A Fast and Accurate Eye Tracker Using Stroboscopic Differential Lighting

Frank H. Borsato
Federal University of Technology - Paraná
frankhelbert@utfpr.edu.br

Fernando O. Aluani
University of São Paulo
aluani@ime.usp.br

Carlos H. Morimoto
University of São Paulo
hitoshi@ime.usp.br

Abstract

The use of video-based eye trackers (VETs) for gaze interaction have allowed people with severe motor disabilities to communicate more effectively. Nonetheless, current low cost VETs present limited accuracy and precision that impose several design constraints for gaze-based computer applications. In this paper we present an extension of the differential lighting (DL) technique for pupil detection and tracking using active light sources. The original technique was developed for analog interlaced cameras with external sync signal to synchronize the light sources with the video signal. In this paper we introduce the Stroboscopic DL technique that can be used with any rolling shutter camera, even those with no external sync. We have developed a computer vision technique to adjust the firing of the stroboscopic lights. Our new algorithm also exploits characteristics of pupil images to improve the accuracy of the tracking algorithm. Another advantage of the method is that using flashed pulses of light creates a virtual exposure time, reducing motion blur and temporal shear in the video volume. A real-time 187 fps prototype of the system was implemented using a low cost PS3 camera. Experimental results comparing the performance of our algorithm with Starburst show significant accuracy and speed improvement.

1. Introduction

Gaze-based interaction is an effective, rapid and natural communication means for people with severe motor disabilities such as patients with ALS (Amyotrophic Lateral Sclerosis) and LIS (Locked-In Syndrome). Video based eye trackers estimate the gaze position which can be further used in the development of assistive interfaces. Such interfaces include applications to letter typing, Internet usage, and environmental control, increasing the users independence and quality of life [17].

Despite recent advancements in computer and video technology that allowed for significant improvements in performance and miniaturization, commercial eye trackers remain too expensive for general use. Though low cost solutions have been suggested [7, 13, 16, 14] (including commercial systems under US$ 1K) they are typically limited to 30 or 60 frames per second (fps). Higher frame rates might be used to create more accurate and precise eye trackers [11], and are necessary to estimate fast eye movement trajectories (saccades) during eye gestures used for communication and interaction. Faster eye trackers might also create more responsive gaze interfaces by reducing latency.

Processing high frame rate video requires efficient computer vision algorithms to detect and track the eye features used for gaze estimation. Many algorithms use simplifying assumptions about the brightness of the pupil, for example, that pupil candidates correspond to the darkest regions [13, 32] or are circular and uniform regions [23]. These assumptions might work well in controlled environments but are prone to fail in other general conditions. Appearance based tracking techniques have been used to improve the robustness of eye tracking in uncontrolled environments without the use of infrared lights [27, 31].

The differential lighting (DL) scheme [5, 19] is a robust and computational efficient method that exploits physiological eye properties to detect the pupil. Most eye trackers use a near infrared (IR) illuminator to enhance the quality of the eye image. When the illuminator is placed near the camera optical axis, the camera is able to see the IR light reflected from the back of the eye and the pupil appears bright, as seen in Figure 1a. When the illuminator is placed sufficiently far, the pupil image is dark as seen in Figure 1b.

DL synchronizes two light sources, one on-axis and a second off-axis, with the camera frames to generate alternate bright and dark images. From the difference of two consecutive frames (Figure 1c), high contrast regions are detected as pupil candidates as seen in Figure 1d. The detected region in the difference image corresponds to the overlap between the dark and bright pupils, and require further refinement to track the real pupil contour, as suggested by Hennessey et al. [10].

Another issue that affects the gaze estimation accuracy using rolling shutter cameras (that scan the scene instead of taking a snapshot of the whole scene) is the distortion caused by very fast movements such as saccades. Thought

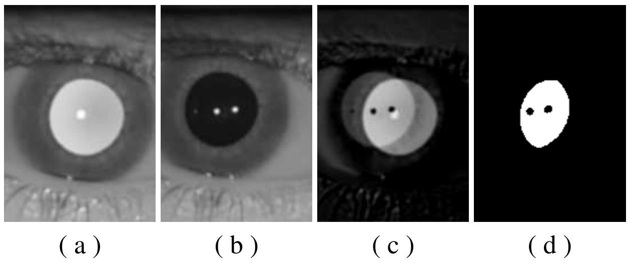
Figure 1. Pupil detection using the differential lighting technique. a) Bright pupil; b) Dark pupil; c) Difference image; d) Thresholded difference image showing the pupil overlap region.

this distortion is reduced when the frame rate increases (compare Figure 2-1a with 1b, and 2a with 2b), we propose the use of stroboscopic lighting to obtain higher quality images of the eye (compare Figure 2-3a, taken at 30Hz and stroboscopic illumination with 1b, taken at 187Hz and continuous illumination). We will call this new method stroboscopic differential lighting (SDL) technique for efficient pupil detection.

The main contributions of this paper can be summarized as the following:

- introduction of the stroboscopic differential lighting technique for accurate pupil detection, that is appropriate for any digital rolling shutter camera without external sync signal;
- development of an accurate, low cost, and high frame rate (187 fps) complete eye tracker prototype using a Sony PS3 camera;
- performance evaluation of the technique and comparison with the Starburst algorithm.

The rest of the paper is organized as follows. The next section introduces the stroboscopic structured lighting technique and how it is used to build an eye tracker. Section 3 presents experimental results comparing the performance of SDL with other state of the art pupil detection technique based on the Starburst algorithm, and Section 4 concludes the paper.

2. Eye Tracking System

SDL could be more easily implemented using cameras that provide an external video sync signal which is, unfortunately, not common in low cost cameras. An important contribution of this work is the introduction of a computer vision based synchronization technique that estimates the beginning of each frame by detecting a banding in the image and adjusting the firing of the on and off-axis light sources accordingly. The conditions in which the banding is produced are throughout discussed, including the camera parameters and their relation to the temporal characteristics of the stroboscopic light firing. The method can, therefore, be used with any rolling shutter video camera.

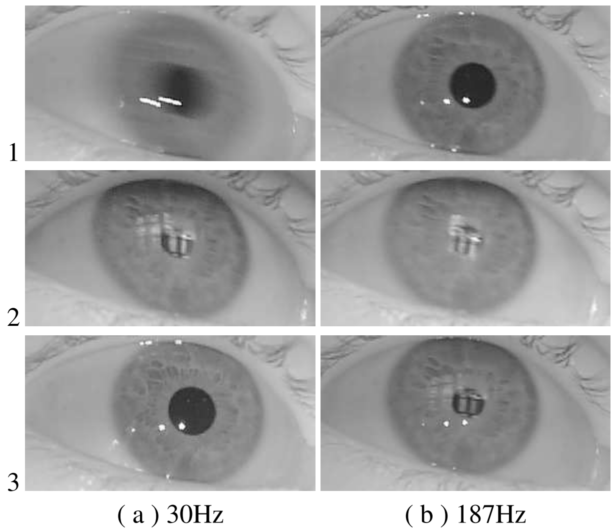
Figure 2. Eye captures during large amplitude saccades. The gain and exposure are the same, except for image 2a, where the exposure was reduced. 1ab) dark room (measured 20 lux at eye) and continuous IR light; 2ab) next to window in a sunny day (measured 1900 lux at eye) and no artificial IR; 3a) dark room and stroboscopic IR illumination; 3b) next to window and stroboscopic IR illumination.

2.1. Stroboscopic Structured Lighting

One problem that reduces eye tracking accuracy is the lack of consistent light [12]. In dim lit environments, cameras typically compensate poor illumination by increasing the exposure and gain. The former increases motion blur of fast moving objects while the latter increases noise (Figure 2-1a shows blurring). In bright places, the exposure can be reduced as well as the gain. This results in better images, with low noise, and less motion blur. Another problem caused by the use of rolling shutters, which is employed by the majority of CMOS video cameras today, is the artifacts such as skew and smear, that affects the image quality around fast moving objects such as the eye (Figure 2-2a shows skew on the pupil and iris).

The use of structured lighting to locate the eyes improves the illumination but poses further challenges to image processing using rolling shutter cameras since the exposure of one frame might include the illumination from different lights. This problem is only aggravated with the use of low cost high speed cameras, so that switching lights on and off with a frequency close to the frame rate of the camera becomes impractical. To overcome the limitations of rolling shutter cameras, Theobalt et al. [30] have used stroboscopic lighting to capture high speed motion of a baseball using standard still cameras and high power stroboscope, and Bradley et al. [3] have used stroboscopic light to synchronize an array of consumer grade cameras. In [14], Kim et al. used infrared leds tied to a simple independent timer to synchronize eye and scene image streams manually. Next we introduce the stroboscopic structured lighting technique for robust and accurate pupil detection.

2.1.1 Camera Model

We adopt a rolling shutter camera model similar to the one of Bradley et al. [3]. Figure 3 illustrates the capturing process within this model. Observe that if a vertical line is drawn perpendicular to the time axis, it may cross two integration areas from different frames (it will certainly happen if the exposure is set to maximum). The slope of the clear and read lines represent the shear of the exposure intervals along the time axis. The slope is a function of the frame period Δt and number of scanlines S. Just before actual pixel data is read out, the sensor works on optical black and/or dummy pixels for D_0 line periods, in a so called vertical blanking time. After the blanking time, N valid scanlines are output, followed by D_Z dummy scanlines at the end of the frame. This is a general model compatible with most camera chips currently available from several vendors [24, 25] and image sensor receivers [29]. Note that D_Z can be zero without loss of generality.

To obtain even exposed frames using pulsed light, the lights must be well synchronized. Let t_{line} be the time taken to read a scanline, which is given by $\Delta t/S$. Let Δ_{strobo} be the stroboscopic light duration. The value of Δ_{strobo} must be at most $\Delta e - (N \cdot t_{line})$ with Δe in the open interval $]N \cdot t_{line}, \Delta t[$.

Consider the start of a frame at t_0 (the start of the first scanline read which is when the VSYNC signal is set) and t_{strobo} as the onset of the illumination. Given Δ_{strobo}, t_{strobo} is bound to

$$t_0 \leq t_{strobo} \leq t_0 + (D_0 \cdot t_{line}) - \Delta_{strobo} \qquad (1)$$

for $\Delta_{strobo} \leq (D_0 \cdot t_{line})$. Otherwise, t_{strobo} must be triggered before the camera VSYNC.

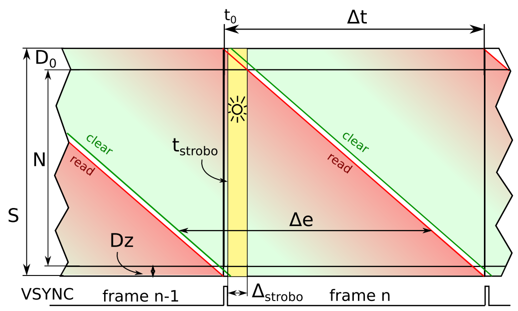

Figure 3. Producing an even frame illumination with stroboscopic light in a rolling shutter camera.

In practice, one should choose the lower Δ_{strobo} as possible, but it is generally limited by the underlying hardware, as the light power output must increase accordingly as the time interval is reduced. For instance, a value of just $20\mu s$ was used in the setup from [30].

In our experimental setting, $S = 278$ and $N = 240$ lines, with $D_0 = 26$ and $D_Z = 12$ lines [24]. For this particular setup, in order to obtain frames with uniform illumination, Δ_{strobo} cannot be greater than $748\mu s$ for a 125fps setting and $t_{strobo} = 0\mu s$ (1). For a 187fps output, Δ_{strobo} is limited to only $500\mu s$.

2.2. Strobo triggering in the absence of camera sync

Regular cameras do not have a synchronization output. Without that, it is hard to get structured light to work properly, as the majority of the frames will suffer from double exposure. To keep the structured light synchronized with a regular camera, we have developed computer vision algorithms that adjust the period of an external clock clk_e, running independently of the camera, that minimizes double exposure in the center of the image.

Figure 4 shows a specific timing diagram where the stroboscopic light causes double exposure, illustrated by the eye image in the same figure. The image is composed by scanlines from two different exposures of the strobe light, fired approximately Δt apart from each other. Region **A** shows the dark pupil effect while region **D** shows the bright one. The dark banding in the middle of the image is the result of no stroboscopic light reaching the sensor during those scanlines.

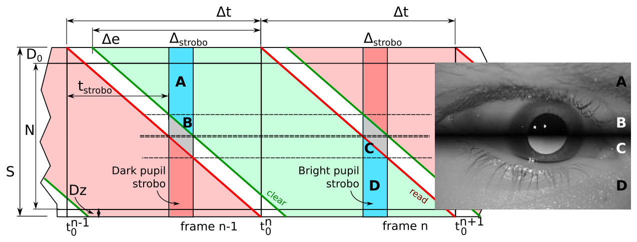

Figure 4. Rolling shutter camera with a delayed strobo and actual frame capture of an eye with double exposure.

Let D_{band} be the position of the first line to receive no light from the LEDs, and assuming we know t_0, it is possible to estimate the value of t_{strobo} as follows:

$$t_{strobo} = t_0 + (t_{line} \cdot (D_0 + D_{band})) - \Delta_{strobo} \qquad (2)$$

If t_0 is unknown, which is the case when there is no sync output from the camera, (2) can be used to compute the relative drift between t_{strobo} and t_0.

Assume now that the triggering of the stroboscopic lights is controlled by an external clock clk_e. Considering t_{strobo} to be constant, (2) can be rewritten to give the drift between the camera clock and the external one as follows:

$$\Delta_{drift} = (t_{line} \cdot (D_0 + D_{band})) - \Delta_{strobo} \qquad (3)$$

Instead of searching for a single dark horizontal line, which intensity depends on the scene, ambient light, and camera gain, we try to determine the position of regions

B and **C** (refer to Figure 4). This is done by finding the strongest responses of a gradient operator on the average column image obtained from the pixel sum of each frame line. The averaged column image can be expressed as follows:

$$\overline{I_c}[l] = \sum_{k=0}^{C} I[l][k]/C \qquad (4)$$

where C is the number of image columns, and $\overline{I_c}[l]$ is the average line intensity of line l.

Regions **B** and **C** are defined as the points of maximum gradient response.

$$I_{band} = I_c * K \qquad (5)$$

where I_{band} is the column image I_c convoluted by kernel K, a 1-d kernel of size 15.

Defining the strongest positive and negative response location as D_{band}^{max} and D_{band}^{min}, D_{band} can be approximated by:

$$D_{band} = (D_{band}^{max} + D_{band}^{min}) \cdot 0.5 \qquad (6)$$

With D_{band} computed from (6), we can use (3) to estimate the drift between the true camera VSYNC (t_0 in the camera model) to the external clock clk_e which is used to trigger the LED strobos.

2.2.1 Strobo triggering adjustment

To produce images without double exposure, the external clock must be adjusted to match the camera clock ($1/\Delta t \approx clk_e$). The external clock must also tick approximately at the beginning of each frame. These two problems are dealt separately. First, the camera period is estimated by measuring the delivery rate to the computer. Then, our software makes fine adjustments to the external clock until the best match is reached. Finally, the external clock is shifted in time in order to tick at the beginning of a frame. Algorithm 1 shows the steps aforementioned. Those steps are done every time the system is started to adapt the clock to the camera being used.

The fine adjustments to the clock are made by observing the change in position of the banding (D_{band}). If the banding moves upwards, in the direction of the image origin, the external clock is lower than of the camera. If the banding moves in the opposite direction, downwards, the external clock is higher in frequency than of the camera. The period of the external clock is adjusted iteratively (by comparing Δ_{drift} obtained by (3) in two observations at different times) until D_{band} becomes stable.

Both (2) and (3) assume that the banding will be in the visible scanlines range. We assert this is true while doing the first two steps by temporarily changing clk_e in order to bring it back. However, the goal is to keep the black banding hidden on the invisible scanlines ($[0, D_0] \cup [D_0 + N, S]$). An

```
Input: Images with a dark horizontal banding
Output: Adjusted external clock ($clk_e$)
/* Stage 1: Gross estimation */
1 Capture frames from camera during an interval $T$;
2 Estimate frame period $\Delta' t$ by dividing $T$ by the number of frames;
3 Adjust external clock $clk_e$ according to $\Delta' t$;
/* Stage 2: Fine estimation */
4 Set not stable;
5 while not stable :
6     Capture a new frame;
7     Create a column image $I_c$ using Eq. 4;
8     Calculate the intensity gradient $I_{band}$ of $I_c$;
9     Find the minimum and maximum points;
10    if maximum location is greater than minimum :
11        Calculate the direction and speed of the horizontal banding
          movement using Eq. 3;
12        if direction is zero :
13            Set stable and go to Stage 3;
14        elif direction is negative :
15            Increment period;
16        elif direction is positive :
17            Decrement period;
18        Adjust external clock using the new period;
19        Wait changes to take effect;
/* Stage 3: Move dark banding to invisible
   scanlines */
20 Push actual period;
21 Adjust external clock to a biased value;
22 Wait until the horizontal banding goes to the frame border
   ($D_{band}^{min} > D_{band}^{max}$) and pop period;
```

Algorithm 1: Overview of the external clock calibration algorithm

important observation led to a simple solution. While in the visible range, the intensity shades on top and bottom of the banding make D_{band}^{min} to be located at a lower position than D_{band}^{max}. However, when the banding hides on the invisible scanlines, the residual shades make D_{band}^{max} to be located at a lower position than D_{band}^{min}.

Once the external clock (clk_e) is set and ticks approximately at t_0, the images from the camera can be used in the usual way. Due to the limited clk_e resolution (half of a microsecond) and different temperature coefficients, some slippery of the banding is expected. To avoid any frame loss, the system must check, from time to time, the position of the banding and if necessary, temporarily increase or decrease clk_e to make it go back to the invisible range (see Figure 11).

2.3. Image Processing

Figure 5 shows a block diagram for the gaze estimation process using SDL.

The initial pupil candidates are computed by detecting high contrast regions between two consecutive frames, similar to the differencial method described by Morimoto et al. [19].

The eye camera provides a stream of bright and dark pupil images. Each image is pre-processed by a Gaussian filter to reduce noise artifacts. The two most recent frames

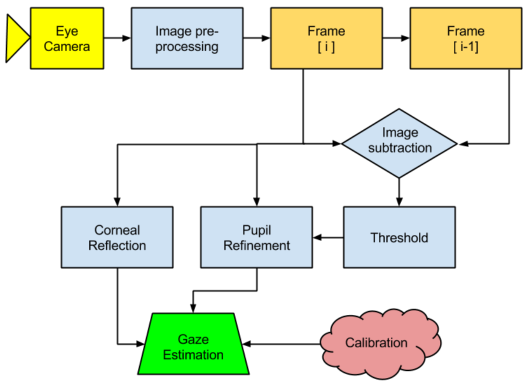

Figure 5. Block diagram of the gaze estimation process.

are kept to estimate the pupil overlap regions from the thresholded difference between Frame$[i]$ and Frame$[i-1]$. As we know if Frame$[i-1]$ is a bright or dark pupil image, a signed threshold is used to reduce outliers. Figure 1 shows the bright and dark pupil images, the difference image, and the resulting pupil overlap region after a user defined threshold.

Pupil candidates are filtered from the pupil overlap regions using size and shape constraints, and the best remaining candidate is used for pupil contour refinement, which computes the actual pupil contour.

2.3.1 Pupil contour refinement

As pointed out by Hennessey et al. [10], the high contrast region computed from the difference between consecutive frames correspond to the ovelap region of the dark and bright pupil images, and the true pupil contour must be computed to achieve higher accuracy. To refine the pupil contour, we use an extension of the Starburst algorithm [15].

Using the center of the overlap region as candidate for the pupil center, our algorithm traces rays outwards to detect the pupil edges. Each ray is only traced in a small region close to the estimated pupil border, based on the overlap region, and then convolved with a gradient filter. The number of traced rays is fixed. Assuming a dark pupil image, the pupil edge is defined as the maximum positive gradient along the ray (and negative for bright pupil images, i.e., the gradient in bright pupil images point inwards). Starburst traces rays by stepping through pixels, calculating difference and marking them as feature points if it passes a threshold. Then traces back to the center, refining feature points until their geometric center converges.

Because corneal reflections may cause artifacts on the pupil contour, pupil edge candidates belonging to regions close to a corneal reflection are discarded. (Figures 6a and

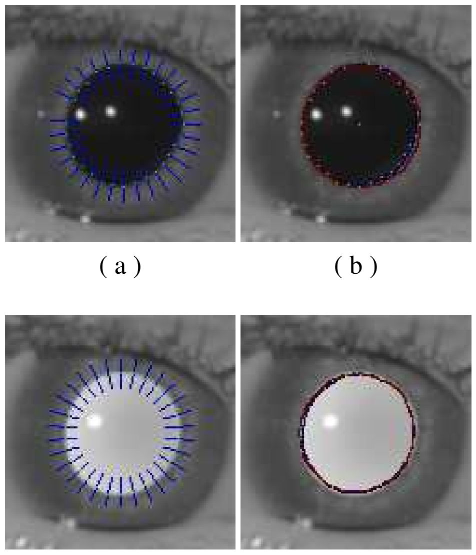

Figure 6. Ray tracing and ellipse fitting results for a dark and a bright frame.

c: Blue points are the ray pixels and green are the candidate points). Starburst treats corneal reflections during ray tracing by previously removing them from the image by interpolation of the surrounding area.

The pupil edge candidates are used to fit the best ellipse in a least-squares manner [9]. Starburst, on the other hand, refine the pupil ellipse using RANSAC [8], as seventeen percent of the feature points were reported as outliers [15]. As we use a gradient kernel with good spacial support, added to the fact that the rays are limited to neighbor regions of the overlap, the number of outliers and their distance to the true pupil border are severely reduced. (Figures 6b and d: green are border points, blue is the refined ellipse while brown is the overlap ellipse).

2.3.2 Corneal reflections

Corneal reflections are detected using a simple threshold method. Bright small blobs, with intensity near to the global maximum, are selected based on their shape and proximity to the pupil center. One such region is selected for bright pupil images, and two for dark pupil images. The center of the corneal reflection is computed as the weighted average of pixels within the blob.

2.4. Implementation

The system is composed of a camera, a structured light source, a controller to trigger the light source and a computer. Instead of connecting the lights directly to the camera, our implementation uses a separate controller which accepts commands from the computer. The camera employed is a PlayStation Eye Camera [28], capable of sustaining a 187 fps in a Linux system with a simple driver tweak.

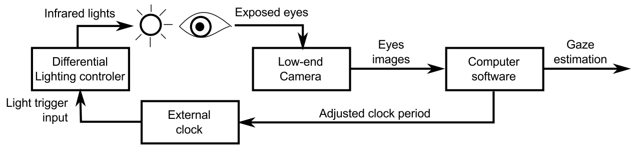

Figure 7. Physical setup of the proposed system.

The structured light source has two main parts. The boards equipped with the LEDs (mounted at the camera plane), and the power circuit (responsible for lighting the LEDs). The electronic circuit that drives the infrared lights is shown in Figure 8. The LEDs are arranged in series and are driven by a low internal resistance N-MOSFET IC. The voltage is controlled by a separate board based on the LM2596 switching power converter IC. In total, sixteen OSRAM SFH4050 [26] LED chips are used.

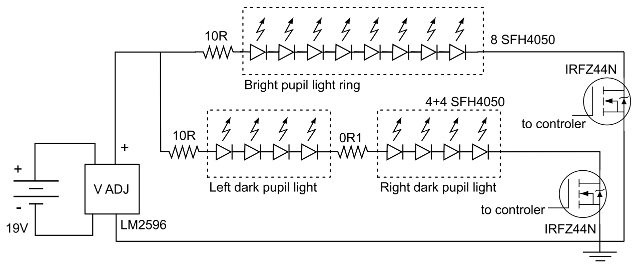

Figure 8. Light emitting diodes power board

The on-axis LEDs, responsible for the bright pupil effect, were soldered to a ring with $13mm$ of external diameter, and mounted directly onto a CCTV 12mm lens. The off-axis LEDs were soldered to a board adapted to the lens mount, divided into two groups $45mm$ apart of each other. Figure 9 shows the actual prototype.

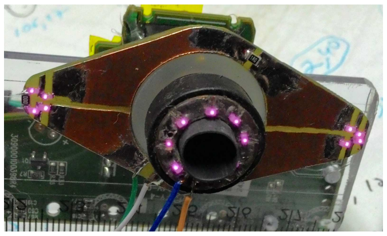

Figure 9. Actual LED boards.

The controller is built around Arduino Boards that controls the pulses which drive the LED boards. The pulse is triggered by an external interrupt pin on the board or by an internal timer (clk_e). As part of the interrupt routine, the microcontroller configures an internal timer to turn off the LEDs after a desired period (Δ_{strobe}).

The controller and the camera are connected via USB ports to the computer. The variables that can be controlled are the period between two strobes ($1/clk_e$) and the strobe length (Δ_{strobe}). The computer employed was equipped with an AMD Turion(tm) II P560 Dual-Core Processor with 2.5 Ghz and 6 GB of RAM. The operating system installed was a Debian Wheezy, 64-bit kernel version 3.2.65-1.

3. Experimental Results

To evaluate the synchronization capabilities of our method, we have collected data about the estimated camera period and the drift in synchronization over time. The camera estimation is done in two steps, a gross estimation based on frames delivered to the host computer and a fine estimation, which finds the clk_e period that is closest to the camera's, using the microcontroller crystal oscillator as reference. To obtain the drift, we modded the camera [1] and configured the board to capture the true VSYNC and compared it against clk_e over time.

To avoid the problems related to getting reliable ground truth data of real pupil contours, we compared the performance of two eye trackers, one using the SDL pupil tracker and the second using the Starburst algorithm [15]. Except for the pupil detection and tracking mechanism, the remaining components of the eye trackers are the same.

The Starburst integrates feature-based and model-based approaches to the task of tracking the eye movements. It first detects features of the pupil boundaries, fits an ellipse to the feature locations and then use a model-based optimization to improve the ellipse fitting. The algorithm also finds the corneal reflection and through interpolation remove it from the image. The algorithm was made available in an open-source package which makes it a good choice for testing.

3.1. Experimental Protocol

Five people (2 female) from 30 to 57, average 37.4 years old, volunteered for the data collection. Each volunteer was asked to sit comfortably at about 60 cm away from a 22" monitor.

The accuracy of each method was evaluated using a typical eye tracking experiment [20] that consists of showing a collection of points on the computer screen at known locations and measure the error distribution.

To evaluate the accuracy, the users were asked to maintain their head as stable as possible while looking at the center of a white target (a cross hair) presented on a black background. A total of 35 targets were displayed, one at a time, approximately positioned on a 7×5 grid covering the whole surface of the screen. For each target, users had to press a key while fixating their gaze at the target. 9 out of the 35 points were used to fit the parameters of a second order polynomial used to map the pupil-corneal reflection vector to a point on the computer screen.

The same data was used to evaluate the performance of the SDL and the Starburst based eye trackers. Data was collected at 187 Hz using our head mounted PS3 prototype with stroboscopic lights. Because the Starburst algorithm from [15] only process dark pupil images, only such images were used to evaluate the Starburst algorithm.

The Starburst implementation used for comparison is available online [15]. The C version lacks the optimization step, a Nelder-Mead Simplex search [21], which was added for completeness. Despite our effort to make a fair work, the effect of some changes are hard to predict. In the original work, only one glint was produced. Therefore, the algorithm had to be changed to accommodate the change in illumination, as the off-axis lights now produces two glints, instead of one. Moreover, to detect the pupil border, the algorithm utilizes a step parameter, found to be optimal when set to 7 in the original work. However, no studies were made to assess if this value is best for our setup.

3.2. Results

Figure 10 shows the gross camera period estimation as described in the first section of Algorithm 1 (top), and the fine estimation using the column image gradient (bottom). The former is computed based on the computer's clock, while the second uses the clock of the microcontroller. The figure is the result of one hundred external clock adjustments done independently, showing similar results over the sections. With similar, we understand that a error of $2\mu s$ is sufficiently small, considering that a line takes about $19\mu s$ to be read on the actual setup.

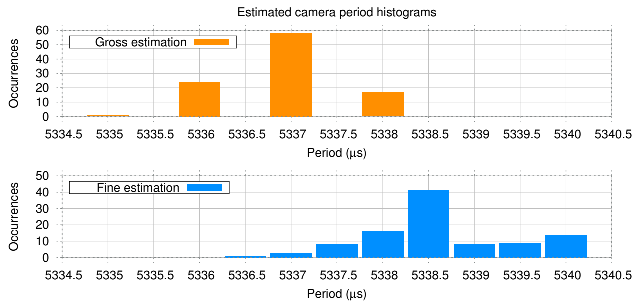

Figure 10. Estimated camera periods.

Figure 11 shows the drift, in time, between the true camera VSYNC and the adjusted external clock for a section of about $64s$ of USER1. Positive values means that clk_e was triggered after the camera VSYNC, while negative values means the opposite. In both cases, the results show that the triggering was kept inside the invisible scanlines.

We have also calculated the time each algorithm takes to process a frame. Figure 12 depicts the results for USER3. The other users showed similar results. It is important to note that Starburst was unable to process all the frames at the time window of $1/187$ of a second.

The figure does not take into account the time needed to keep the external clock synchronized. In the actual implementation, the drift in synchronization is evaluated by the algorithm only five times per second, consuming a mean of $405\mu s$ ($SD = 35.7\mu s$), giving a time averaged $10.9\mu s$ of CPU time per frame. The evaluation is done at such steps

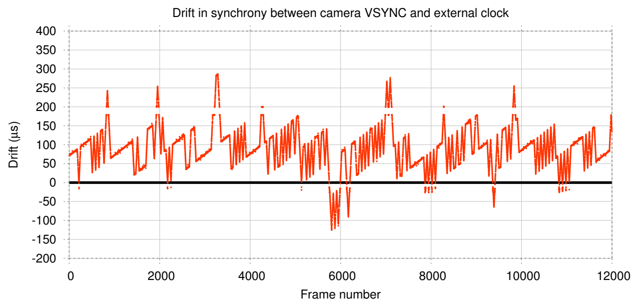

Figure 11. Drift in synchrony as detected by the strobo controller board during a $64s$ capture.

because the clk_e is almost on sync with the camera. Consequently, in $1/5$ of a second, the banding moves no more than 6 scanlines.

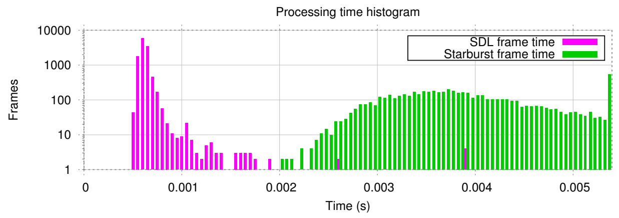

Figure 12. Histogram of frame processing time for USER3. The last column to the right accumulates processing times greater than $1/187$ of a second.

Figure 13 and Figure 14 show the error distribution using the 35 points placed on a 7×5 grid, using the Starburst and the SDL algorithm to detect the pupil contours. The circle diameters are proportional to the standard deviation of the distance to the true point locations. 9 out of the 35 points were used to calibrate a second order polynomial [19], so both methods are compared using the same gaze estimation function. Table 1 summarizes the overall error of each algorithm for each section. A paired t-test indicated that Starburst ($M_{Starburst} = 2.30°, SD_{Starburst} = 1.09°$) performed significantly different from the SDL ($M_{SDL} = 1.29°, SD_{SDL} = 0.62°$) for the frames analyzed ($t(9941) = 35.6, p < 0.001, 99\% CI[0.93°, 1.08°], d = 1.00°$).

The parameters used for the SDL algorithm were a dark pupil glint threshold of 0.8 and a bright pupil glint threshold of 0.91. The thresholds are percentages of the brightest spot in the frame. The number of rays used in the refinement was 30. Lastly, a threshold is defined in a per session fashion for the sensitive detection of the overlap area, as the bright pupil response is idiosyncratic and changes with gaze direction [22, 2]. The parameters used for the Starburst algorithm were a corneal reflection window size of 100×60 pixels, a starting threshold of 20 and a minimum candidate feature of 3. The number of rays used was 10, a number which guarantees the performance of the algorithm [15]. The starting point was manually initialized inside the pupil.

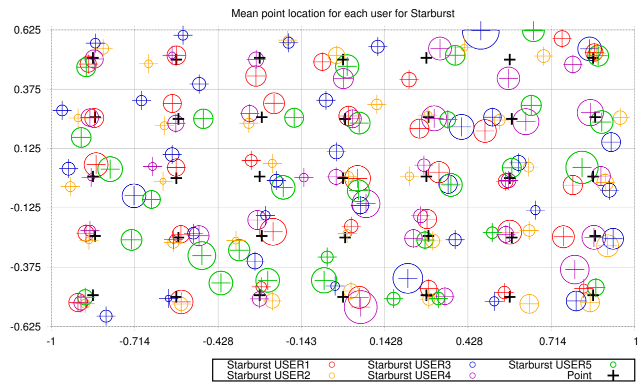

Figure 13. Starburst, mean error per user.

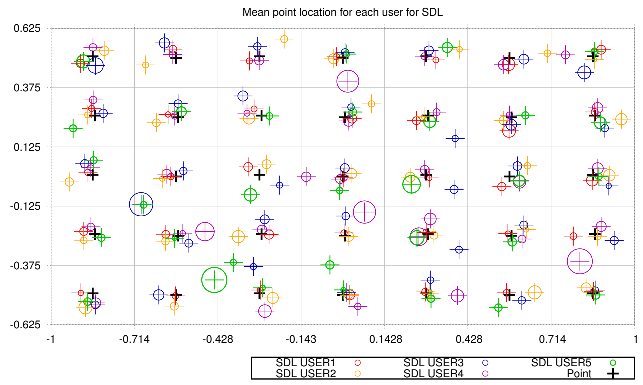

Figure 14. SDL, mean error per user.

Volunteer	$M_{Starburst}$	$SD_{Starburst}$	M_{SDL}	SD_{SDL}
USER1	2.755	1.252	0.846	0.516
USER2	1.331	0.720	1.135	0.526
USER3	3.100	1.010	1.382	0.572
USER4	2.459	1.259	1.732	0.784
USER5	2.957	1.247	1.588	0.715

Table 1. Mean error and standard deviation in degrees over the 35 points.

3.3. Discussion

The mean error of the Starburst is greater than found on literature [15]. We attribute such errors to both the experimental protocol, and the eye images. In [15] the users fixated their heads in a chin rest, which reduces the head movements considerably. Fixating the head is common place in literature [18, 4], and have already been shown to reduce the overall error [6]. However, as the algorithms are compared with each other, the same errors introduced in one due to the head slippery are expected to affect the other.

In the images taken in [15], the eye occupies a bigger portion of the image, which in turn, increase the range of the pupil-glint vector. In our tests, a translation of only one pixel resulted in more than a degree of error. We believe that both the unrestricted head and the small eye images contributed for the increasing in error of the Starburst.

The SDL have three thresholds, one for the bright pupil glint, one for the dark and one for the pupil overlap region. However, they are straightforward to set. The glint thresholds, in particular, did not have to be changed for the different users once set. They were chosen as a percentage of the brightest image region which maximized the glint coverage. Therefore, despite being fixed, the actual thresholds may vary throughout the experimental section. The overlap area threshold is more user sensitive, as is the bright pupil response [22, 2]. The minimum difference between the mean bright and dark pupil, over various gaze directions, is chosen as threshold in each section.

4. Conclusion

The gaze estimation is an important part in the design of assistive interfaces for people with severe high-level motor disabilities. In this context, we have presented a computation efficient method for pupil detection and tracking using stroboscopic differential lighting (SDL).

The original differential lighting technique uses light sources synchronized with the video frames to create bright and dark pupil images. The method was originally developed for interlaced analog cameras and, therefore, was limited to 60 frames per second (fps). The SDL technique uses very short light pulses that will, most likely, be captured during only one frame. When the pulse is close to the beginning or end of the frame, our method detects a banding that is used to delay the next pulse, i.e., no external sync is required from the camera.

Another advantage of our technique is the reduction of motion blur during fast eye movements, which contributes to a higher accuracy. SDL also extends previous DL by computing the real pupil contours instead of the overlap pupil region between consecutive frames, further improving the accuracy of the method.

We have implemented a 187 fps prototype using a low cost PS3 digital camera. The on and off-axis light sources are controlled by an Arduino board. We have conducted an experiment with 5 volunteers. Using a full second order polynomial with 9 calibration points, our method showed significant better results than the Starburst algorithm. Using 35 target points on a 22” monitor, SDL average error was 1.29°, while the average error of Starburst was 2.30°.

Our technique can be used with any digital camera, which allows the construction of high quality eye trackers from more affordable and readily available hardware. Therefore, more cost-effective eye trackers can be built, benefiting a vast number of users with impairments.

Acknowledgements

This work was funded by Fundação Araucária (projeto DINTER UTFPR/IME-USP) and CNPq.

References

[1] *Camera Synchronization Report*, Jun 2012. *Available at* `http://nuigroup.com/?ACT=28&fid=34&aid=7360_7LpLFlSc5HWnVlvjHemf`.

[2] J. S. Agustin, A. Villanueva, and R. Cabeza. Pupil brightness variation as a function of gaze direction. In *Proceedings of the 2006 symposium on Eye tracking research & applications*, pages 49–49. ACM, 2006.

[3] D. Bradley, B. Atcheson, I. Ihrke, and W. Heidrich. Synchronization and rolling shutter compensation for consumer video camera arrays. In *Computer Vision and Pattern Recognition Workshops, 2009. CVPR Workshops 2009. IEEE Computer Society Conference on*, pages 1–8. IEEE, 2009.

[4] M. De Luca, D. Spinelli, P. Zoccolotti, and F. Zeri. Measuring fixation disparity with infrared eye-trackers. *Journal of biomedical optics*, 14(1):014013–014013, 2009.

[5] Y. Ebisawa. Improved video-based eye-gaze detection method. *IEEE Transactions on Instrumentation and Measurement*, 47(4):948–955, 8 1998.

[6] O. Ferhat, F. Vilarino, and F. Sánchez. A cheap portable eye-tracker solution for common setups. *Journal of Eye Movement Research*, 7(3):2, 2014.

[7] O. Ferhat, F. Vilarino, and F. Sanchez. A cheap portable eye-tracker solution for common steups. *Journal of Eye Movement Research*, 7(3):1–10, 2014.

[8] M. A. Fischler and R. C. Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, 1981.

[9] A. W. Fitzgibbon and R. B. Fisher. A buyer's guide to conic fitting. In *Proceedings of the 5th British Machine Vision Conference*, pages 513–522, 1995.

[10] C. Hennessey, B. Noureddin, and P. Lawrence. A Single Camera Eye-gaze Tracking System with Free Head Motion. In *Proceedings of the 2006 Symposium on Eye Tracking Research & Applications*, pages 87–94, 2006.

[11] C. Hennessey, B. Noureddin, and P. Lawrence. Fixation Precision in High-Speed Noncontact Eye-Gaze Tracking. *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, 38(2):289–298, April 2008.

[12] C. Holland and O. Komogortsev. Eye tracking on unmodified common tablets: challenges and solutions. In *Proceedings of the Symposium on Eye Tracking Research and Applications*, pages 277–280. ACM, 2012.

[13] M. Kassner, W. Patera, and A. Bulling. Pupil: An Open Source Platform for Pervasive Eye Tracking and Mobile Gaze-based Interaction. *arXiv:1405.0006*, April 2014.

[14] E. S. Kim, A. Naples, G. V. Gearty, Q. Wang, S. Wallace, C. Wall, M. Perlmutter, J. Kowitt, L. Friedlaender, B. Reichow, F. Volkmar, and F. Shic. Development of an untethered, mobile, low-cost head-mounted eye tracker. In *Proceedings of the Symposium on Eye Tracking Research and Applications*, ETRA '14, pages 247–250, New York, NY, USA, 2014. ACM.

[15] D. Li and D. J. Parkhurst. Starburst: A robust algorithm for video-based eye tracking. *Elselvier Science*, page 6, 2005.

[16] K. Lukander, S. Jagadeesan, H. Chi, and K. Müller. OMG!: A New Robust, Wearable and Affordable Open Source Mobile Gaze Tracker. In *Proceedings of the 15th International Conference on Human-computer Interaction with Mobile Devices and Services*, MobileHCI '13, pages 408–411, New York, NY, USA, 2013. ACM.

[17] P. Majaranta, H. Aoki, M. Donegan, D. Hansen, J. Hansen, A. Hyrskykari, and Räihä. *Gaze Interaction and Applications of Eye Tracking: Advances in Assistive Technologies*. IGI Global, October 2011.

[18] A. Minken and J. Van Gisbergen. A three-dimensional analysis of vergence movements at various levels of elevation. *Experimental Brain Research*, 101(2):331–345, 1994.

[19] C. Morimoto, D. Koons, A. Amir, and M. Flickner. Pupil detection and tracking using multiple light sources. *Image and Vision Computing*, 18(4):331–335, 2000.

[20] C. H. Morimoto and M. Mimica. Eye gaze tracking techniques for interactive applications. *Computer Vision and Image Understanding*, 98:4–24, 2005.

[21] J. A. Nelder and R. Mead. A simplex method for function minimization. *The computer journal*, 7(4):308–313, 1965.

[22] K. Nguyen, C. Wagner, D. Koons, and M. Flickner. Differences in the infrared bright pupil response of human eyes. In *Proceedings of the 2002 symposium on Eye tracking research & applications*, pages 133–138. ACM, 2002.

[23] T. Ohno, N. Mukawa, and A. Yoshikawa. FreeGaze: A Gaze Tracking System for Everyday Gaze Interaction. In *Proceedings of the 2002 Symposium on Eye Tracking Research & Applications*, pages 125–132, 2002.

[24] OmniVision Technologies, Inc. *OV7725 Color CMOS VGA OmniPixel2TM CAMERA CHIP Sensor*, Mar. 2007.

[25] ON Semiconductor. *AR0130: 1/3-Inch CMOS Digital Image Sensor*, Nov. 2014.

[26] OSRAM Opto Semiconductors. *SFH 4050 High Power Infrared Emitter (850 nm)*, Nov. 2014.

[27] B. Pires, M. Devyver, A. Tsukada, and T. Kanade. Unwrapping the eye for visible-spectrum gaze tracking on wearable devices. In *Applications of Computer Vision (WACV), 2013 IEEE Workshop on*, pages 369–376, Jan 2013.

[28] C. Sony. PlayStation Eye Camera. [Online, 14/jan/2014], 2014.

[29] Texas Instruments Incorporated. *SN65LVDS324 1080p60 Image Sensor Receiver*, Mar. 2015.

[30] C. Theobalt, I. Albrecht, J. Haber, M. Magnor, and H.-P. Seidel. Pitching a baseball: tracking high-speed motion with multi-exposure images. *ACM Transactions on Graphics (TOG)*, 23(3):540–547, 2004.

[31] A. Tsukada and T. Kanade. Automatic Acquisition of a 3D Eye Model for a Wearable First-person Vision Device. In *Proceedings of the Symposium on Eye Tracking Research and Applications*, pages 213–216, 2012.

[32] D. Zhu, S. Moore, and T. Raphan. Robust pupil center detection using a curvature algorithm. *Computer Methods and Programs in Biomedicine*, 59:145–157, 1999.