Accelerated exhaustive eye glints localization method for infrared video oculography

Zihan Ding
University of Science and Technology of China
zhding@mail.ustc.edu.cn

Jiayi Luo
University of Science and Technology of China
kdluojy@mail.ustc.edu.cn

Hongping Deng
University of Science and Technology of China
denghp83@ustc.edu.cn

ABSTRACT

Human eye glints localization could significantly and directly improve accuracy of gaze tracking, especially when several glints are localized precisely at the same time. However, traditional algorithms have not ensured the accuracy of glints localization to be used for gaze tracking. Aiming at the problem in infrared video oculography that those glints reflected on the surface of eye iris could be hard to identify and localize, we propose an algorithm for precisely and exhaustively locating eye glints of high accuracy with accelerated process. Our contributions are two-folded. (1). We propose an exhaustive eye glints localization algorithm, which could guarantee both 94.9% recognition rate and < 1 pixel positional accuracy under various brightness conditions. (2). We propose methods to accelerate our algorithm, including a modified quadratic ellipse difference algorithm and video frame difference examination, which guarantee the real-time video processing requirement for gaze tracking system.

CCS Concepts

• **Computing methodologies → Artificial intelligence → Computer vision → Computer vision problems → Tracking.**

Keywords

glints localization, glints tracking, gaze tracking, pupil detection, ellipse difference method

ACM Reference format:

Z. Ding, J. Luo, H. Deng. 2018. Accelerated exhaustive eye glints localization method for infrared video oculography. In *Proceedings of ACM SAC Conference, Pau, France, April 9-13, 2018 (SAC'18)*, 8 pages. https://doi.org/10.1145/3167132.3167200

SAC 2018, April 9–13, 2018, Pau, France

ACM ISBN 978-1-4503-5191-1/18/04$15.00
https://doi.org/10.1145/3167132.3167200

1. INTRODUCTION

The Gaze tracking is an active and important research area for recent decades. And there are many methods [3] proposed to achieve the function. One of the most important is feature-based gaze tracking method [1,4,8,9,12-15,18,19] using infrared (IR) video oculography(VOG), in which the video frames with IR light reflected back from the eye surface brings information like eye comers, iris reflections, localization and shape features of pupil to track the gaze of eye.

Among those features, the location of glints around the pupil is a key element for computing the accurate eye gaze direction in this infrared oculography method, as shown in [1,2,4,9,10,13-15,18,19,21]. Paper [14] gave a time-consuming double ellipse fitting algorithm, which is not stable because of uncertainty of glints. Method in [20] searches glints around a rough pupil boundary. Paper [18] proposes an improved Otsu threshold method and opening-and-closing operation, but the fixed binarization threshold shows poor effects for glints localization under variational illumination conditions. The accuracy of gaze estimation decreases dramatically when accurate iris reflections and pupil features are not available, especially when glints fall into the margin area of iris or even sclera and become hard to detect during the movement of the eye. For above gaze tracking methods, this serious problem will cause a failed tracking of eye gaze under some circumstances. Aiming at this problem, we proposed an accelerated exhaustive eye glints localization method for infrared oculography in this paper, which is also a need for real-time high-accuracy gaze tracking system.

Our novel algorithm proposed in this paper can locate 4 glints precisely and robustly in every frame of the infrared video. Compared with former glints localization algorithms proposed in paper [7,11,22], we use a multivalued binarization threshold value to make it, instead of Otsu threshold or other single-valued threshold. And with accelerated process methods, our algorithm has higher efficiency and recognition rate in glints localization. In the experiment, our videos come from one camera with 4 infrared LED lights illumination for each eye, similar as the equipment used in [4,18,19]. The gaze tracking system we used is a camera on a home-made head-mounted device that captures a close-up shot of the eye like in [16]. Using the algorithm proposed in this paper, we acquired precise location of pupil and glints around the pupil, which guarantees the effect of gaze tracking system.

Our algorithm has multiple-rounds glints localization process. We localize the typical glint contour firstly and then localize other glints around the pupil complementarily, using a self-adapting

multivalued threshold to search the possible contours of glints under different brightness conditions, which is characteristic in our algorithm. We also use a modified ellipse difference method to localize the position of pupil and detect its contour. Above will be elucidated clearly in section 2; In section 3 we propose some methods to accelerate our algorithm, including the quadratic utilization of modified ellipse difference algorithm [5,20] and examination between consecutive frames; In section 4 we display our experiment results, in which the robust and high accuracy recognition rate shows our algorithm satisfies the requirements of real-time gaze tracking system.

2. ANALYSIS AND ALGORITHM

The eye images used in this paper are frames of standard videos of eye illuminated by 4 infrared LED lights. The videos are converted to frames of size 640×480 pixels. Under normal circumstances, there should be 4 glints in each frame corresponding to 4 LED spot lights. And our final goal of algorithm proposed in this paper is to locate these 4 centers of glints precisely and efficiently. Our localization process could be separated into the following 5 steps:

1). Image preprocessing;

2). Localization of typical glint;

3). Localization of glints around;

4). Localization of glints as supplement with multivalued threshold;

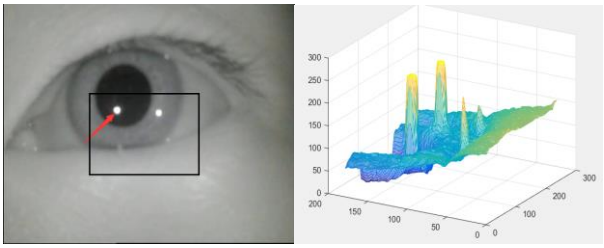
5). Examination of glints location.

But before the formal 5 steps of localization process, we have a glints analysis section to elucidate the typical threshold we use in image preprocessing step.

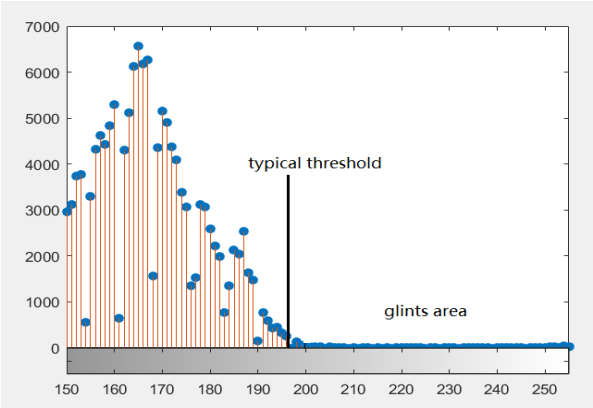
Considering the glints recognition experience by human naked eyes, we have the criteria of determination of glints around pupil in eye images mainly based on the following two characteristics: (1) the glints have relatively high brightness in the image; (2) the edge of glint has a relatively great brightness gradient. Above are the core features we used in localization.

2.1 Glints analysis

We need to determine an image binarization threshold which can effectively extract the glints contours in eye images. Before that we take a sample set from our video library to analyze glints features. And a glint always shows up as a steep nearly Gaussian distribution [17] with average diameter around 10 pixels in our gray scale images set. We take a representative eye image to display our result, and we get its gray-value 3D graph and grayscale statistics graph as in Figure 1. The white area marked by the red arrow in Figure 1(a) is a highlighted glint we could recognize by our naked eye. We can also see from Figure 1(b) that there are several glints with high gray value, which are just the areas we want to extract. Combined with Figure 1(c), we could tell that the two glints with highest gray value in the image have an average gray level above 200, which indicates the range of gray value above 200 is the scope where glints with highest brightness exist, and almost all pixels in this scope belongs to these brightest glintsarea. Therefore, the gray value within this scope can be used as binarization threshold for extracting these glints. Our ultimate aim is to locate all 4 glints around pupil, but we first localize the



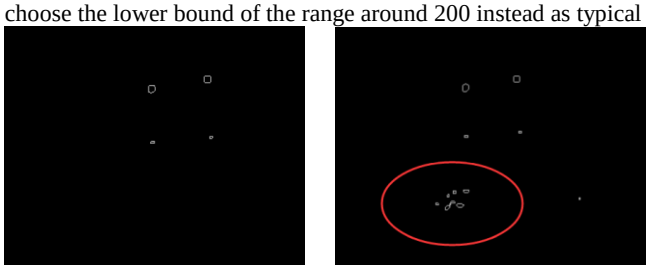
(a) **(b)**



(c)

Figure 1: (a) is the origin eye image frame; (b) is gray-value 3D graph of the frame; (c) is grayscale statistics graph of the frame, the horizontal axis is the gray value of the image, while the vertical axis is the number of pixels in that gray value.

contour with the most obvious characteristics of a glint, called the typical glint, as is also the area marked by the red arrow in Figure 1(a). In order to effectively extract the typical glint, our selection of binarization threshold has specific requirements: on the one hand, the threshold is supposed to be lower than the average gray value of typical glints, allowing the overall contour of the glint to be extracted to localize the center of it; on the other hand, the threshold value is supposed to be higher than that of other regions, so that the contour subjects after binarization process are the glint areas. Considering the two points mentioned above, take the frame we used in Figure 1 as an example, we should not select a relatively higher threshold in the gray value range 200-255, but choose the lower bound of the range around 200 instead as typical

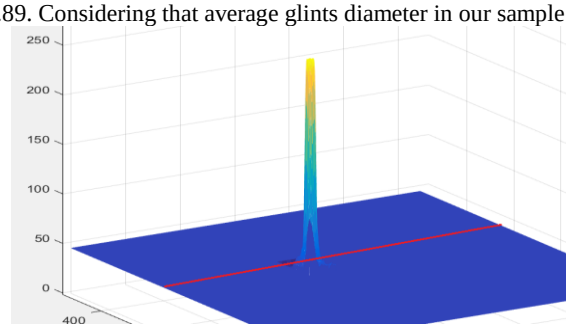


(a) **(b)**

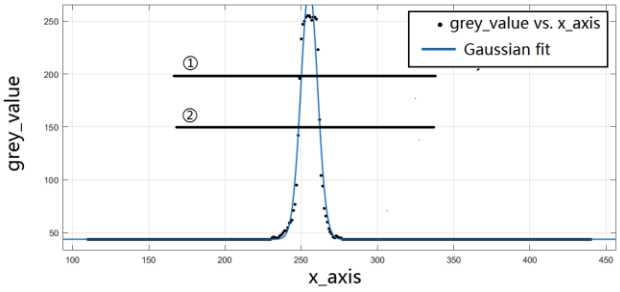
Figure 2: (a) is contour image using 200 as binarization threshold while (b) is from the same origin eye image using 196 as binarization threshold instead.

glint localization binarization threshold. And we found that this lower bound of gray value has characteristics: when the binarization threshold is lower than the lower bound value, the number of pixels will increase dramatically during a unit decrease of threshold value, which is caused by the non-glint area filling into the extracted pixels set, as shown in Figure 2. Figure 2(a) is the image using 200 as binarization threshold and taking a finding-contour process, while Figure 2(b) is the same image using 196 as threshold instead. The red circle area shows the irrelative area extracted by a lower threshold outside the 4 glints area. Therefore 200 could be an ideal threshold for extract the glints area in this image, while 196 would be a poorer one. Though a bit more pixels when we take 196 as a threshold than 200, we can feel ease to take all value in 196-200. But lower threshold than 196 may cause problems. We will take 196 as binarization threshold in this example, as is also the typical threshold in Figure 1(c).

In addition, we analyze glints separately in our sample images set. A representative one shown in Figure 3(a) is for illumination. And its cross section graph is shown in Figure 3(b), in which the light intensity distribution approximates the Gaussian distribution [17]. Given the two threshold line in Figure 3(b), line ① represents the typical threshold we get from the algorithm above; line ② represents a general glints contour threshold we identify by our naked eyes. We would like to see how much difference is there between contour sizes of these two threshold. We take a Gaussian function fitting for the brightness distribution in Figure 3(b), and calculate according to the fitted curve to get that the contour sizes ratio of threshold ① to ② is about 0.79, indicating diameter ratio is about 0.89. Considering that average glints diameter in our sample images is about 10 pixels, the glint center localization error maximum would be about 0.54 pixel only when the highlighted center of glint deviates furthest.



(a)



(b)

Figure 3: (a) is the separate glint area in gray-value 3D graph and (b) is its cross section graph, x_axis is red line in (a).

We analyzed glints under all kinds of lighting situations separately like above with frames from our video library. The largest localization error is no more than 1 pixel. Therefore we could safely say the typical binarization threshold above is accurate. Besides, in order to display a more accurate outline of glints as seen by our naked eyes, we could multiply the contour size of threshold above by a scale factor, which is contour area size ratio of threshold ① to ②.

2.2 Image preprocessing

We preprocess the image to get the typical threshold as analysis in Section 2.1. Algorithm are designed based on following thoughts: The threshold is initialized as 255 and decreases from it, and the number of pixels filled in during each unit decrease of threshold is recorded. When the threshold reaches a certain bound value, the number of pixels filled in during a unit decrease of threshold this round will be several times (we set this parameter to be 10 in experiments) the average of increased number of pixels before, which shows a similar case of the example image above in Figure 2: the number of pixels increases dramatically when the threshold meets the bound value 196, which is also the typical threshold in Figure 1(c). We therefore derive the typical threshold for locating typical glints. Pay attention to two points in this step:

(1) When calculating the average number of pixels filled in during threshold decrease, we take a preheating process to avoid the instability of the average at the initial decreasing period (that is, from 255-245). We do not take the 10 times judgement during that period. In this way, we can find the exact bound value of threshold without making mistakes. (In the image, the gray level of the glints is much higher than that of the other area. Therefore, there will be less likely to have a large non-glints area above gray value 245);

(2) The threshold above for typical glints localization shall be further converted into a ratio value called typical threshold ratio, namely ratio of the number of pixels with higher gray value than this threshold to the overall number of pixels, especially for further utilization in Section 2.5. The ratio value has better adaptability than real threshold value in a variety of images for further use.

Algorithm for determining the typical threshold as follows:

1) **initialization**: num_incr=0, avr_num_incr=0;
2) **for**(threshold=254 ;threshold>0 ; threshold--)
3) avr_num_incr=num_pix/(255-threshold);
4) num_incr=num_pix-last_num_pix;
5) **if**(threshold<245) **then** #255-245 is for preheating of avr_num_incr.
6) **if**(num_incr > ratio* avr_num_incr) **then**
7) typical threshold=threshold;
8) break;
9) **end if**;
10) **end if**;
11) **end for**;
12) finally output the typical threshold used for extracting the typical glint;

13) #num_incr means increase of number of pixels per unit gray value decrease; avr_num_incr means average of num_incr before current threshold; num_pix means number of pixels with gray value greater than the threshold; last_num_pix means num_pix for last threshold.

The above algorithm can be used effectively to capture typical glints area with characteristics like high gray value in centers and large gray value gradient on the edge of the region.

2.3 Localization of typical glint

In preprocessing steps above we got a typical binarization threshold which is effective for extracting the contour area with the most typical characteristics of glints. And this very contour area is called typical glint in our paper.

Definition of a typical glint: The typical glint we defined is an area considered with highest recognition degree by our naked eyes to be a glint around the pupil in the image. There have been two essential features of glints: 1).high center gray value 2).large gray value gradient on edge and an additional feature: considering that the beam of a light point would be reflected as a rounded pattern on the smooth surface near the pupil, we make a circle fitting for those contours, and therefore 3).the closer the contour is to a perfect circle, the more we recognize it as the typical glint. The above three features are the main criteria for typical glint we use in our algorithm. Our algorithm are designed in following procedure.

According to the typical binarization threshold we get above in Section 2.2, we first find contours in the properly dilated (noticing that a proper dilating process could make those contours smooth) binarization image and the minimum circumcircle of each contour, then calculate the following two functions for each contour:

(1) the gray value difference function M:

To calculate the average gray value difference between inside and outside the contour, we could use the circular difference algorithm [6]. However, considering the small size of glints and to save computing resources, we just use the gray value of some specific internal and external points to represent the average gray value inside and outside the contour. As shown in Figure 4, internal gray value is represented by the average of those internal points on the cross line while the average gray value outside comes from the 4 external points. The greater the average gray level difference of inside and outside, the more typical characteristics of glints the contour has.

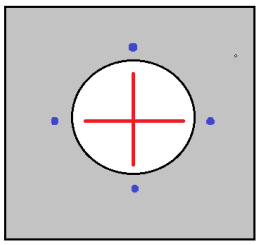


Figure 4: the white circle area represents the contour; the pixels on red cross line are used for represent average gray value inside the contour; the blue points are used for calculating the average gray value outside the contour.

The formula of M is shown in formula (1):

$$M = \frac{\sum_{i}^{n} in[i].data}{n} - \frac{\sum_{j}^{4} out[j].data}{4} \quad (1)$$

In formula (1), variable *in* is the pixel point set on the cross line inside the glint; *in[i].data* is gray value of the *i* th pixel point in the *in* set; variable *out* is the set of 4 points outside the glint;

out[j].data is gray value of the *j* th pixel point in the *out* set; variable *n* is the number of pixels points in the *in* set.

(2) the normalized circle fitting error function N:

$$N = \frac{\sum_{i}^{s} \left[\left(c[i].x - x_0 \right)^2 + \left(c[i].y - y_0 \right)^2 - r^2 \right] / r^2}{s} \quad (2)$$

In formula (2), variable *s* is the size of one contour, namely the number of pixels on the contour; *c[i].x* is the x coordinate value of *i* th pixel on the contour; *c[i].y* is the y coordinate value of *i* th pixel on the contour; *r* is radius of minimized circumcircle of the contour; x_0 is the x coordinate value of the center of circumcircle of the contour; y_0 is the y coordinate value of the center of circumcircle of the contour.

The final criterion function of typical glint is F:

$$F = M - Ratio * N \quad (3)$$

The *ratio* in formula (3) is used as a normalization factor to ensure the values of M function and N function or variation of M and N are of the same magnitude. We can identify the contours with both of the criteria in this way. The typical glint would have both a large difference between inside and outside average gray value, and a little circle fitting error at the same time, and this is also the reason for the 'minus' before N function in formula (3). Therefore the contour with the maximum value of F function in all contours would be considered the typical glint. And we also get an effective binarization threshold from steps above.

Actually, before we search the maximal F function, we first give a lower limitation value of M function to restrict the minimum of gray value gradient on edge of typical glint, which is shown in formula (4):

$$M_{limit} = GV_{top\,0.1\%} - typical\_threshold \quad (4)$$

In formula (4), $GV_{top\,0.1\%}$ is the lower bound of top 0.1% high pixels gray value in all pixels of image, as is also near the center of most highlighted glint; the $typical\_threshold$ is just the typical threshold we get from Section 2.2. We can easily see that the M function of typical glint should be larger than M_{limit}, or else the possibility of this contour to be typical glint would be ruled out directly. We use above principles to make sure we localize the most typical glint in this step. And this is a key step in our proposed algorithm. The accuracy of typical glint localization could ensure accuracy of other glints localization and pupil localization.

Algorithm in Section 2.3 is as follows:

1) **initialization**: $F_{max} = 0$;

2) **traverse** all contours found after binarization with typical threshold in Section 2.2 and **do**:

3) calculate functions M, N and F for the contour;

4) **if** ($M < M_{limit}$) continue;

5) **else if** ($F > F_{max}$)

6) $F_{max} = F$ and store the parameters of the contour;

7) finally the contour with maximal F function value is the typical glint we want.

2.4 Localization of glints around

The typical glint we localized above provides a position range for localizing other glints around. We know that the 4 glints are generated by the reflection of light on human eye surface, iris specifically. Therefore all the glints we want locate in the range of the eyeball and the distances of them are close to the diameter of pupil. So the pupil position and size are referred to when localizing other glints. In order to localize the center of pupil and measure the size of it, we use ellipse difference algorithm [5,19] to make it. According to circle difference method [6] thoughts, the extended ellipse difference algorithm is as follows:

$$E = \left| G_\sigma(a) * \frac{\partial}{\partial a} \oint_{a,\varepsilon,\theta,x_c,y_c} \frac{I(x,y)}{2\pi a\varepsilon + 4a(1-\varepsilon)} ds \right| \quad (5)$$

In formula (5), $G_\sigma(r)$ is Gaussian smoothing function; $I(x,y)$ is intensity of image; a is the major axis of ellipse; ε is ratio of minor axis to major axis ; θ is the inclination angle of ellipse; (x_c, y_c) is the center ordinate of ellipse; $2\pi a\varepsilon + 4a(1-\varepsilon)$ is the perimeter of ellipse. On the ellipse periphery ds whose center is (x_c, y_c), major axis is a, minor axis is $a\varepsilon$ and inclination angle is θ, the operator integrates the pixel gray value, normalize it, and calculate the difference of it. And the maximum of the difference is corresponding to the actual pupil ellipse with parameters $(x_c, y_c, a, \varepsilon, \theta)$, as is shown in Figure 5.

There should be the maximum of ellipse difference when the ellipse contour is the actual pupil contour. We only need the rough size of pupil as reference of distance limit. In the case of a near round pupil, we might as well take the diameter of $(a+b)/2$ as pupil size reference, with $b = a\varepsilon$.

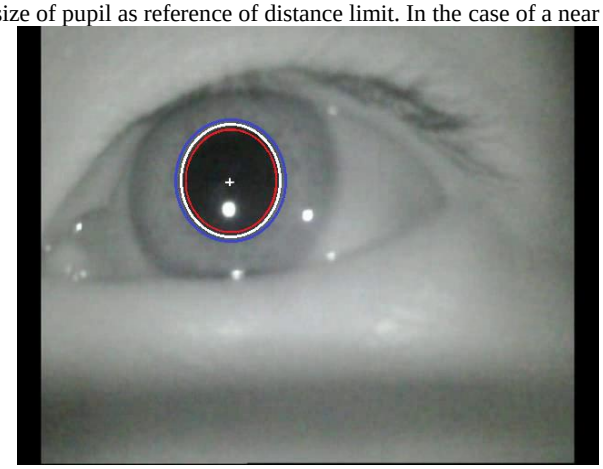


Figure 5: Ellipse difference diagram. The white ellipse contour is the pupil contour we found, with the maximal gray value difference of ellipses, which are the blue one outside and the red one inside.

Besides, we have got the circumcircles of contours with typical threshold in Section 2.2, 2.3. We can set distance range parameters directly according to the pupil size we get above. Meanwhile, according to the two criteria of glints as center gray value and gray value gradient on edge, we also use function M in Section 2.3, which is the difference of average gray value inside and outside the contour, to determine the contours. The lower limitation of M is set to be 1/3 of M_{limit} used in Section 2.3 as we want to make sure find as many glints as possible.

2.5 Localization of glints as supplement with multivalued threshold

Sometimes the number of overall glints we locate through the process above is less than 4, which is an usual case because the monodromy threshold could not handle with various lighting conditions, we take this step to locate glints with multivalued threshold complementarily, trying to locate at least 4 most highlighted glints in all. We also use similar criteria as Section 2.4: (1) the relative position of the contours; (2) the gray value gradient on the edge of contours; (3) central gray level of contours. Details about settings in algorithm are the same for the first two criteria as in Section 2.4, while for the third criteria we use multivalued binarization threshold this time.

Take one frame as an example in Figure 6. We could see that there are 2 glints extracted with 170 as binarization threshold, noticing that the typical threshold we use to locate the typical glint is 196, which is apparently too high to extract other glints. When threshold equals to 160 we could extract 4 glints effectively as an ideal case. However, when 150 we could only extract 3 ef fectively as one relatively weak glint has fused into the contour of

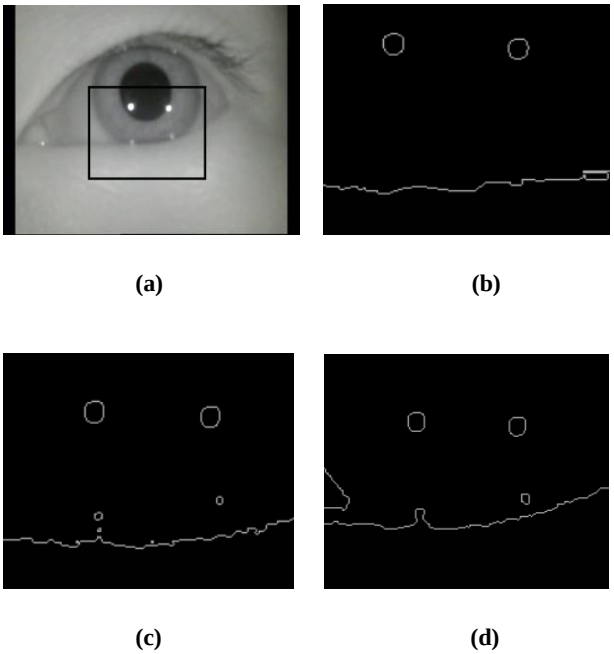


Figure 6: Different binarization thresholds paradigm. (a) is the origin image; (b)(c)(d) are contour images using binarization threshold 170, 160, 150 respectively.

eyelid. Therefore there will be an appropriate range of threshold for us to locate other glints with multivalued threshold complementarily when the number of overall glints have been found is less than 4. We could take a gradient descend for threshold to ensure all the glints are involved in. This is the core principle of Section 2.5.

In our algorithm, we use the threshold ratio value instead of the threshold value itself in this step to have a better adaptation for a variety of video frames. Threshold ratio value of 5% means there are 5% pixels among all images pixels are extracted with this threshold value. Therefore we need to convert between threshold value and threshold ratio value at first. Then we increase the typical threshold ratio from the initial value by 5% per step, until the ratio is large enough to ensure no glints else could be found, which is 40% in our algorithm. Actually this upper limit parameter could be set freely as long as it is big enough. With each value of the multivalued threshold or say threshold ratio we get, we take the following process similar as it is in Section 2.3, 2.4: binarization with threshold value, dilating process, finding contours and their circumcircles in the processed image, and determining if the contours are qualified glints with other two criteria (1)(2) as we mentioned in the opening paragraph of Section 2.5.

When we find a qualified glint contour to fill into our glints set, we still need to examine if the contour is the one we already found before. And we set a distance range for examining the new centers from the old ones. If the new glint center has a shorter distance from any of those old ones than the lower limitation of distance range $R_{\min}$, even if there is a position fluctuation when using different threshold, we would rather have reason to believe the two glints contour are actual the same one because of the extremely short distance.

Algorithm in Section 2.5 are as follows:

```
initialization： threshold_ratio = typical threshold ratio;
while(threshold_ratio < 40%)
    threshold_ratio+=5%;
    convert threshold_ratio to threshold_value;
    take processes: binarization with threshold_value, dilating,
finding contours and their circumcircles in the processed image；
    for i th contour in contours set：     #traverse all contours
found above.
        if ( dis(center of contour[i], center of typical contour)
< R_max && dis(center of contour[i], center of pupil) < R_max )
then
            for j th glint in glints set:     #traverse all glints
found in Section 2.4.
                if ( dis(center of contour[i] , center of  glint[j])
< R_min ) then
                    put contour[i] into glints set;
                end if;
            end for;
        else continue;
    end for;
end while;
finally we get the glints set with new glints found;
#dis(a, b) means distance of points a and b;
```

$R_{\max}$, $R_{\min}$ are parameters we set for examine the contours we find. $R_{\max}$ is set with reference to the diameter of pupil and acts as a range limitation of the distance between new contours and typical glint, and distance between new contours and pupil center; $R_{\min}$ is the lower bound of distance to judge if the new contour is actually one of the old.

2.6 Examination of glints location

Before the final output we need to examine the location relationship among all glints we found, especially when the total number of glints is larger than 4. We need to sum the total distance of one glint from the other glints and rule out the one with largest distance incessantly until there are only the 4 glints caused directly by the infrared spot lights left.

However, when the reflected glints move onto the sclera, in which case it will cause huge glints or several decentralized glints, we output an error in our algorithm.

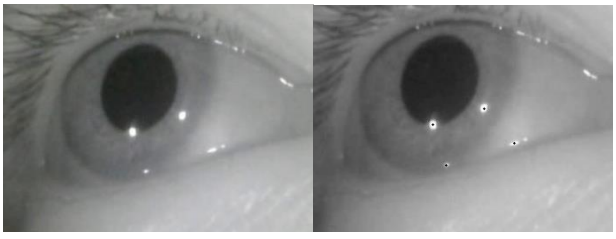
As is shown in Figure 7，when detecting 5 glints (one glint falls into the sclera area and disperse into two little glint), we exclude the further one and maintain the main part of this glint. More results see in Section 4.1 experiment results.

3. ACCELERATION PROCESS

3.1 Modified ellipse difference algorithm

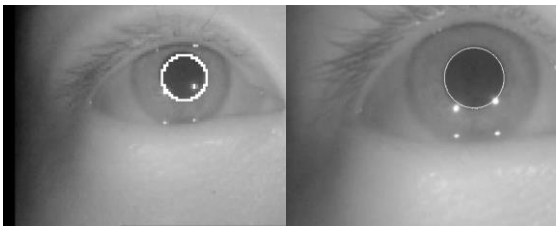
In Section 2.4 we use the ellipse difference algorithm to search the ellipse best matching the pupil contour. We need at least 5 parameters to represent an ellipse: coordinate of ellipse center which includes 2 parameters, the major axis and minor axis of ellipse or ratio of minor to major axis instead, and the inclination angle of ellipse. Therefore, the problem of searching a proper ellipse converts to an issue of searching a scalar (ellipse difference) extreme value in a 5-dimentional space. The search process will take considerable time without any accelerated techniques.

To save computing resources without reducing the accuracy of the method, we propose a modified ellipse difference algorithm instead, by using the ellipse difference algorithm twice in both a large area and a small area separately. We also adopt multithread processing in our algorithm. We first compress the image into 160×120 pixels, then we use ellipse difference algorithm in a relatively large range near the typical glint we localized. We scale up the parameters of contour achieved from this large area ellipse difference process back to the origin image. With the coarse location of pupil center, we use the ellipse difference algorithm again in a small range around the location we have got above. Finally,



(a) **(b)**

Figure 7. Glints localization with final examination.



(a) **(b)**

Figure 8: Results of modified ellipse difference algorithm. (a) is pupil localization result of large area ellipse difference; (b) is result of small area ellipse difference.

we locate the pupil and receive its diameter parameters with both high accuracy and lower time consuming. The results of large area and small area ellipse difference methods are shown in Figure 8. With this modified pupil ellipse difference algorithm, we accelerate our image processing from hundreds of milliseconds to 20 milliseconds per frame.

Another modification of ellipse difference algorithm is that when we use it in Section 2.4, we already have a glints set in Section 2.2, 2.3, and therefore the ellipse difference process skips areas of glint contours already found to avoid the influence of glints when localizing the pupil. For this reason our algorithm could give accurate pupil localization results shown in Figure 8 neglecting serious interference of glints on edge of pupil. Section 2.4 is a perfect position for using the ellipse difference algorithm to benefit both localization of pupil and further localization of glints.

3.2 Consecutive frames difference examination

To speed up our image processing, we set some default parameters for video processing and change them with difference examination of consecutive frames. We calculate once every 5 frames the average of pixel gray value difference and difference of average gray value over the whole images and compare these two values with those of the frame corresponding to the present default parameters. The former value reflects mainly pixels variation between the compared frames, while the latter value reflects mainly the overall gray level change. Both of the values should not change greatly when the parameters remain default values, which are the values since last time parameters change. Those default parameters we set includes the typical threshold, typical threshold ratio, the lower limitation of M function in Section 2.3 and diameter of pupil, etc. Therefore it is not necessary to take ellipse difference processing for each frame in one video. The consecutive frames difference examination also save time for calculating those adaptive parameters, which is another primary time consumption. The difference examination and default parameters makes the average time consumed by ellipse difference part reduce to less than 10 ms. And it ensures the parameters adaptation for videos when the important elements in frames change considerably.

4. EXPERIMENT

4.1 Experiment setting

The video frames used in this experiment come from infrared microspur miniature camera with focal length 2.79 mm, viewing angle 60°, depth of field for 30-60 mm, 720×576 effective pixels, using 640×480 pixels images when processing the video frames, fame rate 50Fps. Around each camera is equipped with 4 infrared LED lights of wavelength 850 nm as light source. The divergence angle of LED lights is about 120°. In front of the cameras are equipped with 800 nm wavelength near infrared optical filter, which can keep 800-1100 nm light and filter light with wavelength lower than 800 nm.

The glints localization algorithm is simulated on a computer of i7 2.60GHz and 8GB memory, platform of Microsoft Visual Studio with Opencv2 library. The program used in the simulation implements the complete algorithms and functions mentioned in this paper as follows including modified ellipse difference algorithm: 1. Image preprocessing; 2. Localization of typical glint; 3. Localization of glints around; 4. Localization of glints as supplement with multivalued threshold; 5. Examination of glints location.

4.2 Experiment results

We sampled 10 videos randomly and selected 500 frames from each video, including continuous frames and dispersive frames. We firstly compare our original proposed method (without any acceleration process) with two acceleration methods: (1)modified ellipse difference algorithm; (2)consecutive frames difference examination on the above sampled dataset. Mean accuracy and Time per frame are shown in Table 1. Secondly, methods in paper [14,18,20] are compared with our final accelerated method on the same dataset. Mean accuracy, Time per frame and Error in pixels are shown in Table 2. We have totally 256 frames failed to localize all glints correctly in 5000 frames test set, while for methods in paper [18,14,20] the numbers of failed frames are 591, 1190, 795 separately. Our method shows better effects in accuracy and time performance. The accuracy of localization of glints center is less than 1 pixel. With effective acceleration process, the average time for processing one frame is 18.7ms, which could satisfy the requirement of continuous video data processing in real-time system for gaze tracking.

Table 1: Our Glint Detection method with acceleration process

	Time(ms)	Accurracy
Original method (without acceleration)	>570	95.2%
Original method + modified ellipse difference algorithm	28.3	95.2%
Modified ellipse difference method + frames examination	18.7	94.9%

Table 2: Comparison of different Glint Detection methods

Method	Accuracy	Time(ms)	Error(in pixels)
Our method	94.9%	18.7	0.73
Paper [18]	88.2%	20.3	0.71
Paper [14]	79.2%	35.7	2.20
Paper [20]	84.1%	31.3	1.93

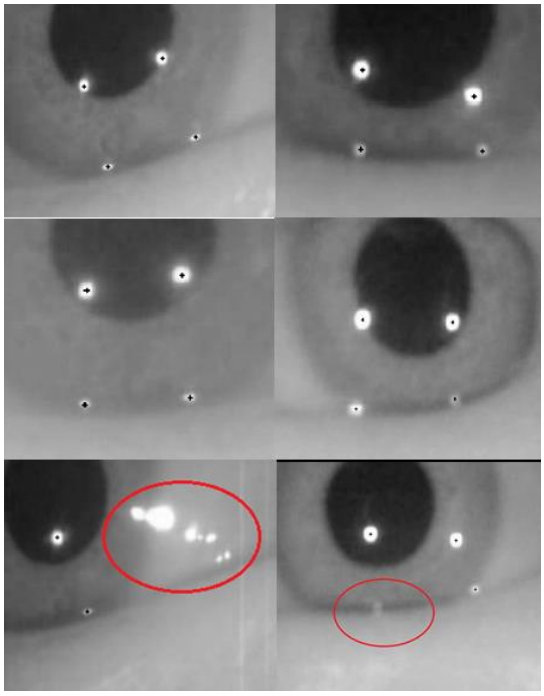


Figure 9: Final results of glints localization algorithm.

Some of the localization results are shown in Figure 9. The last two images in Figure 9 shows the case of huge glints because of its location on sclera and the case of extremely weak glint with eyelids shelter, both of which outputs an error in experiments.

5. CONCLUSION

According to the experiment results above, the robust localization performance and high accuracy recognition rate shows our algorithm could satisfy the requirements of gaze tracking system. Based on the essential features of glints around the pupil, we use a self-adapting multivalued threshold to search every possible contours of glints under various brightness conditions, which is characteristic in our algorithm. We also use a modified ellipse difference method to locate the position of pupil and detect its contour, which could help search other glints besides the typical one. With the acceleration methods, our algorithm satisfies not only accuracy need of gaze tracking function but also real-time processing ability. Generally, it is an algorithm with a robust, precise, high-efficiency eye glints localization performance for infrared oculography in gaze tracking system.

6. REFERENCES

[1] Beymer, David, and Myron Flickner. "Eye gaze tracking using an active stereo head." *Computer vision and pattern recognition, 2003. Proceedings. 2003 IEEE computer society conference on*. Vol. 2. IEEE, 2003.

[2] Blignaut, Pieter. "Mapping the pupil-glint vector to gaze coordinates in a simple video-based eye tracker." *Journal of Eye Movement Research* 7.1 (2013).

[3] Chennamma, H. R., and Xiaohui Yuan. "A survey on eye-gaze tracking techniques." *arXiv preprint arXiv:1312.6410*

[4] Cheng, Hong, et al. "Gazing point dependent eye gaze estimation." *Pattern Recognition* 71 (2017): 36-44.

[5] Daugman, John. "New methods in iris recognition." *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)* 37.5 (2007): 1167-1175.

[6] Daugman, John. "How iris recognition works." *IEEE Transactions on circuits and systems for video technology* 14.1 (2004): 21-30.

[7] Goni, Sonia, et al. "Robust algorithm for pupil-glint vector detection in a video-oculography eyetracking system." *Pattern Recognition, 2004. ICPR 2004. Proceedings of the 17th International Conference on*. Vol. 4. IEEE, 2004.

[8] Gwon, Su Yeong, et al. "Robust eye and pupil detection method for gaze tracking." *International Journal of Advanced Robotic Systems* 10.2 (2013): 98.

[9] Hennessey, Craig, Borna Noureddin, and Peter Lawrence. "A single camera eye-gaze tracking system with free head motion." *Proceedings of the 2006 symposium on Eye tracking research & applications*. ACM, 2006.

[10] Lai, Chih-Chuan, Sheng-Wen Shih, and Yi-Ping Hung. "Hybrid method for 3-D gaze tracking using glint and contour features." *IEEE Transactions on Circuits and Systems for Video Technology* 25.1 (2015): 24-37.

[11] Hiley, Jonathon B., Andrew H. Redekopp, and Reza Fazel-Rezai. "A low cost human computer interface based on eye tracking." *Engineering in Medicine and Biology Society, 2006. EMBS'06. 28th Annual International Conference of the IEEE*. IEEE, 2006.

[12] Morimoto, Carlos H., et al. "Frame-rate pupil detector and gaze tracker." *Proceedings of the IEEE ICCV*. Vol. 99. 1999.

[13] Ohno, Takehiko, and Naoki Mukawa. "A free-head, simple calibration, gaze tracking system that enables gaze-based interaction." *Proceedings of the 2004 symposium on Eye tracking research & applications*. ACM, 2004.

[14] Ohno, Takehiko, Naoki Mukawa, and Atsushi Yoshikawa. "FreeGaze: a gaze tracking system for everyday gaze interaction." *Proceedings of the 2002 symposium on Eye tracking research & applications*. ACM, 2002.

[15] Peréz, Antonio, et al. "A precise eye-gaze detection and tracking system." (2003).

[16] SensoMotoric Instruments. EyeLink Gaze Tracking. www.smi.de

[17] Shortis, Mark R., Timothy A. Clarke, and Tim Short. "A comparison of some techniques for the subpixel location of discrete target images." *Proc. SPIE*. Vol. 2350. 1994.

[18] Wang, Jianzhong, Guangyue Zhang, and Jiadong Shi. "Pupil and glint detection using wearable camera sensor and near-infrared led array." *Sensors* 15.12 (2015): 30126-30141.

[19] Yoo, Dong Hyun, and Myung Jin Chung. "A novel non-intrusive eye gaze estimation using cross-ratio under large head motion." *Computer Vision and Image Understanding* 98.1 (2005): 25-51.

[20] Yun, Zhang, et al. "EyeSecret: an inexpensive but high performance auto-calibration eye tracker." *Proceedings of the 2008 symposium on Eye tracking research & applications*. ACM, 2008.

[21] Yoo, Dong Hyun, et al. "Non-intrusive eye gaze estimation using a projective invariant under head movement." *Robotics and Automation, 2006. ICRA 2006. Proceedings 2006 IEEE International Conference on*. IEEE, 2006.

[22] Zhang, Jing, et al. "A low complexity method for real-time gaze tracking." *Multimedia Signal Processing, 2008 IEEE*