Explore New Eye Tracking and Gaze Locating Methods

Alexander Kadyrov, Hui Yu, Joe Eyles, and Honghai Liu
School of Creative Technologies, University of Portsmouth, UK

Abstract—**Eye tracking has been used extensively in research, often for plotting the gaze location of research participants. Many of the existing methods on tracking gaze locations, however, require special equipment. This equipment can be very expensive and/or cumbersome to use, restricting the use of it to specialist labs. By producing a method of eye tracking which requires minimal equipment and set up time, eye tracking might be more widely used as a research tool, especially in exploratory areas where the outcomes of the research are not known. This paper introduces two novel eye tracking methods which use only a webcam feed as input and require no specialist equipment. The camera used in this research is a standard webcam built into a normal laptop; any current commercial webcam could be used. Experiments using the proposed method showed more accurate tracking results than some existing eye trackers.**

Index Terms—**Eye tracking, gaze estimation, player experience, circle detection.**

I. Introduction

EYE tracking has a great many applications. However, due to the cumbersome and expensive nature of much of the required equipment, the technology has yet to make its way into our daily lives. There exist a number of methods for eye detection using a webcam, however these are not very efficient and robust processes, and can rely on lengthy initialisation processes.

This paper aims to explore methods by which the required equipment can be reduced to a simple webcam, and the initialisation process can be reduced to just the first few frames of the video (or even omitted altogether), thus allowing a widespread use of eye tracking and gaze location software.

A. Related Work

Many eye detection methods have been proposed in the literature. Many commercial eye detectors require specialist equipment, for example infra red cameras and lasers. They also often require a lengthy initialization and calibration process. Some of these are detailed in [26], [10], [19]. We will be focusing on methods that do not require these, and instead use only a simple camera or image.

A large majority of eye detectors use circle detection as the main tool, as the iris is well known to be a dark circle on a white background, consisting of the white sclera of the eye. The Hough transform was introduced as a feature detection method [23], and has since been modified to be used for circle detection, as detailed in [5], [6], [11], [24], [3], [1] among others. There has also been work done to the Hough transform in order to make the method more efficient [5], [3], [11]. Wavelets were used by [6], [14] to filter or modify the image to prepare it for a circle detection method. [4], [13] have also modified classical wavelet theory to create "curvelets", which can be used directly in circle detection.

Although many eye detection techniques use circles as the primary feature, [25] uses projection functions to look for horizontal and vertical features. [10] uses multiple techniques to locate the eye centres, including Haar-like object detectors, which were proposed in [22], [9]. The local center of mass is then used, and finally a specialized deformable template algorithm finds the center of the eyes.

It is evident that information on eye movement has connection to the emotional, cognitive and psychological state of the human. Attempts to analyse human response use special equipment [15], [16], [17] or even require human annotation [18], that is people are asked to describe their emotional state themselves.

In a review [17] it was stated that eye movement is still not explored in full when assessing human conditions. In this paper we describe a system that is able to provide the same information using just an ordinary internet camera. However, the approach is different from [17] where direction of the gaze was not really used, but instead EEG information was used. The main contribution of the presented paper is in showing that progress can be achieved by using a new, simple, mathematical approach for pupil and iris tracking, with no need for special equipment or arrangement.

II. Methods used in Gaze Estimation

A. Eye Tracking Methods

Although there already exist many eye trackers, the majority of these require special equipment, as before these are mentioned in [26], [10], [19]. In our work we endeavoured to use no special equipment, and hoped to match these devices using only a simple webcam.

1) Isophote Centers: [20], [21] introduce a circle detection method, which is used for eye detection, and proved to be the best contemporary method. We implemented their method so as to give an existing method from the literature against which we could compare our proposed methods.

The papers consider isophotes, or lines of constant intensity in the image. If we let $f(x, y)$ be the intensity at the pixel (x, y), then this can be characterized by the isophote property:

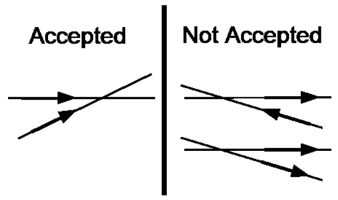

Fig. 1. Possible accepted and not accepted formations of gradients.

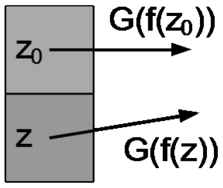

Fig. 2. The variables used in the Gradient Intersection method.

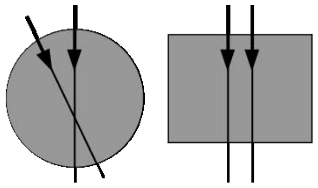

Fig. 3. An example of the gradients intersections in a circle but not a square.

$f(x, y) = f_0$. Every pixel is assumed to be on an isophote, the curvature is calculated for that pixel, and then the center of a circle with equivalent curvature is saved in an accumulator. The curvature, κ, is given by $\frac{d^2y}{dx^2}$. This can be calculated by implicit differentiation, using the isophote property, and gives:

$$\kappa = -\frac{f_y^2 f_{xx} - 2f_x f_{xy} f_y + f_x^2 f_{yy}}{(f_y^2 + f_x^2)^{\frac{3}{2}}}.$$

It is known that the reciprocal of this gives us the radius of a circle with curvature κ. After some calculations [20], the estimate given by the pixel (x_0, y_0) will be saved in the accumulator at the point:

$$(x, y) = (x_0, y_0) - \frac{(f_x, f_y)(f_x^2 + f_y^2)}{f_y^2 f_{xx} - 2f_x f_{xy} f_y + f_x^2 f_{yy}}. \qquad (1)$$

It is then recommended that the curvature, given by:

$$C = \sqrt{f_{xx}^2 + 2f_{xy}^2 + f_{yy}^2},$$

should be used to weight the estimates. We used the curvature to create an upper and lower bound on the curvature that was accepted. Bounds of 3 and 16, corresponding to radiuses of approximately 4 and 20, were found to be effective. Finally, the accumulator is convolved with a Gaussian kernel so that groups of estimates form single estimates.

2) Gradient Intersection: This section details a similar but not equivalent method used for eye tracking. We created and propose this method as an alternative to the aforementioned methods.

The method is actually a circle detector, as the iris is well known to be a dark circle. At the edge of a dark circle, all the gradients will be pointing inwards, approximately towards the center of the circle. This fact is exploited by finding the intersections of the gradients of adjacent pixels, which should all point towards the center of the circle they are on. If a pixel is not on a circle, then the adjacent gradients are unlikely to point towards a common center, so these can be discarded. These gradients can be seen in Fig. 3.

First the image undergoes a Gaussian blur to reduce the effects of noise and other imperfections. Then the gradient of the image is found. The computer then loops through every pixel in the image, and for each pixel the four directly adjacent pixels are considered one at a time. The intersection of the gradients of these two adjacent pixels is calculated. Let, as shown in Fig. 2 $z \in \mathbf{C}$ be the position of the pixel, and $f(z)$ be the intensity of the image at pixel z. Let $G(f(z))$ be the gradient of $f(z)$ at z; this vector will be treated as a complex number, that is z and $G(f(z)) \in \mathbf{C}$ to simplify further formula. This gives two lines, parametrised by $\alpha, \beta \in \mathbf{R}$, originating at each pixel and following along the gradient:

$$w = z_0 + \alpha G(f(z_0)),$$

and

$$w = z + \beta G(f(z)),$$

where $w \in \mathbf{C}$. It can be shown that the values of α and β, assuming intersection of the lines, are:

$$\alpha = \Re\left(\frac{z + \beta G(f(z)) - z_0}{G(f(z_0))}\right),$$

and

$$\beta = \frac{\Im(z_0 - z) - \Im(G(f(z_0)))\Re(\frac{z - z_0}{G(f(z_0))})}{\Im(G(f(z))) - \Im(G(f(z_0)))\Re(\frac{G(f(z))}{G(f(z_0))})},$$

here $\Re$ and $\Im$ denote the real and imaginary parts, respectively. Note that we first calculate β, and then use that to calculate α. These can then then be used to compute the intersection point. If the intersection is in the positive direction, then the intersection point is added to an accumulator Fig. 1. This occurs if $\alpha > 0$ and $\beta > 0$. The estimates are weighted by the absolute size of the gradient $G(f(z))$. The accumulator then undergoes a Gaussian blur to group adjacent estimates into one estimate. The maximum is then found and this is considered the eye estimate.

3) Ray Image through Convolutions: We also present another circle detection method - the ray image method. It is an alternative to the methods already stated, and its computation is based on Fast Fourier Transform (FFT).

Let, as before, the image be a function $f(z) \in \mathbf{R}$ for $z \in \mathbf{C}$, and let $G(z)$ be the gradient of $f(z)$ at z, and the gradient is again treated as a complex number. Obviously, we can retain interpretation of complex numbers as 2D vectors and as points in the plane when it is convenient. Please notice a modification:

in this section the gradient is denoted by $G(z)$ rather than $G(f(z))$.

We will first introduce a function $R : \mathbf{C} \to \mathbf{R}$, whose local maxima will manifest centres of sought circles. We will then describe the motivation behind this and discuss the scalability of the method.

Let us define:

$$R(z_0) := \sum_z K\left(e^{i\angle[z_0 - z, G(z)]}, |z_0 - z|\right) P\left(|G(z)|\right),$$

here $\angle[v, w]$ is the angle between a vector v and a vector w, $|w|$ is the complex modulus of w, and K and P are to be defined. Here K is a function of the angle between the vector $z_0 - z$ and $G(z)$, and the distance between z and z_0. It is intended that a point z_0 that lies almost along the ray spanned on $G(z)$, and is close to z give a large value of K. Conversely, if z_0 is either very far away from z or does not lie along the vector $G(z)$, then the value of K should be very small. By doing this, any z_0 that is in the center of a circle will have lots of close pixels with gradients that point towards z_0, and so a very large value of K. Any z_0 that does not have these properties will have a small value of K. The function P is intended to be somehow proportional to the magnitude of $G(z)$, thus having a large value for circles that are in focus, and a smaller value for circles that are out of focus.

For the purpose of defining K and P, it is convenient to introduce a few notations:

$$h(z) := \frac{z}{|z|} = e^{i\arg(z)},$$

$$g(z) := \frac{G(z)}{|G(z)|} = e^{i\arg(G(z))}.$$

Using these functions we can re-write $R(z_0)$ as:

$$R(z_0) := \sum_z K\left(g(z)\overline{h(z_0 - z)}, |z - z_0|\right) P\left(|G(z)|\right);$$

here $\overline{w}$ is the complex conjugate of w. Let us suppose that we can split K into factors:

$$K(w) = K_1\left(\frac{w}{|w|}\right) K_2(|w|) = K_1\left(e^{i\arg(w)}\right) K_2(|w|).$$

Please note that K_1 takes a form of the angle between $z_0 - z$ and $G(z)$ as its argument, while K_2 takes $|z_0 - z|$ as its argument. We will define

$$K_1(u) := a_0 + a_1\cos(\arg(u)) + a_2\cos(2\arg(u)) + \ldots$$

and choose the constants

$$a_0 = 0,\ a_1 = 1,\ a_2 = 0.77,\ a_3 = 0.5,\ a_4 = 0.15,$$

and $a_i = 0 \quad (\forall i > 4)$. This gives a function shown by its graph in Fig. 4.

We will define $K_2(r) := r^\beta$ for some $\beta \in \mathbf{R}$, $\beta < 0$, and choose $\beta = -1$. Recall that here $r = |z - z_0|$; if $r = 0$ we have to take $K_2(0) := 0$. Lastly we define $P(r) := r^\alpha$ for some $\alpha \in \mathbf{R}, \alpha > 0$, and choose $\alpha = 0.75$.

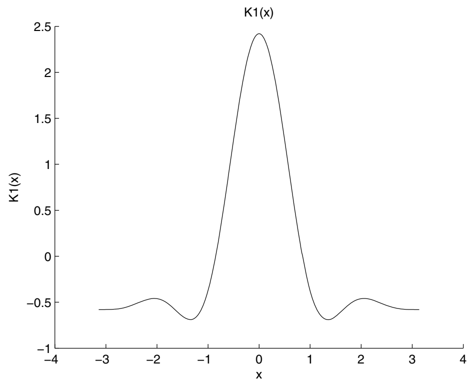

Fig. 4. The graph for K_1 vs. angle.

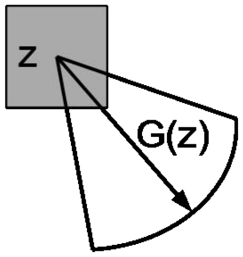

Fig. 5. The domain over which z has a positive influence.

This choice of K gives us a domain for each z where it will have a positive influence on $R(z_0), \forall z_0$. Fig. 5. This picture elucidates the name given to the method. Although there are many functions that could have been chosen for K_1, K_2 and P, these particular functions were chosen with a view to the speed of the method.

After all the definitions, we can analyse the proposed formulae. We notice that K_1 can be written as the real part of a polynomial:

$$K_1(u) = \Re\left(a_0 + a_1 u + a_2 u^2 + \ldots\right).$$

By splitting up K, and using the form of K_1 as a polynomial expansion, this can in turn be re-written as:

$$\sum_z \Re\left(\sum_{k=0}^{n} a_k g^k(z)\overline{h^k(z - z_0)} K_2\left(|z - z_0|\right) P\left(|G(z)|\right)\right),$$

which can be presented through n cross-correlations, denoted with '$\star$':

$$R = \Re\left(\sum_{k=0}^{n} a_k \left[g^k(.)P(|G(.)|)\right] \star \left[\overline{h^k(.)}K_2(.)\right]\right).$$

As cross-correlations can be computed through FFT, this formula is computationally efficient.

One of the desired properties of this model is scalability, so that no matter the size of the image, $R(z_0)$ is somewhat the same. This is required due to the method needing to find the iris regardless of its size. We proved scalability for a continuous version of the method. We will consider an integral form of $R(z_0)$, given by

$$R(z_0) =$$

$$\iint K_1\left(g(z)\overline{h(z_0 - z)}\right) K_2\left(|z_0 - z|\right) P\left(|G(z)|\right) dx\, dy.$$

Let the image, denoted $f(z)$ be scaled by $\rho > 0$ to give $f_\rho(z) = f(\frac{z}{\rho})$. This gives us a new set of scaled functions:

$$G_\rho(z) := \frac{G(\frac{z}{\rho})}{\rho},$$

$$g_\rho(z) := \frac{G_\rho(z)}{|G_\rho(z)|} = \frac{G(\frac{z}{\rho})}{|G(\frac{z}{\rho})|}.$$

Note that $h(z)$ does not need a scaled version. This means that for $w = x + iy$ in our scaled image, we now have:

$$R_\rho(w_0) =$$

$$\iint K_1\Big(g_\rho(w)\overline{h(w_0 - w)}\Big) K_2(|w_0 - w|) P(|G_\rho(w)|) dxdy.$$

Using the substitution $w = z\rho$, we can derive $R_\rho(\rho z_0) = \rho^{1+\alpha+\beta} R(z_0)$. Hence the method is scalable. In particular, with our chosen constants, we have

$$R_\rho(\rho z_0) = \rho^{0.75} R(z_0).$$

B. Template Matching Methods Used

1) Square Difference: One of the template matching techniques used was simply the OpenCV function cvMatchTemplate(), described in [2]. This function has a couple of different modes, but we used it in Square Difference mode. This function takes three main arguments: an image, denoted I, a template image, denoted T, and a resulting image, denoted D. We must have both the height and the width of I larger than that of T, and also that D has dimensions $(I_{width} - T_{width} + 1) \times (I_{height} - T_{height} + 1)$. The function then finds the closest match for T in I according to certain criteria. The matches are then displayed in D, found by the formula:

$$D(z) = \sum_{\{z_i \in T | z_i + z \in T\}} \left(T(z_i) - I(z + z_i)\right)^2.$$

This positions T on I at every possible location (not allowing T to extend outside of I), and the sum of the square difference of every covered pixel in I with the corresponding pixel in T is found. These differences are then stored in R, and the estimated position of T in I can be found as the minimum value in D. Fig. 6 shows this method in action. The template is searched for in the original image, and the result can be seen in Fig. 6. Note that this window has been normalized so as to make the result clearer for us to see.

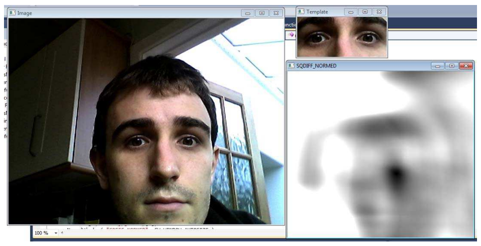

Fig. 6. On the left we have the original image, in the top right we have the template, and in the bottom right we have the result of the square difference method.

2) Fast Robust Correlation and Orientation Correlation: We have also used the more complicated Fast Robust Correlation and Orientation Correlation, which were proposed in [8], [7] and are fast due to usage of FFT. Furthermore, the Orientation Correlation method allows even faster computation [12] than using ordinary FFT approach, which is important for real time systems.

It aims to have both speed of computation, using convolutions, and insensitivity to small deviations from the assumptions. It is important to match to the face regardless of eye movement inside the eye socket. In other words, an ideal template matching technique would ignore a mismatch in the eye area, paying more attention to the larger surrounding area.

3) How the Template Matching was used: The Square Difference template matching method was predominantly used to complement the Isophote and Gradient Intersection methods, whilst Orientation Correlation was primarily used alongside the Ray Image through Convolutions method.

The first frame of the recorded or live webcam video is displayed, and the user is asked to draw a box around the eye region. The image in this box then becomes the template, and is found in each subsequent frame. Because the eyes can safely be assumed to lie within the resultant region, found by template matching, we can limit our search for eyes to just this region. This not only speeds up the method, but also increases the accuracy, as any unwanted shapes in the background are not considered. The template matching also tells us how the head moves, which is necessary to predict gaze location.

C. Gaze Estimation

We propose a simple gaze estimation model. When estimating the gaze location, the eyes and head were assumed to move in a flat plane, parallel to the screen, and the shoulders and neck were assumed to have little or no movement. These assumptions are acceptable, as when looking at a computer screen, the head has to move only very slightly to view the whole screen. An initialisation procedure was also included, in which the person would look at the center of the screen, and keep their head still for a small number of frames. This gave the initial eye position. This also allowed us to set up the template of the eye region, and find its initial position. Both eyes were considered separately, and the average gaze location was found.

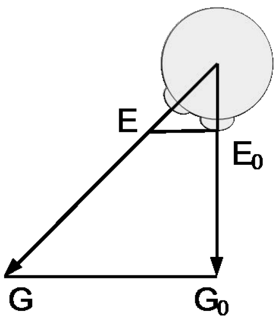

Fig. 7. The movement of the gaze, with the head stationary and eye moving.

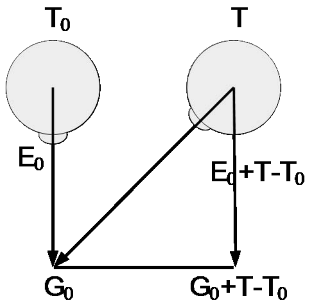

Fig. 8. The gaze remaining stationary while the head and eyes move.

Let E be the eye position, E_0 be the initial eye position, T be the template position, T_0 be the initial template position, G be the gaze position, and G_0 be the initial gaze position. Note that we know the initial gaze position is the center of the screen.

Supposing the head remains stationary, we could calculate G by using the displacement of E from E_0, see Fig. 7. Call r the approximate distance the eye has to move to look from the center of the screen to the side of the screen, and s the dimensions of the screen. Then, as we are assuming E is moving in a plane, we have

$$G = \frac{(E - E_0)s}{2r} + G_0.$$

The head, however, may not remain stationary, but since we are assuming it moves only slightly, and in a plane, we can simply move G_0 and E_0 by an appropriate amount. This yields

$$G = \frac{(E - (E_0 + T - T_0))s}{2r} + (G_0 + T - T_0).$$

Note that here we are using $T - T_0$ as an approximation of the head movement, see Fig. 8. Although somewhat crude, this formula provides us with the approximate gaze location and it suffices for applications presented in the next section.

III. Experiment and Discussion

The three eye tracking methods discussed in detail were implemented, and are compared in this section. A simple output from each method is shown in Fig. 9.

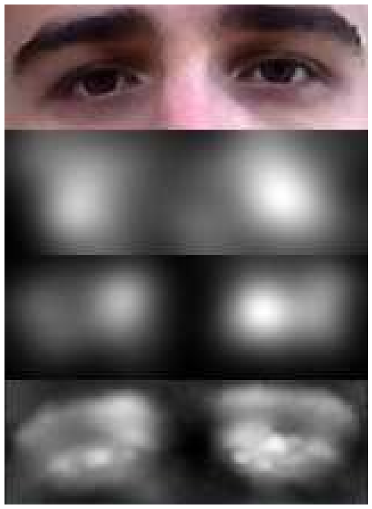

Fig. 9. From top to bottom row: the original image, the result of the Isophote method, the result of the Gradient Intersection method, the result of the Ray Image through Convolutions method.

To compare the accuracy of the methods, the subjects were asked to follow a moving point. The point started at the center, moved to the left of screen (from our point of view), then to the top, the bottom and finally the right. Between each step the point returns to the center of the screen. Each video contained 80 frames, and four subjects were recorded.

The subjects were then asked to click on the center of both the left and right eyes in each frame; this created the true values for the eye positions. The error was calculated as the Euclidean distance between the true position and estimated position. No distinction was made between the error for the left eye, and the error for the right eye. This allowed each frame to yield two error estimates.

The subjects would occasionally blink, and were then asked to put the true eye position for that frame at the center of the eye lids. This would create a large error whenever the subject blinked. To remove these outliers (as we are testing for accuracy in estimating pupil position) the standard deviation was found, and everything two standard deviations away from the mean was removed.

The mean errors of each method before and after removing the outliers are shown in Table. I. It can be seen that the Gradient Intersection method is an improvement on the Isophote method, however the Ray Image through Convolutions Method is significantly more accurate. Note that in the webcam images, the pupil has a diameter of approximately 4 pixels. This means that although the error for the Ray Image through Convolutions method is stated as approximately 3.5 pixels, this could be down to discrepancies in the true values created by the subject. Note also that an error of 14.7 (found for the Isophote Method) from the center of the pupil would still lie within the area of the iris.

Ray Image through Convolutions was implemented in Matlab, whilst the Gradient Intersection and Isophote method

TABLE I

THE MEAN ERROR FOR EACH METHOD, WITH AND WITHOUT THE OUTLIERS.

Method	With outliers	Without outliers
Isophote	15.8449	14.7564
Gradient Intersections	10.6880	9.8338
Ray Image	3.6014	3.4465

were implemented in C++. In addition the programs were not refined. This meant that the speed of the methods could not be accurately compared.

The difference between the methods compared in Table I, can be explained. It is evident that using derivatives of images and the differences of the derivatives is error prone. The Isophote method employs second derivatives and then differences between them (1). The Gradient Intersection method employs first derivatives and then differences between them when finding the intersection of the two lines. The last method, the Ray Image, uses the first derivatives, that is a gradient, but it does not take differences between them. One can see that for these reasons the obtained ranking for the methods is naturally expected.

IV. CONCLUSION

This paper proposed two novel and accurate eye tracking methods: Gradient Intersections and Ray Image methods. The Gradient Intersections method is an improvement of the existing Isophote method. And the Ray Image method achieves the high accuracy through calculating image convolutions. The speed of the Fourier Fast Transform was utilised to allow the method to run in real time. This method was then coupled with the template matching method Orientation Correlation. Experiments have been conducted showing the advantages of the proposed methods, especially the Ray Image method has achieved much higher accuracy then other compared methods. In future, we would conduct more experiments and compare with state-of-art methods. In addition, we plan to test this method in real-time application including designing some gestures for human-machine interactions etc.

REFERENCES

[1] TJ Atherton and DJ Kerbyson. Size invariant circle detection. *Image and Vision computing*, 17(11):795–803, 1999.

[2] G. Bradski and A. Kaehler. *Learning OpenCV: Computer vision with the OpenCV library*. O'Reilly Media, 2008. Book found online at http://www.cse.iitk.ac.in/users/vision/dipakmj/papers.

[3] J. Cauchie, V. Fiolet, and D. Villers. Optimization of an hough transform algorithm for the search of a center. *Pattern Recognition*, 41(2):567–574, 2008.

[4] H. Chauris, I. Karoui, P. Garreau, H. Wackernagel, P. Craneguy, and L. Bertino. The circlet transform: A robust tool for detecting features with circular shapes. *Computers & Geosciences*, 37(3):331–342, 2011.

[5] ER Davies. A modified hough scheme for general circle location. *Pattern Recognition Letters*, 7(1):37–43, 1988.

[6] T. D'orazio, C. Guaragnella, M. Leo, and A. Distante. A new algorithm for ball recognition using circle hough transform and neural classifier. *Pattern Recognition*, 37(3):393–408, 2004.

[7] AJ Fitch, A. Kadyrov, W.J. Christmas, and J. Kittler. Orientation correlation. In *British Machine Vision Conference*, volume 1, pages 133–142, 2002.

[8] A.J. Fitch, A. Kadyrov, W.J. Christmas, and J. Kittler. Fast robust correlation. *Image Processing, IEEE Transactions on*, 14(8):1063–1073, 2005.

[9] N. Guil and EL Zapata. Lower order circle and ellipse hough transform. *Pattern Recognition*, 30(10):1729–1744, 1997.

[10] I. Ibrahim and K. Jin. A 2d eye gaze estimation system with low-resolution webcam images. *EURASIP Journal on Advances in Signal Processing*, 2011.

[11] J. Illingworth and J. Kittler. A survey of the hough transform. *Computer vision, graphics, and image processing*, 44(1):87–116, 1988.

[12] A. Kadyrov and M. Petrou. The invaders' algorithm: Range of values modulation for accelerated correlation. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 28(11):1882–1886, 2006.

[13] J. Ma and G. Plonka. The curvelet transform. *Signal Processing Magazine, IEEE*, 27(2):118–133, 2010.

[14] S. Mallat. Wavelets for a vision. *Proceedings of the IEEE*, 84(4):604–614, 1996.

[15] A. Metallinou, M. Wollmer, A. Katsamanis, F. Eyben, B. Schuller, and S. Narayanan. Context-sensitive learning for enhanced audiovisual emotion classification. *Affective Computing, IEEE Transactions on*, 3(2):184–198, April-June 2012.

[16] S. Pun, Y. Gao, P. Mak, H. Ho, K. Che, H. Ieong, H. Wu, M. Vai, and M. Du. Galvanic intrabody communication for affective acquiring and computing. *Affective Computing, IEEE Transactions on*, 3(2):145–151, April-June 2012.

[17] M. Soleymani, M. Pantic, and T. Pun. Multi-modal emotion recognition in response to videos. *Affective Computing, IEEE Transactions on*, 3(2):211–223, April-June 2012.

[18] J. Tang, Y. Zhang, J. Sun, J. Rao, W. Yu, Y. Chen, and A. Fong. Quantitative study of individual emotional states in social networks. *Affective Computing, IEEE Transactions on*, 3(2):132–144, April-June 2012.

[19] D. Torricelli, S. Conforto, M. Schmid, and T. DAlessio. A neural-based remote eye gaze tracker under natural head motion. *Computer methods and programs in biomedicine*, 92(1):66–78, 2008.

[20] R. Valenti and T. Gevers. Accurate eye center location and tracking using isophote curvature. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pages 1–8. IEEE, 2008.

[21] R. Valenti, N. Sebe, and T. Gevers. Combining head pose and eye location information for gaze estimation. *Image Processing, IEEE Transactions on*, (99):1–1, 2012.

[22] P. Viola and M.J. Jones. Robust real-time face detection. *International journal of computer vision*, 57(2):137–154, 2004.

[23] B.G. Weber, M. Mateas, and A. Jhala. Applying goal-driven autonomy to starcraft. In *Proceedings of the Sixth Conference on Artificial Intelligence and Interactive Digital Entertainment*, 2010.

[24] HK Yuen, J. Illingworth, and J. Kittler. Detecting partially occluded ellipses using the hough transform. *Image and Vision Computing*, 7(1):31–37, 1989.

[25] Z.H. Zhou and X. Geng. Projection functions for eye detection. *Pattern Recognition*, 37(5):1049–1056, 2004.

[26] Z. Zhu and Q. Ji. Novel eye gaze tracking techniques under natural head movement. *Biomedical Engineering, IEEE Transactions on*, 54(12):2246–2260, 2007.