Real-Time Eye Tracking System with Parallel Image Processing

Bo-Chun Chen
coslate@media.ee.ntu.edu.tw
Credit Program on Colleges of Electrical and Computer Engineering and Computer Science
The Center for Continuing Education and Training
National Chiao Tung University
HsinChu, Taiwan

ABSTRACT

Eye tracking system has high potential as a natural user interface device; however, the mainstream systems are designed with infrared illumination, which may be harmful for human eyes. In this paper, a real-time eye tracking system is proposed without Infrared Illumination. To deal with various lighting conditions and reflections on the iris, the proposed system is based on a continuously updated color model for robust iris detection. Moreover, the proposed algorithm employs both the simplified and the original eye images to achieve the balance between robustness and accuracy. Multiple parallelism techniques including TBB, CUDA, POSIX and OpenMP can be used to speedup the process. The Experimental results are targeted to show that the proposed system can capture the movement of user's eye with standard deviation smaller than 10 pixels compared to ground truth, and the processing speed can be up to 30fps.

KEYWORDS

Eye tracking, eye localization, eye movement, Parallel Programming, POSIX, CUDA, OpenMP, TBB

ACM Reference Format:
Bo-Chun Chen. 2018. Real-Time Eye Tracking System with Parallel Image Processing. In *Woodstock '18: ACM Symposium on Neural Gaze Detection, June 03–05, 2018, Woodstock, NY*. ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/1122445.1122456

1 INTRODUCTION

Eye tracking tracking is a highly potential technique for natural user interface, especially for wearable devices. Most existing wearable eye trackers employ infrared illumination to achieve robust performance by stable lighting conditions with infrared light sources close to the eyes. Since infrared light is invisible, it does not distract users, and the pupil becomes obvious and easy to detect under infrared illumination. However, infrared light sources illuminating eyes in such a close distance may cause harms to eyes. Radiation in the near infrared light is the most hazardous and is transmitted by the optical components of eyes [8]. It has higher transmission rate and lower absorption rate than visible light, and near infrared light can eventually reaches the retina while visible light is always absorbed by cornea [5]. If one is overexposed to infrared light, it may cause the thermal retina burn [1].

In this work, we propose a new solution with parallel programming for eye tracking system without infrared illumination. To deal with the variation of uncontrolled lighting conditions and maintain the accuracy as well, several strategies are proposed. First, an iris color model is utilized for robust iris detection, which is continuously updated to address various lighting conditions. Second, the proposed system employs a simplified and original eye images at the same time as a kind of multi-scale algorithm. The simplified image is employed to detect coarse seed point and mask impossible feature points. Third, a modified Starburst algorithm is proposed with an iris mask for eye images without infrared illumination. The first and second part includes several filter and morphological operation, which can be speedup by parallelism, while the third part is a sequential procedure which has limited parallelism.

2 STATEMENT OF THE PROBLEM

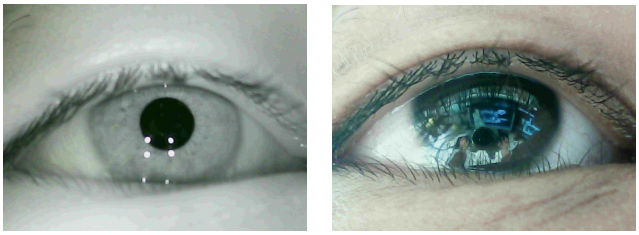
Most application of an eye tracking system is to map information of an eye to a gaze point, which can be used to build attention model or some user interface. The features of an eye must be stable and same when we gaze at same point, different when we gaze at different gaze point, that is, to have high intra-class similarity and low inter-class similarity. Research in the past detect the center of pupil as the feature of an eye according to the fact that infrared light can make the image of pupil clear, noise-free and thus very easy to detect[3][6] as shown in Fig. 1(a). However, without infrared illumination, the most prominent feature of an eye is the limbus, which is the boundary of sclera and iris. Fig. 1(b) shows the eye image without infrared illumination. Due to the reflection in the iris, it is hard to detect the pupil. The only choice is to detect the cornea center because of being concentric with pupil. Unfortunately, the iris is always cropped by eyelids and corners of the eye. If one wants to detect the entire iris region, the best way to do is to open his/her eyes as much as possible, and it would be too tired to use a system. Instead, we only need a stable and distinguishable point to represent the position of an eye. So the problem is to find a way to represent the center of iris with high intra-class similarity and low inter-class similarity as our feature of an eye.

3 PROPOSED APPROACHES

Fig. 2 shows the overview of the whole system. The input of the system is an eye image. Different from the eye images acquired under infrared illumination, the eye images under normal lighting

Woodstock '18, June 03–05, 2018, Woodstock, NY

ACM ISBN 978-1-4503-9999-9/18/06…$15.00
https://doi.org/10.1145/1122445.1122456



(a) (b)

Figure 1: (a) With infrared illumination. (b) Without infrared illumination.

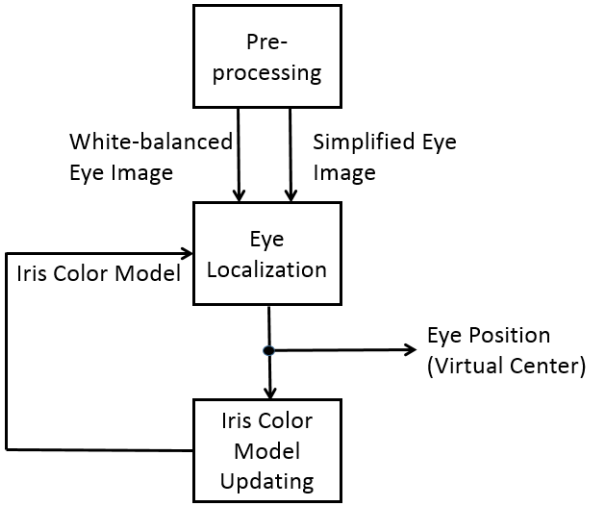
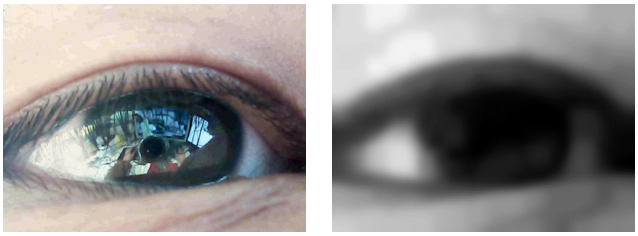


Figure 2: System overview

conditions usually contain noise from lighting condition variation and reflection of environment, as shown in Fig. 3(a). Therefore, the first step is image pre-processing, which aims to filter out the distortion and noise of input eye image. Since the color information is critical to the whole system, we correct the color with automatic white balance to generate image I_E. Moreover, a simplified gray-scale eye image $\hat{I_E}$ is also generated with histogram equalization, morphological gray opening operation, and Gaussian blurring operation to reduce the reflection inside the iris, as shown in Fig. 3(b). As mentioned in Section 1, both these two images are employed in the eye localization as a kind multi-scale processing: the simplified image is used to generate rough detection and masking information robustly, while the white-balanced image is utilized to generated accurate eye location accordingly. An iris color modeled is also maintained in this system to address various lighting conditions. While the iris color model keeps improving itself, the result of eye position, the virtual center, is also getting more accurate, and vice versa. The details of the algorithm are described in the following subsections.



(a) (b)

Figure 3: (a) White-balanced eye image I_E. (b) Simplified eye image $\hat{I_E}$.

3.1 Eye Localization

The purpose of eye localization is to find a stable feature that can represent the location of an eye with high intra-class similarity and low inter-class similarity. We define the stable feature point as a "virtual center." The virtual center in our system is not the location of pupil but is the center of the region surrounded by the feature points at eyelids and left/right sides of limbus. The method of feature point detection is based on Starburst algorithm [4], which is originally invented to detect the boundary of one's pupil under infrared illumination. Because the pupil is invisible under visible light, it is proposed to change the target to limbus and eyelids.

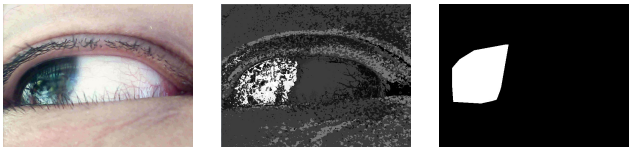
In order to make Starburst algorithm robust enough to detect feature points under visible light and have the ability to adapt to variant lighting conditions, two distinct features are developed and added. The first added feature is the automatic seed point finding algorithm. In the original Starburst algorithm, the seed point is picked as the center point of a frame. In order to automatically find the seed point with illumination invariant property, a color model with 2D h-s (hue-saturation) histogram is constructed for the iris region. During initialization, the rough iris region is estimated by calculating the moment center of the binarized version of $\hat{I_E}$ followed by a region growing process. Within the rough iris region, the h-s histogram of I_E is calculated as the initial iris color model. After the iris color model is established, for an input eye image I_E, the back projection of the iris color model is derived by replacing the value of each pixel (i, j) with the corresponding normalized bin value in the h-s histogram. That is,

$$I_{BP}(i, j) = H(h_{i,j}, s_{i,j}), \tag{1}$$

where the higher value means that the corresponding position has a higher probability to be a part of the iris region, as shown in Fig. 4(b). The seed point is then generated with binarization and moment center calculation. In the following frames, in order to deal with various lighting conditions, the iris color is updated every specified number of frames. To do so, a rough iris mask M_{RI} is generated by finding the convex hull of the binarized $\hat{I_E}$, as shown in Fig. 4(c). A new h-s histogram of I_E is calculated inside the mask M_{RI}. The back projection I_{BP} is then generated as well as an index called iris ratio, which is defined as.

$$\text{Iris Ratio} = \frac{\eta}{\sigma}, \tag{2}$$

where η means the sum of probability of iris point inside the mask M_{RI}, and σ means the sum of probability of iris point outside the



(a) (b) (c)

Figure 4: Hue-saturation histogram model refreshment (a) Input eye image. (b) Back projection I_{BP}. (c) Rough iris mask M_{RI}.

mask. Greater iris ratio comes a more representative model, and we discard those 2D h-s histogram models with low iris ratio.

The second feature added is the iris mask that is used to mask out the reflection in iris region and other impossible feature points to enhance Starburst algorithm. As shown in Figs. 5(a)–(c), the iris region is estimated with projections of the simplified image $\hat{I_E}$. Eyelid fitting is then done by modeling the upper and lower eyelids as two parabolas. With the simplified image $\hat{I_E}$, the valley-peak field, Fig. 5(f), is generated by calculating the convex-hull of the extracted iris rectangle with the biggest circle in it, as shown in Fig. 5(e), and the pixels with large horizontal gradient but low vertical gradient, as shown in Fig. 5(d). We then use the eyelid feature points outside the valley-peak field to fit upward and downward parabola functions, as shown in Fig. 5(g). The intersection of the region inside the eyelid and the estimated iris region as Fig. 5c forms the iris mask M_I for our modified Starburst algorithm described as Algorithm 1, and the feature points are then generated as shown in Fig. 5(h), where the navy blue and light blue feature points represent the inliers of upper and lower eyelid parabolas individually, and green feature points represent the limbus feature points detected by our modified Starburst algorithm 1. All the feature points will be used to form convex hull as shown in Figs. 5(i)–(j), and this is how the accurate iris mask M_{AI} generated. Finally, the location of the virtual center (X_e, Y_e) is derived from the moment center of the inverted version of $\hat{I_E}$ masked by M_{AI}, as shown in Figs. 5(k)–(l) and the following equations.

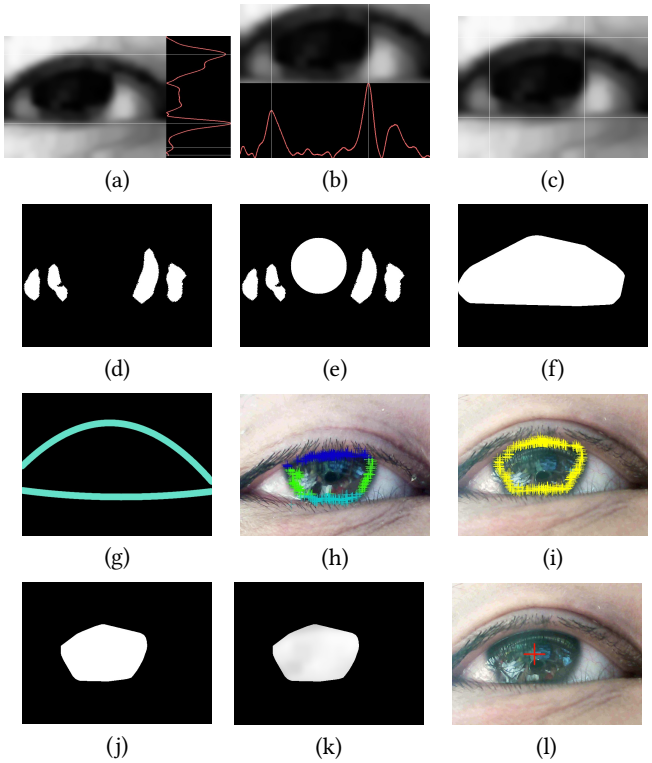
$$m_{i,j} = \sum_{x,y} (255 - \hat{I_E}) \cdot M_{AI} \cdot x^i y^j, \tag{3}$$

$$X_e = m_{1,0}/m_{0,0}, Y_e = m_{0,1}/m_{0,0} \tag{4}$$

3.2 Parallelism

The parallelism of the eye tracking system can be divided into two main categories: 'parallel in block' and 'parallel in channel' as shown in Fig. 6(a)–(b).

If the operation is identical with each pixel, such as Gaussian filtering, RGB to gray convertion and morphological operation in the pre-processing block, iris color model back projection and binary operation in the iris color model updating block, masking and simplified projection in the eye localization block, then we can divide one image frame into several blocks and use 'parallel in block' technique as shown in Fig. 6(a). The number of block can be defined by the number of CPU or GPU cores in one's computer, for example, 4 CPU cores in Fig. 6(a). Each block can be processed through a sequence of image processing pipeline individually until the next



(a) (b) (c)
(d) (e) (f)
(g) (h) (i)
(j) (k) (l)

Figure 5: (a) Input image and the horizontal projection. (b) Input image and the vertical projection. (c) Extracted iris rectangle. (d) High x-directed gradient but low y-directed gradient pixels. (e) Fig. 5(d) plus extracted biggest circle. (f) Valley-Peak field. (g) Fitted parabolas of eyelids. (h) Detected inliers of both parabolas of eyelids and limbus feature points. (i) All the detected feature points. (j) Accurate iris mask M_{AI}. (k) Inverted iris masked simplifed eye image. (l) Virtual center.

image processing stage needs information from other blocks. The longer length of the sequence may produce better efficiency since the fork-join of threads will take some overhead.

If we are dealing with color image frames, which have multiple channels, then we can further apply 'parallel in channel' technique if the operation to all channels are identical, such as auto-white-balencing in the pre-processing block and RGB to HSV convertion in the iris color model updating block. As shown in Fig. 6(b) for example, three channels can be propagated through image processing pipeline individually until the next image processing stage needs information from other channels. Besides, in one channel, we can further deploy 'parallel in block' if the operation to all pixels are identical, for example, the HSV histogram calculation in the iris color model updating block.

4 LANGUAGE SELECTION

C++ with OpenCV library is the programming language chosen to implement in this work according to its speed, efficiency and multiple parallel-library supporting. OpenMP and TBB with OpenCV will be used to fork and join parallel tasks though CPU cores. CUDA

Algorithm 1: Modified Starburst Algorithm

Input: Preprocessed eye image I_E and iris mask M_I
Output: Feature points surrounding iris
initialization;
while *seed point does not converge* **do**
  clear F, the set of final feature points;
  Stage1:
  Emit rays radially from the seed point with angle ranging in $[0, 2\pi]$;
  for *Each ray* **do**
    Move extendedly along the ray from the seed point;
    Calculate derivation of intensity at each pixel;
    if *Outside the iris mask* $\wedge$ *derivation* > 0 **then**
      Push feature point to F;
  Stage2:
  for *Each candidate feature point detected in Stage 1* **do**
    Estimate the angle of the line from the feature point to the seed point, called Ang_{fs};
    Emit rays from the feature point back to the seed points with angle ranging in $[Ang_{fs} - \pi/12, Ang_{fs} + \pi/12]$;
    for *Each ray* **do**
      Move backward along the ray from the the feature point;
      Calculate derivation of intensity at each pixel;
      if *Outside the iris mask* $\wedge$ *derivation* > 0 **then**
        Push feature point to F;
  Seed point ← Geometry center of feature points;

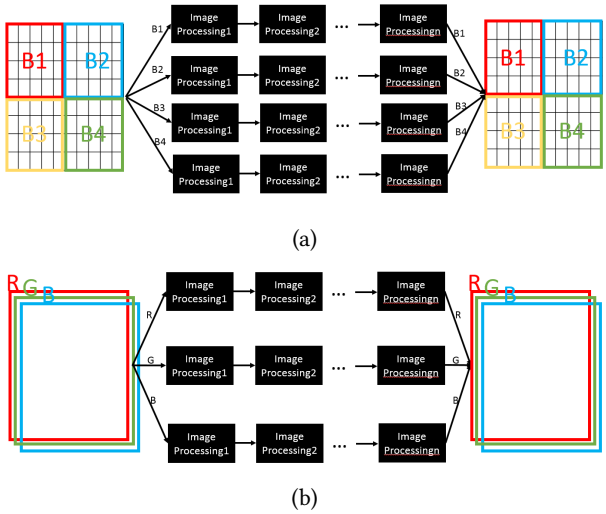


Figure 6: (a) Parallel in block. (b) Parallel in channel

with OpenCV will also be considered to parallelize some tasks on GPU cores. C++ along with OpenCV will also be applied to implement some I/O and GUI implementation.

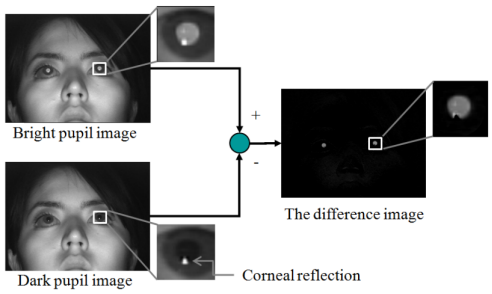


Figure 7: Difference of bright and dark pupil technique[3]

5 RELATED WORK

Ryan et al.[7] switches between limbus detection and pupil detection due to the lightness of environment. They use pupil detection in bright light, and use limbus detection in dim light. Starburst[4] algorithm and ellipse fitting with RANSAC technique are used to find the pupil and cornea in their work. Ebisawa[3] uses dark and bright pupil difference technique to detect pupil. The work is shown in Fig. 7. Bright pupil image is generated by turning on the ring-like Infrared light sources attached around the aperture of a camera. Dark pupil image is generated by turning on other ring-like infrared light sources attached away from the aperture. After getting the difference image, setting a threshold and extract the connected component will give a result of pupil contour[6]. The accuracy of this kind of work is in the range of 5-20 pixels.

6 STATEMENT OF EXPECTED RESULTS

Current system is implemented in Visual C++ and OpenCV library [2] on a personal computer with a 4.0-GHz CPU and 24-GB RAM. The size of the monitor is 22 inches and the aspect ratio is 16:10($47.39 \times 29.61\ cm^2$). The resolution of an eye image is 640×480, and the resolution of the screen is 1680×1050. The distance between the user and the monitor is 50c.m away in front of the monitor, which makes my field of view limited to the size of the monitor. The visual angle ranges horizontally in $[-25.35, 25.35]$ $(degree)$ and vertically in $[-16.49, 16.49]$ $(degree)$. The experiments conducted with the current system shows that the processing speed of the whole system is 10–11fps, and the average accuracy is about 10 pixels, which is comparable to those systems using multiple infrared light sources and multiple cameras. After being parallelized, it is expected to be 1.6 times speedup under 2 CPU cores, which is 16fps, and 3.2 times speedup under 4 CPU cores, which is around 30fps while the accuracy should remain almost same, that is, 10 pixels.

7 A TIMETABLE

The schedule is described as Table 1.

Table 1: Timetable

Item	Schedule	Comment
Experiment environment	4/15-4/22	error format(mean+std).
Profiling	4/23-4/29	cost of time.
Implementation	4/30-5/27	OpenMP+TBB+CUDA
Experiments	5/28-6/10	OpenMP+TBB+CUDA
Presentation slides	6/11-6/24	presentation slides ready

REFERENCES

[1] A.J. Allnutt. 1981. Safety with Lasers and other Optical Sources. *Optica Acta: International Journal of Optics* 28, 10 (1981), 1312–1312. https://doi.org/10.1080/713820456 arXiv:http://dx.doi.org/10.1080/713820456

[2] G. Bradski. 2000. The OpenCV Library. *Dr. Dobb's Journal of Software Tools* (2000).

[3] Y. Ebisawa. 2009. Robust pupil detection by image difference with positional compensation. In *Virtual Environments, Human-Computer Interfaces and Measurements Systems, 2009. VECIMS '09. IEEE International Conference on*. 143–148. https://doi.org/10.1109/VECIMS.2009.5068882

[4] Dongheng Li, David Winfield, and Derrick J. Parkhurst. 2005. Starburst: A Hybrid Algorithm for Video-based Eye Tracking Combining Feature-based and Model-based Approaches. In *Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05) - Workshops - Volume 03 (CVPR '05)*. IEEE Computer Society, Washington, DC, USA, 79–. https://doi.org/10.1109/CVPR.2005.531

[5] Chern Sheng Lin. 1999. Discussion About Harm That Lasers and Other Souces of Lights Do to Eyes @ONLINE. http://140.134.32.129/scteach/scteach88/index.html/.

[6] C.H Morimoto, D Koons, A Amir, and M Flickner. 2000. Pupil detection and tracking using multiple light sources. *Image and Vision Computing* 18, 4 (2000), 331 – 335. https://doi.org/10.1016/S0262-8856(99)00053-0

[7] Wayne J. Ryan, Andrew T. Duchowski, and Stan T. Birchfield. 2008. Limbus/Pupil Switching for Wearable Eye Tracking Under Variable Lighting Conditions. In *Proceedings of the 2008 Symposium on Eye Tracking Research & Applications (ETRA '08)*. ACM, New York, NY, USA, 61–64. https://doi.org/10.1145/1344471.1344487

[8] Fred Seeber. 2007. Light Sources and Laser Safety. *FUNDAMENTALS OF PHOTONICS. Blackwood, New Jersey: sn* (2007).