14), Stockholm, Sweden, (pp. 1869–1873).

[4] D. W. Hansen and Q. Ji, (2010), "In the eye of the beholder: a survey of models for eyes and gaze," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 32, no. 3, (pp. 478–500).

[5] F. Lu, T. Okabe, Y. Sugano, and Y. Sato, (2014), "Learning gaze biases with head motion for head pose free gaze estimation," Image and Vision Computing, vol. 32, no. 3, (pp. 169–179).

[6] J.-G. Wang, E. Sung, and R. Venkateswarlu, (October 2003), "Eye gaze estimation from a single image of one eye," in Proceedings of the 9th IEEE International Conference on Computer Vision (ICCV '03), IEEE, (pp. 136–143).

[7] L. Sesma, A. Villanueva, and R. Cabeza, (March 2012), "Evaluation of pupil centre-eye corner vector for gaze estimation using a web cam," in Proceedings of the Symposium on Eye Tracking Research and Applications (ETRA '12), C. H. Morimoto, H. O. Istance, S. N. Spencer, J. B. Mulligan, and P. Qvarfordt, Eds., ACM, Santa Barbara, Calif, USA, (pp. 217–224)

[8] H. Wu, Y. Kitagawa, T. Wada, T. Kato, and Q. Chen, (2007), "Tracking iris contour with a 3D eye model for gaze estimation," in Computer Vision— ACCV 2007: 8th Asian Conference on Computer Vision, Tokyo, Japan, November 18–22, 2007, Proceedings, Part I, Y. Yagi, S. B. Kang, I.-S. Kweon, and H. Zha, Eds., vol. 4843 of Lecture Notes in Computer Science, Springer, Berlin, Germany, (pp. 688–697).

[9] F. Lu, T. Okabe, Y. Sugano, and Y. Sato, (September 2011), "A head pose-free approach for appearance-based gaze estimation," in Proceedings of the 22nd British Machine Vision Conference (BMVC '11), (pp. 126.1–126.11).

[10] E. Skodras, V. G. Kanas, and N. D. Fakotakis, (2015), "On visual gaze tracking based on a single low cost camera," Signal Processing: Image Communication, vol. 36, (pp. 29–42).

[11] Y.-M. Cheung and Q. Peng, (2015), "Eye gaze tracking with a web camera in a desktop environment," IEEE Transactions on Human Machine Systems, vol. 45, no. 4, (pp. 419–430).

[12] R. Valenti, N. Sebe, and T. Gevers, (2008), "Simple and efficient visual gaze estimation," in Workshop on Multimodal Interactions Analysis of Users in a Controlled Environment (MIAUCE), ICMI, no. 3.

[13] W. Zhang, T.-N. Zhang, and S.-J. Chang, (October 2010), "Gazing estimation and correction from elliptical features of one iris," in Proceedings of the 3rd International Congress on Image and Signal Processing (CISP '10), Yantai, China, (pp. 1647–1652).

[14] C. H. Morimoto and M. R. M. Mimica, (2005), "Eye gaze tracking techniques for interactive applications," Computer Vision and Image Understanding, vol. 98, no. 1, (pp. 4–24).

[15] Y. Sugano, Y. Matsushita, Y. Sato, and H. Koike, (2015), "Appearance-based gaze estimation with online calibration from mouse operations," IEEE Transactions on Human-Machine Systems, vol. 45, no. 6, (pp. 750–760).

[16] F. Lu, Y. Sugano, T. Okabe, and Y. Sato, (2015), "Gaze estimation from eye appearance: a head pose free method via eye image synthesis," IEEE Transactions on Image Processing, vol. 24, no. 11, (pp. 3680–3693).

[17] K .-H. Tan, D. J. Krieg man, and N. Ahuja, (2002), "Appearance-based eye gaze estimation," in Proceedings of the Sixth IEEE Workshop on Applications of Computer Vision (WACV '02), IEEE Computer Society, (pp. 191–195).

[18] J. Chen and Q. Ji, (2008), "3D gaze estimation with a single camera without IR illumination," in Proceedings of the 19th International Conference on Pattern Recognition (ICPR '08), IEEE, December, (pp. 1–4).