Expert Systems With Applications 64 (2016) 69–83

Contents lists available at ScienceDirect

Expert Systems With Applications

journal homepage: www.elsevier.com/locate/eswa

Evaluation of temporal stability of eye tracking algorithms using webcams

Jose Gómez-Poveda, Elena Gaudioso*

Artificial Intelligence Department, Computer Science School, Universidad Nacional de Educación a Distancia, Spain

ARTICLE INFO

Article history:
Received 11 March 2016
Revised 20 June 2016
Accepted 19 July 2016
Available online 22 July 2016

Keywords:
Eye tracking
Webcam
Temporal stability
Noise
Relative error
Efficiency

ABSTRACT

Eye tracking methods are usually focused on obtaining the highest spatial precision as possible, locating the centre of the pupil and the point of gaze for a series of frames. However, for the analysis of eye movements such as saccades or fixations, the temporal precision needs to be optimised as well. The results should not only be precise, but also stable. Eye tracking using low-cost hardware such as webcams brings a new series of challenges that have to be specifically taken into account. Noise, low resolution and low frame rates are some of these challenges, which in the end are the cause of temporal instabilities that negatively affect the results. This paper proposes a measure for temporal stability of pupil detection algorithms, applied on video streams obtained from webcams. The aim of this metric is to compare and evaluate the temporal stability of different algorithms (following a multi-layered approach for pupil detection), in order to identify which one is more adequate to its use for movement detection using low-cost hardware. The obtained results show how the temporal stability of different algorithms is affected by several factors.

1. Introduction

Many applications seek to provide a tailored user experience by collecting data from the user. This data can be obtained by asking the user, by means of application settings or surveys, or it can be inferred from the interactions of the user with the system. Whenever possible, the latter is the preferred method, as it does not interfere with the user.

An important research area nowadays is centred on the use of eye tracking technologies to collect and analyze information about the users when they interact with a computer. Specially, eye tracking technologies are used for understanding the user's distribution of attention (Zhu et al., 2016). For example, gaze tracking systems are created with the purpose of finding out where is the user looking at, called the point of gaze (Jafari & Ziou, 2015). Other properties that can be detected are the duration of fixations (Manor & Gordon, 2003), the patterns of gaze scans and gaze heat maps (Drusch, Bastien, & Dinet, 2011), the diameter of the pupil (Wang, 2011), and the frequency of blinks (Bentivoglio et al., 2011).

Due to their ability to collect such information, eye tracking systems have been widely employed in usability studies (Nielsen & Pernice, 2009), for example, to detect search patterns in maps (Ooms, Andrienko, Andrienko, De Maeyer, & Fack, 2012). Nevertheless, eye tracking systems are not only used to assess how the user is using the software (Vrzakova & Bednarik, 2015), but they are used in very diverse applications (Richardson & Spivey, 2016): analysis of visual attention in learning systems (Tsai, Hou, Lai, Liu, & Yang, 2012), analysis of the students' learning process (Lai et al., 2013), to favour the interaction of people with motor disabilities (Porta & Ravarelli, 2012), or even to monitor the user's mental state (Bixler & D'Mello, 2015) or to investigate mood regulation, emotional information processing, and psychomotor disturbances in depressive disorders (Carvalho et al., 2015). Another important application for eye tracking is language research, where eye tracking is used to detect the level of attention of the user when reading or learning a second language (Ellis et al., 2014; Ibrahim & Raney, 2016).

Typically, eye tracking has been widely studied in the field of artificial vision using sophisticated hardware, such as search coil contact lenses (Kenyon, 1985), cameras mounted in eyeglass frames (Li, Babcock, & Parkhurst, 2006), electro-oculography (Barea, Boquete, Ortega, López, & Rodríguez-Ascariz, 2012; Deng, Hsu, Lin, Tuan & Chang, 2010), or through the use of chin rests (Makris, Hadar, & Yarrow, 2013). Trackers using these hardware solutions are in general able to achieve accurate results since the eye is always centred and zoomed in the picture, with the increase of effective resolution that this implies. Furthermore, tracking algorithms are simpler, as it is not necessary to locate the face and

* Corresponding author.
E-mail address: jgomez1913@alumno.uned.es, elena@dia.uned.es (E. Gaudioso).

http://dx.doi.org/10.1016/j.eswa.2016.07.029
0957-4174/

eyes as a first step, nor is it required to compensate movements, rotations, etc. (San Agustin et al., 2010).

These hardware-based solutions have always been very expensive (Bixler & D'Mello, 2015) and obtrusive for the user. In order to make it convenient and comfortable for the user, a system for gaze tracking should be unobtrusive. Glasses or other devices should not be required, and the movements of the user should also not be limited.

This trend is starting to change, with slightly more affordable solutions appearing in the market, such as Tobii's Dynavox PCEye Mini (Tobii, 2016), Tobii TX300 (Weerasinghe, Elmadani, & Mitrovic, 2015), Kinect (Jafari & Ziou, 2015), faceLAB (Tsai et al., 2012), SMI RED 250 eye tracker (Peterson et al., 2015) or Visual Interaction's myGaze (myGaze, 2014). These new solutions, which use infrared light, are neither economical, nor very widely available.

Given their availability, unmodified webcams are the ideal devices for eye and gaze tracking in computers (Bixler & D'Mello, 2015; Sewell & Komogortsev, 2010), in spite of only being able to capture visible light. In fact, most tablet devices and smart phones have a front-facing camera that could be used as well to estimate the location of the user's gaze on the screen. In the interest of clarity, only webcams are mentioned in the rest of the text, even though most of the algorithms, conclusions, advantages and disadvantages do apply as well to the cameras of mobile or tablet devices.

As a matter of fact, there are already promising alternatives for eye tracking using commodity webcams. For example, for gaze detection (Asteriadis, Nikolaidis, Hajdu, & Pitas, 2006; Sewell & Komogortsev, 2010), or detection of the iris or pupil (Bengoechea, Cerrolaza, Villanueva, & Cabeza, 2014; George & Routray, 2016; Mbouna & Kong, 2012). Some studies have compared the performance of low-cost webcam-based gaze trackers with commercial trackers (San Agustin, Skovsgaard, Hansen, & Hansen, 2009). In any case, these unmodified webcams are more sensitive to variations in lighting and reflections.

Other eye movements, such as drifts, tremors and microsaccades, are very fast and of little scale, so they do require specialised hardware in order to be detected. High-frequency movements can happen as part of the normal fixations, or for example, when users have some kind of vision impairment, such as nystagmus[1] (Quiros & Yee, 2014). Detecting these high-frequency movements is important for accessibility software for people with impaired vision.

Regardless of the hardware used, studies about eye tracking are usually focused on obtaining the highest spatial precision as possible, locating the centre of the pupil and the point of gaze for a series of frames, such as in Sesma, Villanueva, and Cabeza (2012) and George and Routray (2016).

However, for the analysis of the movements such as saccades or fixations, the temporal stability has to be optimised as well. I.e. the results should not only be as precise as possible, but also stable and consistent between consecutive frames. Otherwise, the detected features would appear to shake erratically when applied to a video stream. The eye movement detection algorithms would detect false saccades and other non-existent movements. Applying algorithms in order to smooth the positions retrieved in a sequence of frames would carry a loss of information, which would make it impossible to detect high-frequency movements.

The retrieved position must be consistent between frames even in the presence of noise, which is common with low cost webcams. This noise comes from the CCD[2] or CMOS[3] sensors (Hui, 2000), or from the lossy compression often used by the hardware and the driver (Zhengting, 2007). Noise itself accounts for a big portion of the temporal instabilities that can be observed when working with low-cost and portable cameras such as webcams (Han, Yang, Kim, & Gerla, 2012; Holland & Komogortsev, 2012; Mbouna & Kong, 2012).

There are existing studies that somehow relate to temporal analysis in eye tracking systems. Grother (2013) analyses the stability of iris recognition as persons get older, i.e. that people keep being recognised in spite of the physical changes produced by age. Chamaret and Le Meur (2008) aim to keep stability on the cropping of sections of videos using Kalman and temporal filters, and apply eye tracking to evaluating the number of fixations within the cropped area. These refer to different aspects of temporal stability and use eye tracking differently, as opposed to the current work, which evaluates the temporal stability of the results obtained by pupil detection algorithms.

This paper presents a measure for temporal stability of eye tracking algorithms applied on video sources from webcams. The aim of this measurement is to compare and evaluate the temporal stability of different algorithms, in order to identify which one is more adequate to its use for movement detection using low-cost hardware.

The rest of this paper is structured as follows. Before describing the metric proposed for temporal stability, Section 2 presents the multi-layered approach for pupil detection that has been implemented to allow the comparison of state-of-the-art algorithms. Section 3 presents the proposed measurement for temporal stability of eye tracking algorithms. Section 4 and 5 present the evaluation of temporal stability and the results obtained in this evaluation. Finally, Section 6 presents the conclusions and lines for future work.

2. A multi-layered approach for pupil detection

To allow the comparison of state-of-the-art algorithms for eye tracking in terms of precision and temporal stability, a framework based on a multi-layered approach for pupil detection has been implemented, with the layers being: face detection, eye detection, and pupil detection.

2.1. Face detection

The first step in the face detection layer is to locate the face of the subject using a combination of Local Binary Pattern (LBP) (Liao, Zhu, Lei, Zhang, & Li, 2007) and Haar cascades (Papageorgiou, Oren, & Poggio, 1998).

Haar cascades apply to the image a set of features which are combinations of 2, 3 or 4 black and white rectangles. The sum of the pixels under the white rectangles is subtracted from the sum of the pixels under the black rectangles.

As opposed to that, LBP cascades calculate a pattern on basis on the relative average value of the pixels contained in blocks around the current one, binarising the value to whether the average is higher or lower than the current pixel. This results in more distinctive features than Haar's, therefore requiring less stages for achieving good results.

LBP cascades are much faster than Haar cascades also due to the use of integer arithmetic. On the negative side, the LBP cascade provided by OpenCV (Intel Corporation, Willow Garage, & Itseez, 2015), as used in this work (refer to Section 4) is less precise, fails to detect some cases, and produces some false positives.

[1] Nystagmus, sometimes called "dancing eyes", is a condition in which one or both eyes move involuntary and uncontrollably quickly. The movement usually alternates between slow and fast and involves both eyes.

[2] CCD: Charge-Coupled devices.

[3] CMOS: Complementary Metal-Oxide-Silicon.

Given that these cascades are applied to the full input image, it is important to speed up this process as much as possible. In the present approach, face detection is applied first on a region of interest determined by the position of the face in the previous frame. If a face is not found in that region, the search is extended to the rest of the image. This reduces the search area considerably in most of the cases.

For each of the previously mentioned cases, after a rough and a detailed LBP search, a third search is done using a Haar cascade, in order to recover some of the faces that would have been identified by a Haar cascade, without the performance hit that would cause the use of Haar cascades as a first resort.

2.2. Eye detection

The second step is to locate the eye regions. Using knowledge about the proportions of the face (Brandreth, 1986), a rough estimation of the area where the eyes are located is done. In case this eye region is very small, it is scaled up in order to increase the percentage of times the pupils are found by the eye cascades. This could happen in a combination of several factors: the low resolution of the camera, the face being too far away from the camera, or the Field of Vision of the camera being too high.

Once the approximate eye region is located, Haar cascades are used, as they provide better precision and better temporal stability than LBP cascades. In particular, a version of the Viola–Jones algorithm (Viola & Jones, 2001) enhanced by (Lienhart and Maydt, 2002) is employed.

A multi-stage approach is taken, applying the eye cascade every time with a smaller scale factor. This allows detecting the eyes in most of the cases using a large scale factor (and therefore much faster to execute), while having a smaller, slower to execute factor as a back-up for those cases where the eyes are not found in the first attempt.

In those cases where the face detection algorithm fails to find the face, the eye cascade is applied to the whole image, in a last attempt to find the eyes in a situation where the person may be too close to the camera, so the whole face does not fit into the camera's Field of View, or may be on the left or right sides of the camera and the face may only be partially visible.

If only one eye is detected, a simple heuristic is used for deducing which is the detected eye. If the detected eye lies in the right side of the captured image, this is most likely the right eye, and the left part of the face may not be visible. On the other hand, and following the same logic, if the detected eye lies in the left side of the captured image, this is most likely the left eye.

2.3. Tracking additional facial features

On top of the aforementioned features, it can be useful to track other facial features as well. As Haar-style cascades can often return false positives, the nose, and more specifically the nostrils, mouth and eyebrows can be used in order to help locating and validating geometrically other detected features. I.e. a false positive of a face can be discarded if the nose and/or mouth are not detected. Instead, the face detection algorithm could choose another face candidate.

The nose and mouth detections are implemented following a similar approach as the eye detection in the above subsection. A multi-stage approach is taken using decreasing scale factors; and if the face is not detected, there is also an attempt to find the nose and mouth in the whole input image. This last stage can be omitted if the features are only being located with the purpose of verifying a detected face.

The region of interest estimated for the nose is centred vertically in the face, leaning slightly to the lower part, and centred horizontally. The mouth is sought using the lower part of the face as a region of interest, centred horizontally.

Fig. 1 displays in blue the features detected using cascades, and in orange the regions of interest that were used in each case.

2.4. Cascades stabilisation

The output from the cascade-based feature detection algorithms, used for detecting the face, eyes, nose and mouth, is very unstable. While this output is perfectly valid for still images, it is not consistent from frame to frame, so it is very shaky when processing a recorded video or a live source such as when using a webcam. This has a negative impact not only in the display of the zoomed eyes, but also in the blink and fixation detections.

Hence, the positions and sizes obtained from the cascade feature detection algorithm are smoothed in time, by combining these values with the moving average, and a constant that indicates the velocity of change of this moving average.

This calculation works fine in the current domain, as the user is mostly looking at the camera. Movements are expected to be mostly parallel to the screen, so the sizes will usually remain fairly stable.

However, this is not the case for the spatial locations. Defining spatial moving averages with constant factors leads to the effect in which the averaged position lags noticeably while following the feature (head / eye / nose / mouth) if the movement is very fast. This could be solved by setting a threshold, above which the position is set instead of averaged, but this generates an irregular output as it changes from averages to pass-through positions.

To solve this, an adaptive calculation has been defined where the moving average has more weight the smaller the movement is. This achieves smooth movements, along the ability to quick and gradually catch up with fast movements, avoiding the shaky output of a threshold-based method.

2.5. Pupil detection

Once each eye region is found, the pupil detection algorithm is applied. For each eye, if the corresponding pupil was successfully found in the previous frame of a video, the pupil search area will depend on the location in the previous frame, by expanding the area where the iris was found by a given factor. The staged approach for pupil detection is shown in Fig. 2.

In case the pupil was not found in the previous frame, or if the analysis is done on standalone images (as opposed to a continuous stream of video, such as the one from a video file or a webcam), the region of interest is just based on the current eye region, taking a rectangular window centred both vertically and horizontally, with half the height of the eye region, and just a bit narrower than the detected eye.

The location of the pupil should be retrieved in an as stable as possible way. However, the eye is the fastest muscle in the body, so the stabilisation used for the cascade-based detection algorithms in Section 2.4 would very likely lose information in the case of pupil detections. For that reason, the aforementioned stabilisation is not applied to the pupil detection.

The following subsections describe the algorithms implemented and compared in the present study. These have been selected as being a significant representation of existing pupil detection algorithms with visible light:

- Hough (Yuen, Princen, Illingworth, & Kittler, 1990)
- Edges Analysis (Edge) (Asteriadis et al., 2006)
- Cumulative Distribution Function (CDF) (Asadifard & Shanbezadeh, 2010)
- Projection Functions (GPF) (Zhou & Geng, 2004)

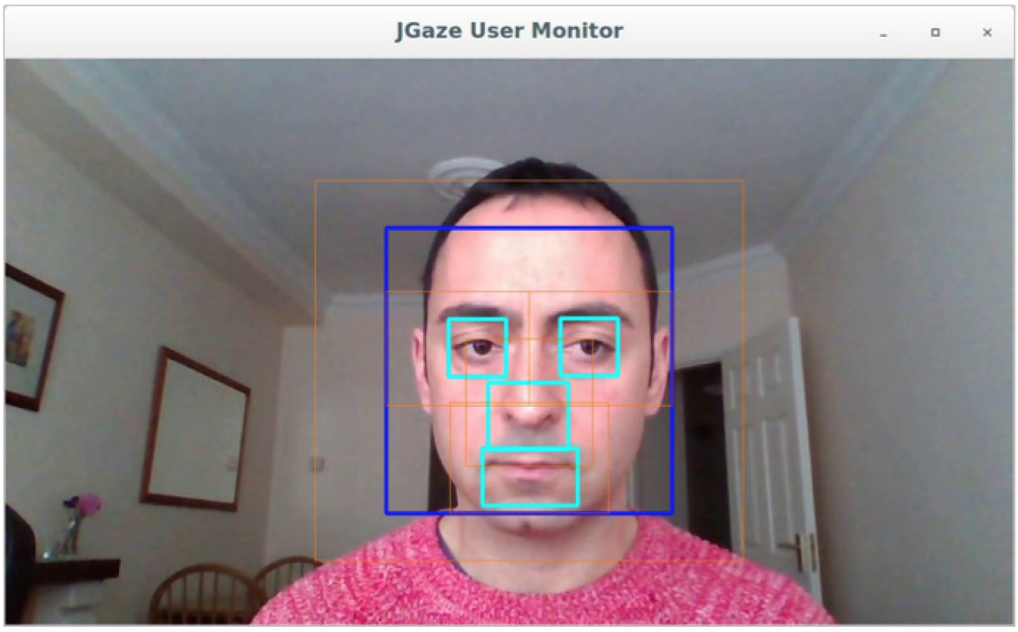

Fig. 1. Detected Facial Features and the areas where they are expected to be found.

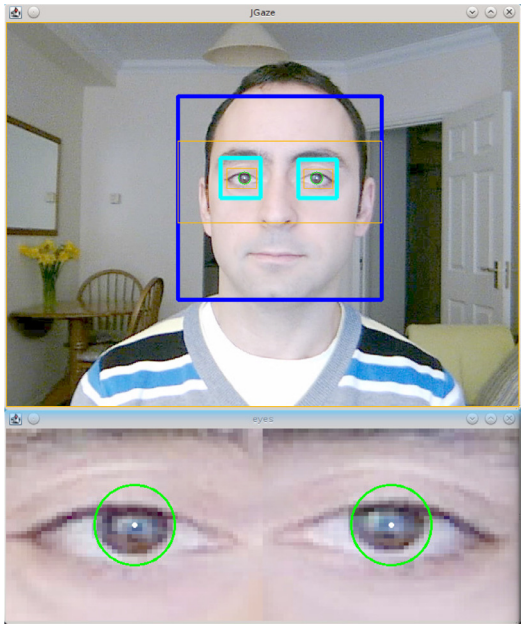

Fig. 2. Pupil detection approach.

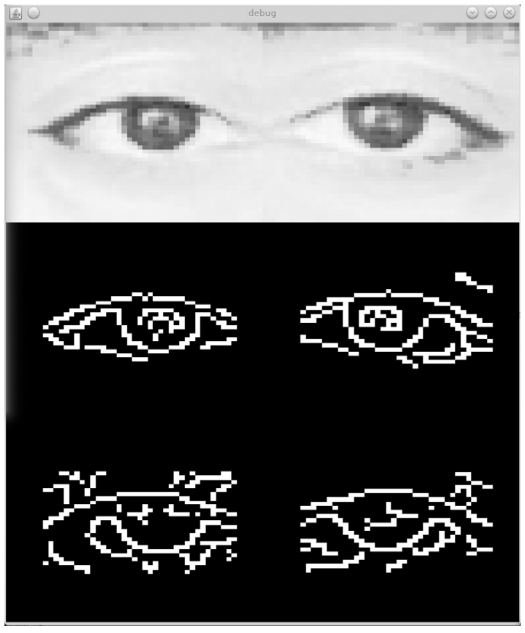

Fig. 3. Steps of the Hough algorithm.

- Combined CDF-GPF (see 2.5.5)
- Gradient (Timm & Barth, 2011)

2.5.1. Hough

After the region of interest of the image is equalised and filtered for removing unwanted noise using a median blur, the Hough transform on circles (Yuen et al., 1990) is applied to the eye region, with the purpose of locating the iris. The pupil is calculated as the centre of the detected iris. Fig. 3 shows the steps of the Hough algorithm.

The approach taken is similar to the one by Kunka and Kostek (2009). Likewise, there is an initial search based on the last position of the pupil. If the pupil is not found in this region, the search is increased to a larger area.

2.5.2. Edges analysis (edge)

The Edge Analysis algorithm (Asteriadis et al., 2006) has been adapted from the original implementation in C++ by Ciesla and Koziol (2012) for their software EyeTracker (Ciesla & Koziol, 2011).

The image within the region of interest is normalised. The Canny algorithm (Canny, 1986) is then applied, with lower and upper thresholds of 1.5 and 2 times the average luminance, respectively. The Edge Analysis algorithm then searches the vertical and horizontal lines with higher number of intersections with detected borders. These steps can be seen in Fig. 4.

If the distance between the two vertical lines with maximum number of intersections is greater than a threshold (i.e. what has been found has a reasonable size), the x position of the pupil is chosen as the middle of these two vertical lines. Otherwise, the

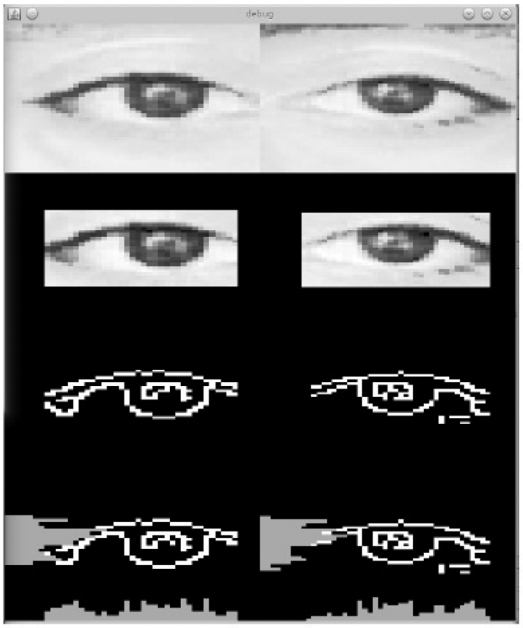

Fig. 4. Steps of the Edge Analysis algorithm.

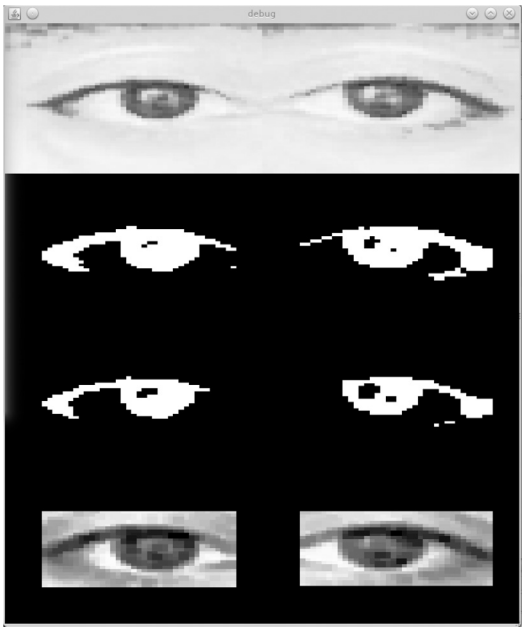

Fig. 5. Steps of the CDF algorithm.

second vertical line is rejected, and the test will be checked against the next vertical line by number of intersections. Finally, the same approach is then taken for horizontal lines, in order to detect the y position of the pupil.

2.5.3. Cumulative distribution function (CDF)

The Cumulative Distribution Function (CDF) algorithm (Asadifard & Shanbezadeh, 2010) has been adapted from the original implementation in C++ by Ciesla and Koziol (2012).

This method is based on the fact that the iris and pupils are dimmer than the cornea.

The image within the region of interest is equalised. Then, the image is binarised using the Cumulative Distribution Function (CDF) of the luminance of the eye. After eroding the image in order to remove artefacts, the minimum value is chosen (PMI or Pixel with Minimum Intensity).

Finally, the average luminance of the area around the PMI is calculated. The pupil centre is chosen as the geometrical centre of the points below this average. These steps are shown in Fig. 5.

2.5.4. Projection functions (GPF)

The GPF algorithm (Zhou & Geng, 2004) has been adapted from the original implementation in C++, also by Ciesla and Koziol (2012).

While the previous algorithms discussed are mostly pixel-oriented, the GPF algorithm is more heavily based on mathematical functions, particularly the General Projection Function and associated functions.

The projection functions algorithm follows a similar concept as the CDF one, but having the pixel intensities projected into horizontal and vertical axes. Quick changes in the projection functions determine the division points in the image. The steps are shown in Fig. 6.

2.5.5. Combined CDF-GPF

This method invokes internally the CDF and GPF pupil detections, and averages their results, thus cancelling some random variations of the results. It follows the de Moivre-Laplace theorem (De Moivre, 1738), considering pseudo-random the temporal variations of the detected eye centres in both algorithms, which may in the first instance be caused by random noise or other pseudo-random variations.

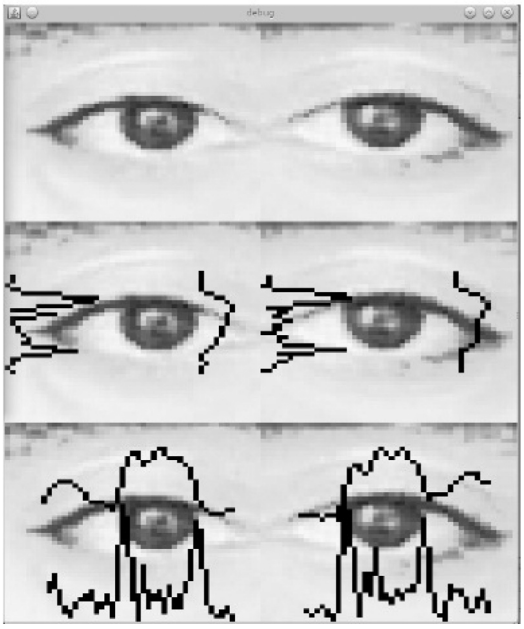

Fig. 6. Steps of the GPF algorithm.

Combination methods may produce better results, especially when mixing pupil detection algorithms that follow different approaches, such as image-based algorithms (like CDF) with feature-based algorithms (such as GPF). The intermediate results can be observed in Fig. 7.

This combined algorithm has been designed in an extensible way, being able to internally invoke any other pupil detection algorithm in the framework, and supporting different combination methods such as average and “best n”.

2.5.6. Gradient

The Gradient algorithm is an adapted and enhanced version from the original implementation in C++ in eyeLike by

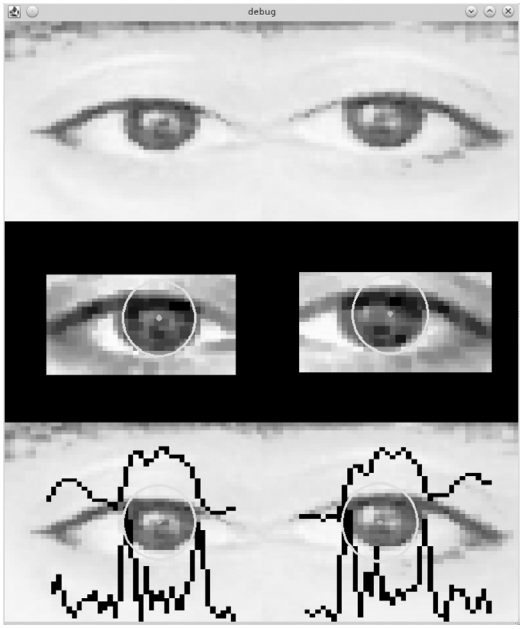

Fig. 7. Steps of the Combined algorithm.

Hume (2013), based on Timm and Barth's gradients algorithm (Timm & Barth, 2011).

This method is based on the search of the centre of a circular object through the analysis of the vector field of image gradients. The circle between the iris and the sclera has a very strong contrast, so this approach can be used to locate it. The centre of the circle is the point where there are more vectors passing through that point towards the circle.

Initially, the part of the image within the region of interest is equalised. After that, all the horizontal and vertical gradients are calculated, along a magnitude of those gradients which is used in order to normalise the original gradients.

For each pixel, a value is set as the accumulation of the vectors that cross from this pixel to the normalised gradients found in the scene. This takes into account the orientation of the vectors (from the point to the feature, and from the feature to its outer side), through the calculation of the dot product of these vectors.

This calculation has been modified from the original implementation, so as to avoid performing many square root calculations per pixel, thus achieving a significant performance increase. Also, most calculations are avoided in cases where the dot product would result negative, as these are not relevant for the final result.

Following, a matrix of weights is created, to assign more weight to the darker pixels (as the iris is darker than the sclera). More weight is also given to the pixels in the central vertical area than in the top and bottom extremes. This is a modification on Timm and Barth's algorithm, which sometimes can confuse the pupil with some areas around the eyebrows, even with post-processing enabled.

A post-processing step removes the areas with maximum values detected around the borders, so as to remove false positives caused by noise or other features (eyebrows, hair, glasses, etc).

Finally, after weighting and post-processing, the point with the highest value is chosen. All these steps are shown in Fig. 8.

3. Metric for evaluating the temporal stability

Temporal stability is typically analysed to compare user identification or other obtained results across long periods of time (years as subjects age, experiments with different parameters, etc.). As opposed to that, the goal of the present work is to optimise the temporal stability of the eye tracking results between consecutive frames in a video sequence. As this is a different scale of the temporal stability, and applied to a separate domain (position of the pupil), a new measure is needed.

This section describes the metric used to measure the stability of the algorithms described in Section 2. In order to provide a full picture, the description begins with the evaluation of the precision of pupil detection algorithms.

3.1. Relative error

The normalised error, defined originally defined in Jesorsky, Kirchberg, and Frischholz (2001), is commonly used as a means to compare the precision of pupil detection algorithms:

$$d_{eye} = \frac{max(d_l, d_r)}{||C_l - C_r||} \tag{1}$$

As per (1), the maximum distance between the estimated and the correct left and right eye centres (d_l and d_r) is divided by the distance between the correct eye centres (C_l and C_r).

This measurement indicates the error obtained by the worst[4] of both eye estimations, and it is usually represented in efficiency graphs, which represent the percentage of samples that provide at least a certain accuracy, using the calculations in (1). This percent is displayed in the y axis, whereas the accuracy is represented in the x axis. The efficiency graphs used have the following ranges:

- $d_{eye} \leq 0.25 \approx$ distance between the eye centre and the eye corners
- $d_{eye} \leq 0.10 \approx$ diameter of the iris
- $d_{eye} \leq 0.05 \approx$ diameter of the pupil

An error just below 0.25 indicates that the obtained result might be located within the eye. In order to support eye tracking, the error needs to be below 0.05

3.2. Measurement of the temporal stability

The comparison and evaluation of the temporal stability of different algorithms will allow to identify which one is more adequate to its use for movement detection using low-cost hardware.

In the current proposal, the calculation of the temporal stability has been carried out on the basis that the difference of the position of the pupil in sequential frames in a video stream should be small. This is especially true when the frame rate is high and/or the average movement is low:

$$\lim_{t \to 0}(p_1 - p_0) = \lim_{t_1 - t_0 \to 0} s \cdot (t_1 - t_0) = 0 \tag{2}$$

where:

- p_0, p_1: position of an element at times 0 or 1.
- s: movement speed within the time interval $((p_1 - p_0)/(t_1 - t_0))$.
- t_0, t_1: time of the frames 0 and 1.

As described in (2), the difference in the position $(p_1 - p_0)$ or $s \cdot (t_1 - t_0)$ tends to 0 when the frame rate is high, and therefore the difference in time between the frames $(p_1 - p_0)$ tends to 0.

A stable eye tracking algorithm should return, for every frame, results very similar to the ones obtained in the previous frame ($p_0 \approx p_1$). Thus, the most temporally stable algorithm is the one

[4] Some articles employ the error obtained by the best or the average of both eye estimations, therefore obtaining a much better score. However, the use of the worst eye estimation is a more standard measure.

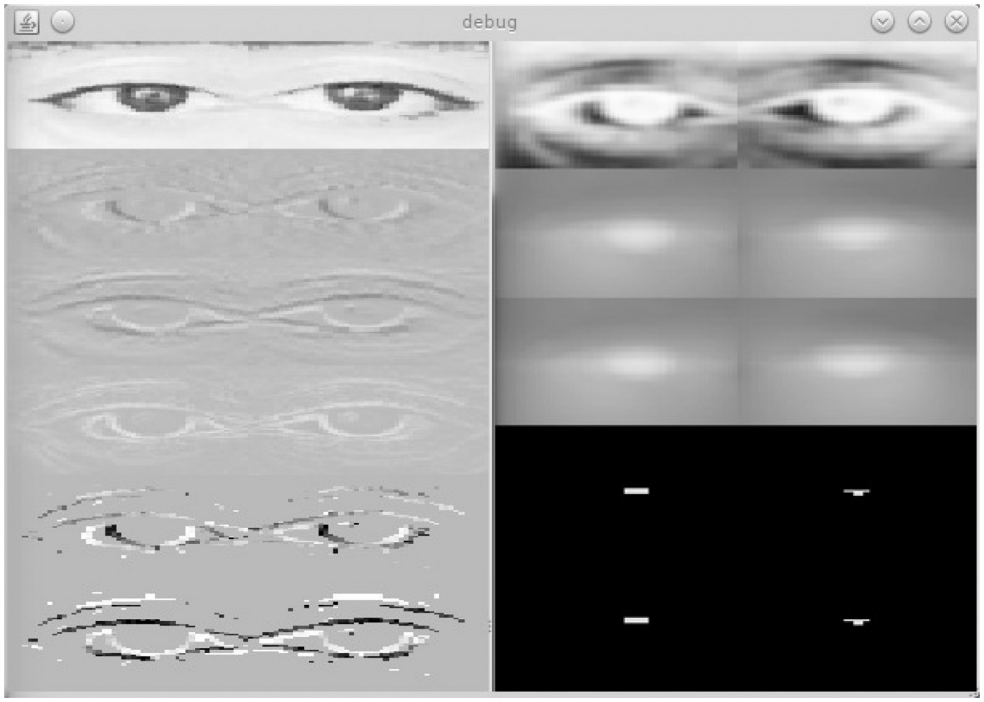

Fig. 8. Steps of the Gradient algorithm.

in which the accumulated differences between positions in successive frames is lower. This is calculated in (3) as the accumulated difference between the positions in the current frame ($position_i$) and previous one ($position_{i-1}$), divided by the number of frames processed minus one.

$$f(x) = \frac{\Sigma_{i=1\ldots n} distance(position_i, position_{i-1})}{numberof frames - 1} \quad (3)$$

This number is then scaled to a resolution-independent measure in (4). In this case, the relation between the vertical resolution ($height_{image}$) and the height of the eye as detected by the eye cascade ($height_{eye}$) has been chosen as a way to make the results generic.

$$f(x) = \frac{\frac{\Sigma_{i=1\ldots n} distance(position_i, position_{i-1})}{numberof frames-1} \cdot height_{image}}{height_{eye}} \quad (4)$$

4. Procedure

The first step in this evaluation has been to capture a series of videos in which the movement was as small as possible. With this very little movement (barely a few pixels in the whole video) divided by the number of frames in the video, for every two frames, the position of the eye should be very similar to the one detected in the previous frame.

To obtain the videos for the experiment, different webcams have been employed:

- Syntek DC-1125 Webcam (HW ID 05e1:0501), integrated in some ASUS laptops released around 2007. Allows the resolutions 1280 × 1024, 640 × 480, 352 × 288, 320 × 240 and 176 × 144, with a very low refresh frequency, which depending on the resolution varies between 8 and 30 Hz.
- Sony Playstation Eye (HW ID 1415:2000), accessory of the Playstation 3 console, sold separately from $13, or included in packs with some video games. This camera presents a higher quality and lower amount of noise, and allows the resolutions 640 × 480 at 30 Hz and 60 Hz, and 320 × 240 at 30 Hz and 120 Hz. These high frequencies are not typical in low-cost webcams. This camera includes the capability of rotating the lens in order to get a much more expanded Field of View (FOV), in which case everything is captured with a lower pixel count, as it appears smaller to the sensor of the webcam.
- Pixart Imaging, Inc. SoC PC-Camera (HW ID 093a:2468), very low-end device which just allows resolutions 176 × 144 and 352 × 288. However, it turns out that this device has a very narrow field of vision, so when the face is in its field of view, the features have very high resolution compared to other cameras with the same resolution. The downside is that the chances that the whole face is completely within the field of view are lower.

With these three cameras, and using the two different FOVs of the Playstation Eye, a series of videos have been recorded, during the day and at night, and with the different frame rates and resolutions supported by each device, making it 32 videos in total. Each video has a fixed length of 10 seconds. This one is called the "Static" dataset in the tests below, focused just on the temporal stability.

In order to make the experiment easier to reproduce, the same procedure has also been followed with the HPEG (Asteriadis, Soufleros, Karpouzis, & Kollias, 2009) and Boston University (La Cascia, Sclaroff, & Athitsos, 2000) public datasets. These have been selected since they are freely available to download, and consist on video recordings, in contrast to the more common BioID database (Jesorsky et al., 2001), which is based on images.

The BioID dataset consists of pictutes taken from 23 subjects; the Boston dataset has videos recorded with 6 subjects in them, and the HPEG dataset uses 10 subjects for its videos. The Static dataset has been recorded with one single subject, since its purpose is to help evaluate the temporal stability across different devices, resolutions, framerates and amount of light, but keeping other parameters constant (same subject, distance to the camera, amount of movement, etc.).

Given that these datasets have been created with the purpose of testing gaze detection algorithms, the point of gaze is tagged in

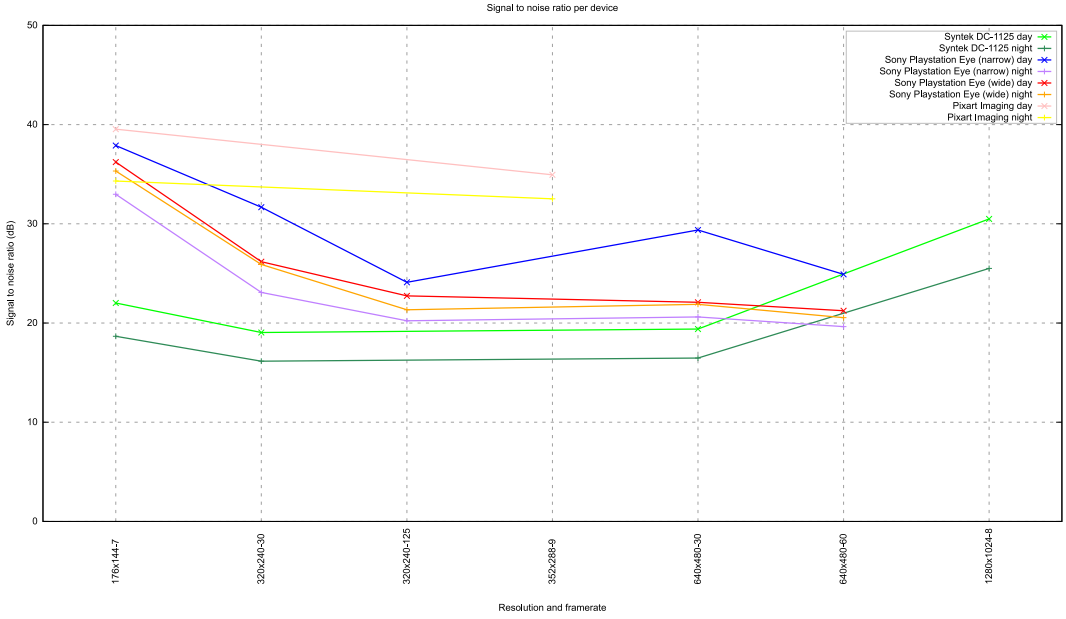

Fig. 9. Signal to noise ratio per device.

each frame; however, the position of the pupil is not. Additionally, the head, eyes and pupils of the persons in the videos do move substantially, as opposed to the Static dataset. Still, a comparison on the positions retrieved in consecutive frames has been calculated, as an alternative measure of temporal stability. Even though the average movement per frame is orders of magnitude higher than in the Static dataset, the average amount of movement in each frame with regard to the previous one should be small (see (2)), as long as the frame rate is kept over a minimum frequency.

Once the videos were recorded, the noise present in the videos in the Static dataset was measured by means of the Signal to Noise Ratio (SNR). Higher values of SNR represent a stronger signal (in this case a clearer image), less interfered by the noise. Lower values represent a weak and noisy signal, where a stable detection will be much harder to achieve.

In this work the SNR is calculated using the PSNR (Peak signal-to-noise ratio) algorithm (Intel Corporation, Willow Garage, & Itseez, 2011), which checks frame by frame the differences between images, starting from the mean squared error (MSE) as indicated in (5):

$$MSE = \frac{1}{c \cdot i \cdot j} \cdot \Sigma (I_1 - I_2)^2 \tag{5}$$

Where I_1 and I_2 are the two images to compare, i and j are the dimensions of these images, and c is the number of channels of the images.

The PSNR is calculated as indicated in (6):

$$PSNR = 10 \cdot log_{10}\left(\frac{MAX_I^2}{MSE}\right) \tag{6}$$

Being MAX_I^2 the maximum value for a pixel (255 in case of an 8-bit image with a single channel), and MSE the mean squared error calculated in (5).

Results per device, resolution and condition (day/night) are shown in Fig. 9, in a logarithmic scale. The following statements are established:

- The Playstation Eye webcam has a much higher signal to noise ratio than the Syntek camera.
- Videos captured during the day, with more illumination, show a higher signal to noise ratio than the videos captured at night.
- Captures performed at higher frame rate show a lower signal to noise ratio than those captured under a lower frame rate.

In order to enable the actual comparison between different algorithms, an automated testing framework has been developed that enables testing automatically all the available algorithms against all the configured data sources. The application has been implemented in Java using the OpenCV library (Intel Corporation et al., 2015) invoked through its Java wrapper, achieving a performance similar to the one expected in an implementation in C/C++.

To test the statistical significance of the obtained results, a series of ANOVA tests were carried out to calculate the statistical significance for multiple groups, while independent samples t-tests were applied when only 2 groups were present. Where required, a post-hoc Games-Howell analysis was performed, as in all cases the Levene tests returned that equal variances could not be assumed. Significance level was set to 0.05. All statistical analyses were performed in GNU PSPP (version 0.8.5).

The developed application is available as an open-source framework (Gomez-Poveda, 2014). It is an extensible framework that enables to easily add implementations of additional pupil detection algorithms. These added algorithms will be run in the test suites, and appear in the comparison graphs, whose generation is automated using the Gnuplot tool (Williams & Kelley, 2012).

5. Results

The results obtained by the algorithms described in Section 2 are discussed in the following sections. These include the relative error for alignment with other eye tracking articles, and the temporal stability.

5.1. Calculation of the relative error

The Relative Error is represented using Efficiency graphs, which, as described in Section 3.1, display in the y axis the percentage of samples that provide at least a certain accuracy, whereas the accuracy is represented in the x axis. These are scaled by the number of images in which the position of both eyes have been located, as it is the only case in which the formula for the relative error can be calculated.

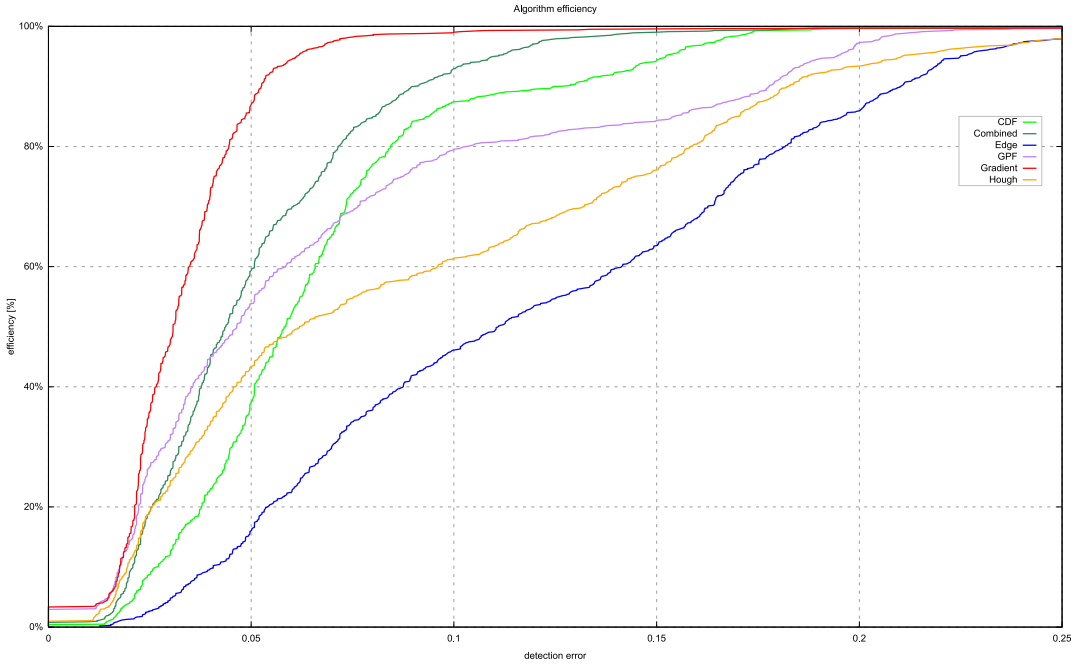

Fig. 10. Efficiency graph when both eyes are found in 1017 out of 1521 images.

The obtained relative errors of the algorithms were very good in general, with the Gradient algorithm getting an efficiency very close to 90% for a detection error lower than 0.05.

A one-way ANOVA was conducted to examine whether there were statistically significant differences among the relative error of the analysed algorithms. The results revealed statistically significant differences among them, $F(5, 6096) = 39.01$, $p < 0.001$. Post-hoc Games-Howell tests revealed statistically significant differences between most algorithms (CDF: $M = 0.07$, SD $= 0.15$; Combined: $M = 0.06$, SD $= 0.15$; Edge: $M = 0.13$, SD $= 0.16$; GPF: $M = 0.07$, SD $= 0.15$; Gradient: $M = 0.04$, SD $= 0.15$; Hough: $M = 0.09$, SD $= 0.16$), other than between CDF and Combined, CDF and GPF, Combined and GPF, and Combined and Gradient, where there were no statistically significant differences.

While this is a fine approach for face or eye detection, it can be misleading for pupil detection algorithms, as the results are dependant on the performance of the previous stages of the process (face and eye detection). The calculated efficiency of the pupil detection algorithm, after the eyes have been located, is highly dependent on the eye detection algorithms used before.

A strict eye detection pass will only return the more standard eyes; i.e. those that are big enough and are facing the camera. Those will be easier to process by the pupil detection algorithm. This can be seen in Fig. 10, where the eye detection algorithm only detected both eyes in the 1017 more standard eyes within the 1521 images of the BioID database.

A smarter eye detection pass will return additional eyes: either smaller or with less standard orientations and lighting, or with glasses. Those will be harder to process by the pupil detection algorithm. In the test corresponding to Fig. 11, the eye detection algorithm detected both eyes in 1375 images. The performance of the algorithms went considerably down compared to the previous test.

Finally, a much smarter eye detection algorithm will locate many more non-standard eyes, smaller sizes, different orientations, etc. In the test corresponding to Fig. 12, a multi-staged approach was taken in order to locate the eyes even if the face was not found. This extended detection can result in an increase of the number of false positives, especially in those cases when the face is not visible. This test is the hardest one for the pupil detection algorithms. The efficiency graph shows an apparent decrease of efficiency, due to an increased performance of the eye detection algorithm.

To support this comparison, a one-way ANOVA was conducted to examine whether there were statistically significant differences among the obtained results across all algorithms, given different face and eye tracking preconditions. The results revealed statistically significant differences among them, $F(2, 23102) = 215.12$, $p < 0.001$. Post-hoc Games-Howell tests revealed statistically significant differences between the three tracking preconditions (Fig. 10: $M = 0.08$, SD $= 0.16$; Fig. 11: $M = 0.08$, SD $= 0.07$; Fig. 12: $M = 0.13$, SD $= 0.25$).

For this reason, it is not valid to compare the efficiency of pupil detection algorithms implemented and executed using different eye detection algorithms or configurations. A given evaluation could be using a very strict eye detection algorithm, while another one may use a more lax eye detection algorithm, thus getting a very low score in the efficiency graphs.

One approach that would solve this problem would be to use exactly the same algorithm and parameters for the face and eye detection, for all the comparisons of pupil detection algorithms. This could be accomplished by using a shared library with a specific implementation of the face and eye detection algorithms.

Alternatively, another option that would give completely objective results would be the use of a different dataset that contained only images or videos of eyes, in order to fully isolate the performance of the pupil detection algorithms from the previous stages. The use of video databases could additionally better evaluate the performance of pupil detection algorithms that take advantage of the position of the eye in the last frame, as opposed to evaluations based on still images.

Finally, as indicated by Timm and Barth (2011), some published articles define the relative error in a non-standard way, by taking the minimum or average errors, thus apparently obtaining much better results than with the standard formula, which represents the worst case. It is just not possible to compare results if the formulas used are different.

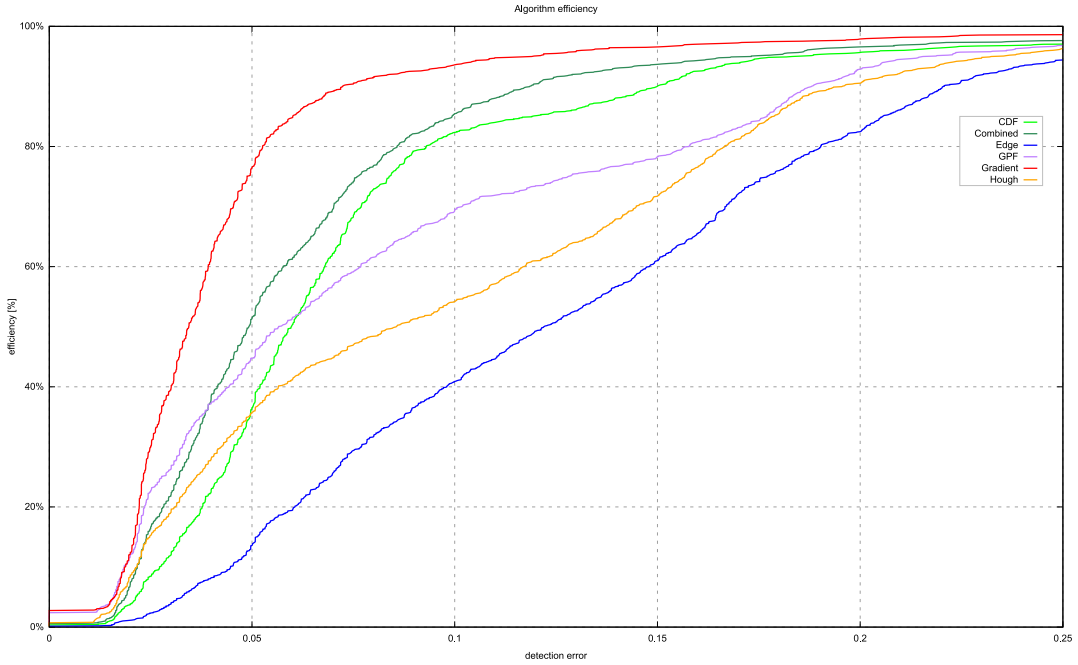

Fig. 11. Efficiency graph when both eyes are found in 1375 out of 1521 images.

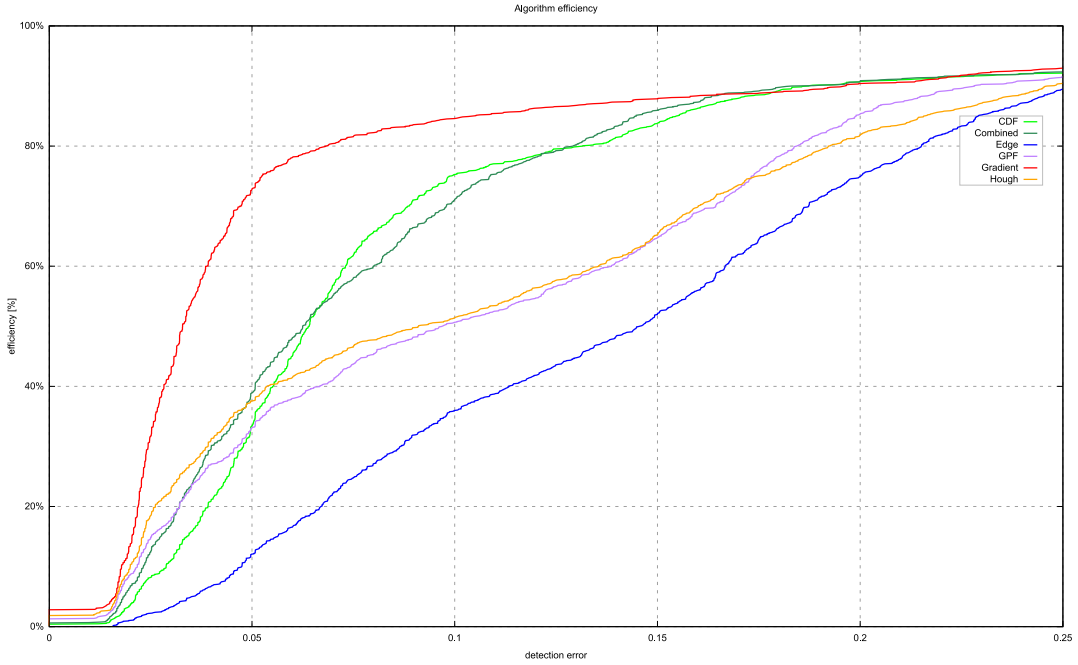

Fig. 12. Efficiency graph when both eyes are found in 1459 out of 1521 images.

While this section covers the spatial precision of the algorithms, eye movement analysis benefits from temporally stable results. The next section discusses how stable the aforementioned algorithms are.

5.2. Temporal stability

In the Static dataset, there is almost no variation in the position of the eyes between frames. The biggest source of variation is the random noise, either the one caused by the sensor of the camera, or the compression noise created when transferring the images from the camera to the computer. There is also a second compression noise created when compressing the captured videos, but this is considered lower, given that for this test the captured videos have been compressed using the highest quality allowed by the codec. Fig. 13 compares the temporal error per algorithm and device, using the Static dataset.

A series of one-way ANOVA analyses and an independent samples t-test were conducted to examine whether there were statistically significant differences among the temporal errors obtained when using different cameras, algorithms, resolution, framerates, and amount of light:

- When different cameras are used, the results reveal statistically significant differences $F(3, 188) = 4.10$, $p = 0.008$. Post-hoc Games-Howell tests reveal that these statistically signifi-

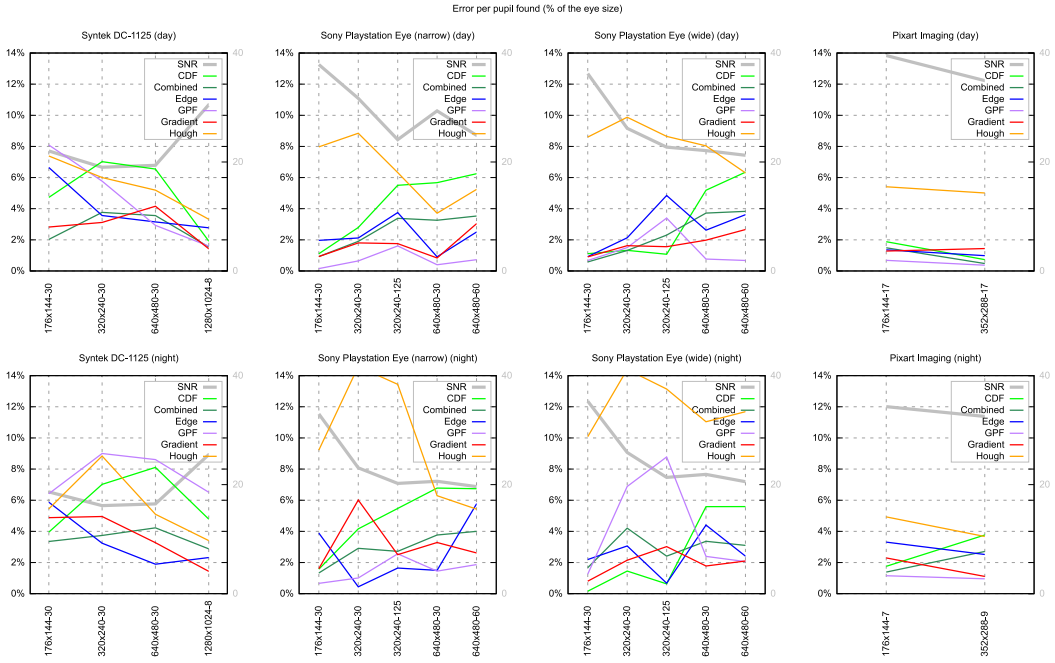

Fig. 13. Average Temporal Error per pupil found.

cant differences are only present between the Pixart Imaging camera and the rest (Pixart Imaging : $M = 2.11$, SD $= 2.88$; Sony Playstation Eye (narrow): $M = 3.58$, SD $= 2.97$; Sony Playstation Eye (wide): $M = 3.84$, SD $= 3.45$; Syntek DC-1125: $M = 4.55$, SD $= 2.13$). There are no other significant differences between the other groups.

- When different algorithms are used, the results reveal statistically significant differences $F(5, 186) = 26.41$, $p < 0.001$. Post-hoc Games-Howell tests reveal that these statistically significant differences are only present between the Hough and all other algorithms, and between the CDF and Gradient ones (CDF: $M = 3.96$, SD $= 2.40$; Combined: $M = 2.67$, SD $= 1.11$; Edge: $M = 2.78$, SD $= 1.53$; GPF: $M = 2.86$, SD $= 2.89$; Gradient: $M = 2.35$, SD $= 1.28$; Hough: $M = 7.71$, SD $= 3.25$). There are no other significant differences between the other groups.
- When different resolutions are used, the results reveal statistically significant differences $F(4, 187) = 3.13$, $p = 0.016$. However, post-hoc Games-Howell tests reveal that these statistically significant differences are only present between the resolutions 320 × 240 and 352 × 288, and between 640 × 480 and 352 × 288 (176 × 144: $M = 3.10$, SD $= 2.71$; 320 × 240: $M = 4.41$, SD $= 3.54$; 352 × 288: $M = 1.98$, SD $= 1.52$; 640 × 480: $M = 4.06$, SD $= 2.42$; 1280 × 1024: $M = 2.83$, SD $= 1.54$). There are no other significant differences between the other resolutions. This difference could be due to this resolution being a special case, as it is only produced by one of the cameras, which has a much smaller field of view and therefore the eyes are captured with much higher resolution.
- When different framerates are used, the results reveal statistically significant differences $F(4, 187) = 2.84$, $p = 0.026$. Post-hoc Games-Howell tests reveal that these statistically significant differences are present between the framerates 7/8/9 and 29/30, 17 and 29/30, 17 and 60 (7/8/9 fps: $M = 2.65$, SD $= 1.39$; 17 fps: $M = 1.76$, SD $= 1.68$; 29/30 fps: $M = 3.99$, SD $= 3.04$; 60 fps: $M = 4.09$, SD $= 2.43$; 125 fps: $M = 4.21$, SD $= 3.57$). There are no other significant differences between the other framerates.
- The t-test reveal a statistically significant difference between the day and night videos ($t = -2.51$, $df = 190$, $p = 0.013$). Videos recorded during the day ($M = 3.21$, SD $= 2.39$) report significantly lower temporal errors than those recorded at night ($M = 4.23$, SD$= 3.23$).

When the Signal to Noise Ratio is higher (such as in the videos recorded during the day), the noise present in the source is lower, and therefore the detection can be more accurate and stable in general. The size of the eye also has a positive contribution, as it means that the cropped eye image will have more detail and resolution.

From the graph in Fig. 13, it can be observed that there is no silver bullet. The GPF algorithm is the most stable one when the Signal to Noise Ratio and the size of the eye are higher (second and fourth columns), but not in other cases. This algorithm seems to be more sensitive to noise than others, and requires a more detailed source.

For more noisy sources (left column), the Combined, Edge and Gradient algorithms perform better than the rest. For a less noisy source, where the size of the detected eyes is smaller (third column) there is no clear winner, even though the Gradient algorithm seems to provide the most stable results.

In general these algorithms achieve more accurate and stable results in the following conditions:

- The user can be situated anywhere within the field of view of the camera, closer or further away from it, as long as they are looking toward the camera and at least one of their eyes is visible in the captured image. However, the closer the user is to the camera, the higher the resolution of the captured eye will be.
- Reflections produced by glasses may introduce errors which lower the accuracy, so even though it is not required, it is recommended for the users not to wear glasses if possible.
- Using higher-definition cameras, or cameras with a narrow field of view, will also help for eyes to be captured with higher resolution. However, do note that the use of high resolutions in cheap cameras will typically limit the number of frames per second that can be captured, which would be counterproductive for movement analysis.

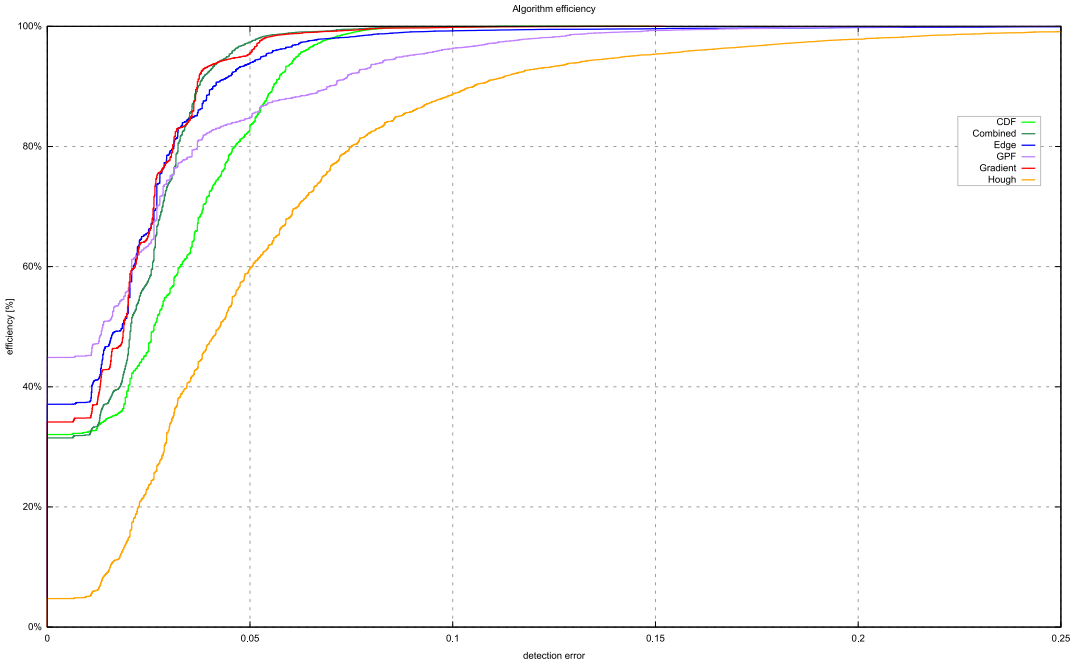

Fig. 14. Efficiency graph of the Temporal Stability in the Static dataset.

- The higher signal to noise ratio, the better. This is typically achieved with a better illuminated scene, or the use of cameras with bigger or more sensitive sensors. A low signal to noise ratio will often trigger noise reduction algorithms in the camera driver, which can create a ghosting effect in the images.
- Lowering the number of frames per second captured can lead to less noise as well. When using higher framerates the exposition time per frame is lower, so the sensor has to be set to a more sensitive mode to capture more light, which will produce a higher amount of noise.

It is worth to mention that the algorithms presented in this paper search for the iris, given its size and the high contrast with the sclera. Therefore, they are not affected by the size of the pupil, which can vary depending of factors such as the luminosity of the screen or the image being displayed to the user (Binda, Pereverzeva, & Murray, 2013), their age (Winn, Whitaker, Elliott, & Phillips, 1994), interest (Hess & Polt, 1977), and cognitive load (Kahneman & Beatty, 1966). In any case, given the low resolution of the eyes in the captured images, it would be very hard to impossible to distinguish the pupils in the images, especially in the frequent presence of reflections in the eyes.

Fig. 14 represents an efficiency graph of the above results combined, independently of the device or the resolution. There were statistically significant differences between algorithms, as determined by one-way ANOVA ($F(5, 75756) = 3136.86$, $p < 0.001$). Post-hoc Games-Howell tests revealed statistically significant differences between all algorithms (CDF: $M = 0.03$, SD $= 0.02$; Combined: $M = 0.02$, SD $= 0.02$; Edge: $M = 0.02$, SD $= 0.02$; GPF: $M = 0.02$, SD $= 0.03$; Gradient: $M = 0.02$, SD $= 0.02$; Hough: $M = 0.05$, SD $= 0.05$), except between Combined and Edge, where there were no statistically significant differences. The Gradient and Combined algorithms are less affected by random noise, so they can be considered more temporally stable than the rest.

For completion, the same procedure has been applied to the HPEG and Boston University datasets. These videos do contain some movement, so the noise is not the only factor that can affect the results. Therefore, graphs in Fig. 15 and Fig. 16 represent a combination of spatial and temporal efficiency.

Additional one-way ANOVA analyses were conducted to examine whether there were statistically significant differences among the temporal stability of the analysed algorithms in the HPEG and Boston datasets. The results for the HPEG dataset revealed statistically significant differences among the algorithms, $F(5, 30150) = 54.10$, $p < 0.001$. Post-hoc Games-Howell tests revealed statistically significant differences between all algorithms (CDF: $M = 0.04$, SD $= 0.10$; Combined: $M = 0.04$, SD $= 0.10$; Edge: $M = 0.06$, SD $= 0.10$; GPF: $M = 0.05$, SD $= 0.11$; Gradient: $M = 0.03$, SD $= 0.09$; Hough: $M = 0.06$, SD $= 0.12$), except between CDF and Combined, and between Edge and Hough, where there were no statistically significant differences.

The results for the Boston dataset also revealed statistically significant differences among the algorithms, $F(5, 80460) = 903.83$, $p < 0.001$. Post-hoc Games-Howell tests revealed statistically significant differences between all algorithms (CDF: $M = 0.07$, SD $=$ 0.07; Combined: $M = 0.07$, SD $= 0.07$; Edge: $M = 0.08$, SD $= 0.09$; GPF: $M = 0.08$, SD $= 0.09$; Gradient: $M = 0.07$, SD $= 0.08$; Hough: $M = 0.12$, SD $= 0.09$), except between CDF and Combined, and between CDF and Gradient, where there were no statistically significant differences.

The Gradient algorithm performs distinctly better than the rest in the HPEG dataset, but it is outperformed in the Boston dataset by the Combined, Edge and GPT algorithms. All in all, the Gradient algorithm seems to be the most stable algorithm overall, leading the results in two of three datasets. This, along the much better efficiency demonstrated by this algorithm (as per the results in Section 5.1), could indicate that it is more appropriate to real-life scenarios, where both movements and noise are present, than the rest of the pupil detection algorithms.

6. Conclusions

Eye tracking seeks to obtain information about users' eye and gaze data, which has an increasing number of applications. It is usually achieved using intrusive and expensive specialized hardware, but eye tracking devices are nowadays becoming cheaper and much less intrusive. Nevertheless, given their availability,

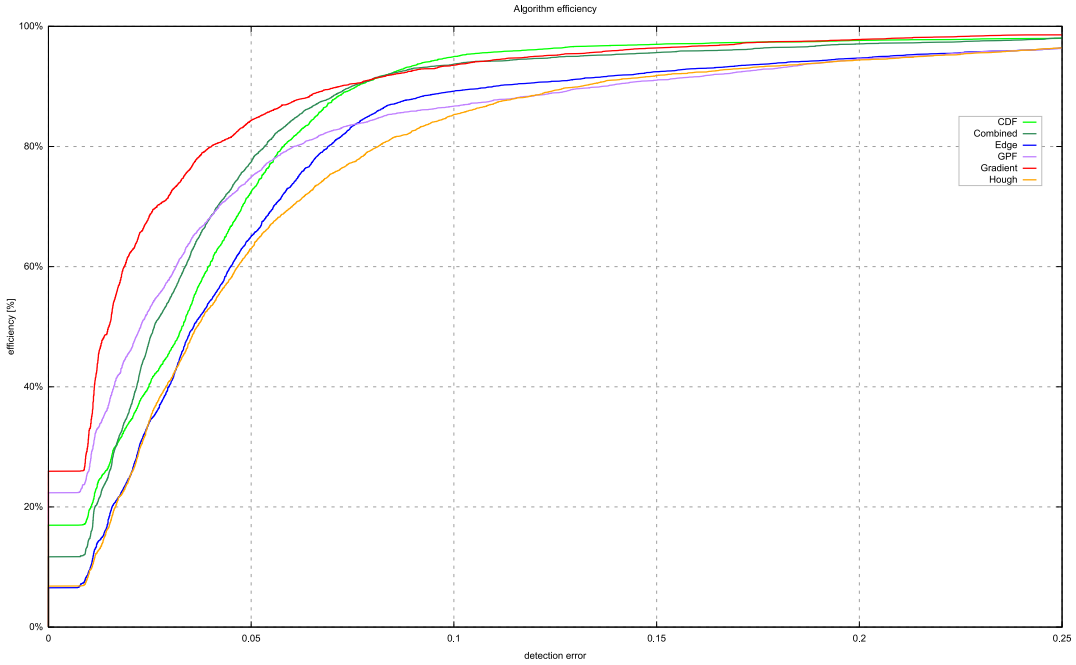

Fig. 15. Efficiency graph of the Temporal Stability in the HPEG dataset.

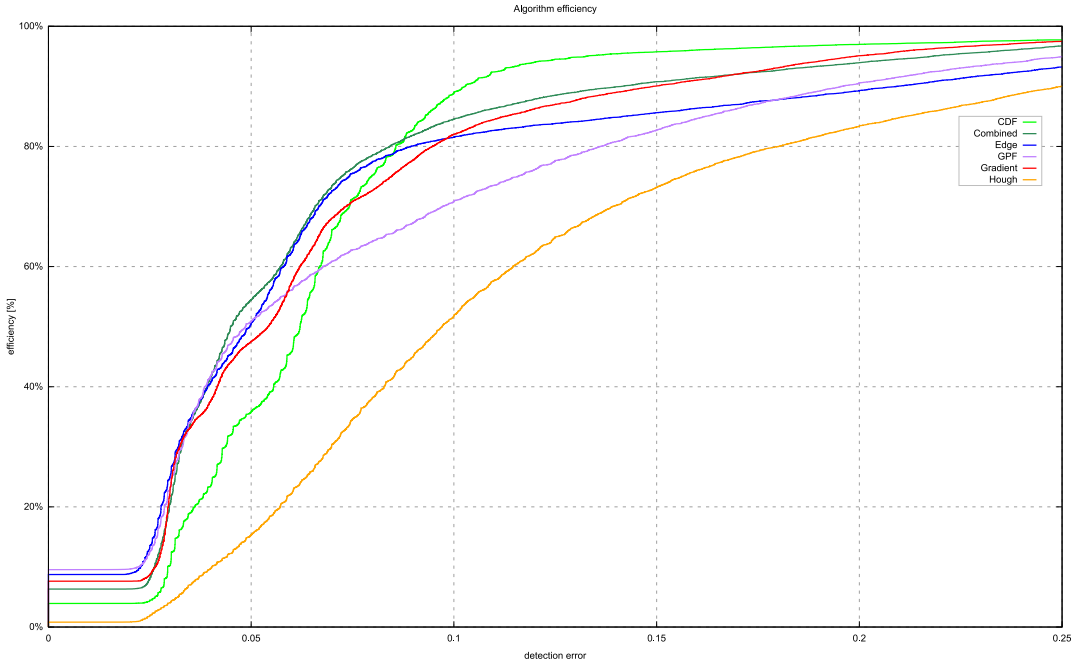

Fig. 16. Efficiency graph of the Temporal Stability in the Boston dataset.

unmodified webcams are the ideal devices for eye and gaze tracking in computers.

In this paper, it has been shown that besides the efficiency of pupil detection algorithms, a key goal in eye tracking systems using webcams is to achieve stable results. Therefore, a metric to measure the temporal stability of gaze tracking algorithms has been proposed. To evaluate this metric, a layered framework for pupil detection has been implemented, employing state-of-the-art algorithms. This metric can be easily applied in the future to other detection algorithms.

The created database of videos with minimum movement, using different devices and resolutions, has served as a tool for testing the temporal stability. The results obtained in Section 5 show which of the analysed algorithms have been found to be more precise and/or temporally stable than others, and which conditions have an impact on these results.

Additionally, the present work has discussed a pervasive issue in the analysis and comparison of the efficiency of pupil detection algorithms. The common representation of the relative error in efficiency graphs can be misleading for pupil detection algorithms, as the results are greatly dependant on the performance of the previous steps. The present work proposes two solutions that would allow objective and comparable performance analysis of pupil detection algorithms.

Since eye trackers based on infrared light are becoming more affordable and more widely used, one line of future work is to

compare the temporal stability of pupil detection algorithms from webcam streams with the temporal stability of other forms of eye tracking.

Our main line of future work is the application of the most stable tracking algorithms to the analysis, with the use of webcams, of eye movements in EVIN, a game-based system for visual training used to help visually impaired children (Matas, Santos, Hernández-del Olmo, & Gaudioso, 2015). In EVIN, children are supported by specialists in low vision, who must be aware of where the child is looking at while interacting with the game, to detect wrong interaction patterns. Thus, the eye movements of the child are going to be used to help specialists in low vision detect when the children have problems with the games.

References

Asadifard, M., & Shanbezadeh, J. (2010). Automatic adaptive center of pupil detection using face detection and CDF analysis. In S. I. Ao, O. Castillo, C. Douglas, D. D. Feng, & J. Lee (Eds.), *Proceedings of the international multiconference of engineers and computer scientists: I* (pp. 130–133). Newswood Limited.

Asteriadis, S., Nikolaidis, N., Hajdu, A., & Pitas, I. (2006). An eye detection algorithm using pixel to edge information. In I. Tabus, & R. Thami (Eds.), *Proceedings of the IEEE-EURASIP international symposium on control communication and signal proccessing*. Marrakech, Morocco: IEEE.

Asteriadis, S., Soufleros, D., Karpouzis, K., & Kollias, S. (2009). A natural head pose and eye gaze dataset. In C. Ginevra, M. Jean-Claude, M. John, K. Kostas, & P. Christopher (Eds.), *Proceedings of the international workshop on affective-aware virtual agents and social robots* (pp. 1–4). New York, NY, USA: ACM.

Barea, R., Boquete, L., Ortega, S., López, E., & Rodríguez-Ascariz, J. (2012). EOG-based eye movements codification for human computer interaction. *Expert Systems with Applications, 39*(3), 2677–2683.

Bengoechea, J., Cerrolaza, J., Villanueva, A., & Cabeza, R. (2014). Evaluation of accurate eye corner detection methods for gaze estimation. *Journal of Eye Movement Research, 7*(3), 1–8.

Bentivoglio, A., Bressman, S. B., Cassetta, E., Carretta, D., Tonali, P., & Albanese, A. (2011). Analysis of blink rate patterns in normal subjects. movement disorders. *Official Journal of the Movement Disorder Society, 12*(6), 1028–1034.

Binda, P., Pereverzeva, M., & Murray, S. (2013). Attention to bright surfaces enhances the pupillary light reflex. *The Journal of Neuroscience, 33*(5), 2199–2204.

Bixler, R., & D'Mello, S. (2015). Automatic gaze-based detection of mind wandering with metacognitive awareness. In F. Ricci, K. Bontcheva, O. Conlan, & S. Lawless (Eds.), *User modeling, adaptation and personalization*. In *Lecture Notes in Computer Science: 9146* (pp. 31–43). Springer International Publishing.

Brandreth, G. (1986). *Your vital statistics: The ultimate book about the average human being*. Citadel Press.

Canny, J. (1986). A computational approach to edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence, PAMI-8 (6*(6), 679–698.

Carvalho, N., Laurent, E., Noiret, N., Chopard, G., Haffen, E., Bennabi, D., & Vandel, P. (2015). Eye movement in unipolar and bipolar depression: A systematic review of the literature. *Frontiers in Psychology, 6*, 1–17.

Chamaret, C., & Le Meur, O. (2008). Attention-based video reframing: Validation using eye-tracking. In *Proceedings of the 19th international conference on pattern recognition* (pp. 1–4). Florida, USA: IEEE Computer Society.

Ciesla, M., & Koziol, P. (2011). EyeTracker (version 1.0).Retrieved from http://th-www.if.uj.edu.pl/zfs/ciesla/main/EyeTracker.html. Accessed: 2016-06-16.

Ciesla, M., & Koziol, P. (2012). Eye pupil location using webcam (arXiv e-print no. 1202.6517). Retrieved from http://arxiv.org/abs/1202.6517. Accessed: 2016-06-16.

De Moivre, A. (1738). *The doctrine of Chances*. Cambridge.

Deng, L., Hsu, C., Lin, T., Tuan, J., & Chang, S. (2010). EOG-based human-computer interface system development. *Expert Systems with Applications, 37*(4), 3337–3343.

Drusch, G., Bastien, J. M. C., & Dinet, J. (2011). From gaze plots to eye fixation patterns using a clustering method based on hausdorff distances. In M. Riveill (Ed.), *Proceedings of the 23th french speaking conference on human-computer interaction* (pp. 1–7). Sophia Antipolis, France: ACM.

Ellis, N., Hafeez, K., Martin, K. I., Chen, L., Boland, J., & Sagarra, N. (2014). An eye–tracking study of learned attention in second language acquisition. *Applied Psycholinguistics, 35*, 547–579.

George, A., & Routray (2016). Fast and accurate algorithm for eye localization for gaze tracking in low resolution images. *IET Computer Vision*. doi:10.1049/iet-cvi.2015.0316.

Gomez-Poveda, J. (2014). JGaze. Retrieved from http://jmgomezpoveda.github.io/JGaze/. Accessed: 2016-06-16.

Grother, P. (2013). IREX-VI temporal stability of iris recognition accuracy. NIST Interagency Report 7948. Retrieved from http://www.nist.gov/customcf/get_pdf.cfm?pub_id=913900. Accessed: 2016-06-16.

Han, S., Yang, S., Kim, J., & Gerla, M. (2012). EyeGuardian: A framework of eye tracking and blink detection for mobile device users. In *Proceedings of the 13th workshop on mobile computing systems and applications*. In *HotMobile '12* (pp. 6:1–6:6). ACM.

Hess, E., & Polt, J. (1977). Pupil size as related to interest value of visual stimuli. *Science, 132*(3423), 349–350.

Holland, C., & Komogortsev, O. (2012). Eye tracking on unmodified common tablets: Challenges and solutions. In *Proceedings of the symposium on eye tracking research and applications*. In *ETRA '12* (pp. 277–280). Santa Barbara, California: ACM.

Hui, T. (2000). *Noise Analysis in CMOS Image Sensors*. Department of Applied Physics. Standford University, USA PhD dissertation.

Hume, T. (2013). eyeLike. Retrieved from https://github.com/trishume/eyeLike. Accessed: 2016-06-16.

Ibrahim, B., & Raney, G. (2016). An eye-tracking study on the role of attention and its relationship with motivation. In S. Liszka, P. Leclercq, M. Tellier, & D. Véronique (Eds.), *EUROSLA Year Book* (pp. 1–35)). Amsterdam: John Benjamins.

Intel Corporation, Willow Garage, & Itseez (2011). Video input with OpenCV and similarity measurement.Retrieved from http://docs.opencv.org/doc/tutorials/highgui/video-input-psnr-ssim/video-input-psnr-ssim.html. Accessed: 2016-06-16.

Intel Corporation, Willow Garage, & Itseez (2015). OpenCV. opensource for computer vision. Retrieved from http://opencv.org/. Accessed: 2016-06-16.

Jafari, R., & Ziou, D. (2015). Eye-gaze estimation under various head positions and iris states. *Expert Systems with Applications, 42*(1), 510–518.

Jesorsky, O., Kirchberg, K., & Frischholz, R. (2001). Robust face detection using the Hausdorff distance. In J. Bigun, & F. Smeraldi (Eds.), *Proceedings of the third international conference on audio- and video-based biometric person authentication*. In *LNCS-2091* (pp. 90–95). Halmstad,Sweden: Springer Berlin Heidelberg.

Kahneman, D., & Beatty, J. (1966). Pupil diameter and load on memory. *Science, 154*(3756), 1583–1585.

Kenyon, R. (1985). A soft contact lens search coil for measuring eye movements. *Vision Research, 25*(11), 1629–1633.

Kunka, B., & Kostek, B. (2009). Non-intrusive infrared-free eye tracking method. In *Proceedings of 13th signal processing algorithms, architectures, arrangements, and applications, spa 2009* (pp. 105–109). IEEE.

La Cascia, M., Sclaroff, S., & Athitsos, V. (2000). Fast, reliable head tracking under varying illumination: An approach based on registration of texture- mapped 3D models. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 22*(4), 322–336.

Lai, M., Tsai, M., Yang, F., Hsu, C., Liu, T., Lee, S., … Tsai, C. (2013). A review of using eye-tracking technology in exploring learning from 2000 to 2012. *Educational Research Review, 10*, 90–115.

Li, D., Babcock, J., & Parkhurst, D. (2006). OpenEyes: a low-cost head-mounted eye–tracking solution. In *Proceedings of the 2006 symposium on eye tracking research and applications* (pp. 95–100). San Diego, CA, USA: ACM.

Liao, S., Zhu, X., Lei, Z., Zhang, L., & Li, S. (2007). Learning multi-scale block local binary patterns for face recognition. In L. Seong-Whan, & S. Z. L. (Eds.), *Advances in biometrics*. In *Lecture Notes in Computer Science: 4642* (pp. 828–837). Springer Berlin Heidelberg.

Lienhart, R., & Maydt, J. (2002). An extended set of Haar-like features for rapid object detection. In *Proceedings of international conference on image processing: 1* (pp. 900–903). IEEE.

Makris, S., Hadar, A., & Yarrow, K. (2013). Are object affordances fully automatic? a case of covert attention. *Behavioral Neuroscience, 127*(5), 797–802.

Manor, B., & Gordon, E. (2003). Defining the temporal threshold for ocular fixation in free-viewing visuocognitive tasks. *Journal of Neuroscience Methods, 128*(1–2), 85–93.

Matas, Y., Santos, C., Hernández-del Olmo, F., & Gaudioso, E. (2015). Towards web-based visual training. *International Journal of Development and Educational Psychology, 1*(2), 55–60.

Mbouna, R. O., & Kong, S. G. (2012). Pupil center detection with a single webcam for gaze tracking. *Journal of Measurement Science and Instrumentation, 3*(2), 133–136.

myGaze (2014). Visual interaction, myGaze eye tracker. Retrieved from http://www.mygaze.com/products/mygaze-eye-tracker/. Accessed: 2016-06-16.

Nielsen, J., & Pernice, K. (2009). *Eyetracking web usability*. New Riders.

Ooms, K., Andrienko, G., Andrienko, N., De Maeyer, P., & Fack, V. (2012). Analysing the spatial dimension of eye movement data using a visual analytic approach. *Expert Systems with Applications, 39*(1), 1324–1332.

Papageorgiou, C. P., Oren, M., & Poggio, T. (1998). A general framework for object detection. In *Proceedings of sixth international conference on computer vision* (pp. 555–562). Bombay, India: IEEE.

Peterson, J., Pardos, Z., Rau, M., Swigart, A., Gerber, C., & McKinsey, J. (2015). Understanding student success in chemistry using gaze tracking and pupillometry. In C. Conati, N. Heffernan, A. Mitrovic, & M. F. Verdejo (Eds.), *Artificial intelligence in education*. In *Lecture Notes in Computer Science: 9112* (pp. 358–366). Springer International Publishing.

Porta, M., & Ravarelli, A. (2012). Some eye tracking solutions for severe motor disabilities. In Z. S. Hippe, J. L. Kulikowski, & T. Mroczek (Eds.), *Human – computer systems interaction: Backgrounds and applications 2: Part 1* (pp. 417–431). Springer Berlin Heidelberg.

Quiros, P., & Yee, R. (2014). Nyastagmus, saccadic intrusions, and oscillations. In M. Yanoff, & J. Duker (Eds.), *Ophthalmology. 4th ed.* (pp. 9–19). Philadelphia: Elsevier Mosby.

Richardson, D. C., & Spivey, M. J. (2016). Eye-tracking: Research areas and applications. In G. E. Wnek, & G. L. Bowlin (Eds.), *Encyclopedia of Biomaterials and Biomedical Engineering* (pp. 572–573). Dekker.

San Agustin, J., Skovsgaard, H., Hansen, J. P., & Hansen, D. (2009). Low-cost gaze interaction: Ready to deliver the promises. In *Proceedings of chi '09, extended abstracts on human factors in computing systems*. In *CHI EA '09* (pp. 4453–4458). San Jose, CA, USA: ACM.

San Agustin, J., Skovsgaard, H., Mollenbach, E., Barret, M., Tall, M., Hansen, D., & J. P. , H. (2010). Evaluation of a low-cost open-source gaze tracker. In *Proceedings of the 2010 symposium on eye tracking research and applications etra'10* (pp. 77–80). Austin,TX, USA: ACM.

Sesma, L., Villanueva, A., & Cabeza, R. (2012). Evaluation of pupil center-eye corner vector for gaze estimation using a web cam. In *Proceedings of the symposium on eye tracking research and applications*. In *ETRA '12* (pp. 217–220). Santa Barbara, CA, USA: ACM.

Sewell, W., & Komogortsev, O. (2010). Real-time eye gaze tracking with an unmodified commodity webcam employing a neural network. In *Chi '10 extended abstracts on human factors in computing systems*. In *CHI EA '10* (pp. 3739–3744). Atlanta, GA, USA: ACM.

Timm, F., & Barth, E. (2011). Accurate eye centre localisation by means of gradients. In *Proceedings of the sixth international conference on computer vision theory and applications, visapp 2011* (pp. 125–130). Vilamoura, Algarve: SciTePress.

Tobii (2016). Access your computer through eye gaze. Retrieved from http://www.tobiidynavox.com/pceye-mini/. Accessed: 2016-06-16.

Tsai, M., Hou, H., Lai, M., Liu, W., & Yang, F. (2012). Visual attention for solving multiple-choice science problem: An eye-tracking analysis. *Computers and Education, 58*(1), 375–385.

Viola, P., & Jones, M. (2001). Rapid object detection using a boosted cascade of simple features. In *Proceedings of the 2001 ieee conference on computer vision and pattern recognition. cvpr: 1* (pp. 511–518). Colorado Springs, USA: IEEE.

Vrzakova, H., & Bednarik, R. (2015). Quiet eye affects action detection from gaze more than context length. In F. Ricci, K. Bontcheva, O. Conlan, & S. Lawless (Eds.), *User modeling, adaptation and personalization*. In *Lecture Notes in Computer Science: 9146* (pp. 277–288). Springer International Publishing.

Wang, J. T. (2011). Pupil dilation and eye tracking. In M. Schulte-Mecklenbeck, A. Kuehberger, & R. Ranyard (Eds.), *A handbook of process tracing methods for decision research: A critical review and user's guide* (pp. 185–204). Pschology Press.

Weerasinghe, A., Elmadani, M., & Mitrovic, A. (2015). Using eye gaze data to explore student interactions with tutorial dialogues in a substep-based tutor. In C. Conati, N. Heffernan, A. Mitrovic, & M. F. Verdejo (Eds.), *Artificial intelligence in education*. In *Lecture Notes in Computer Science: 9112* (pp. 812–815). Springer International Publishing.

Williams, T., & Kelley, C. (2012). Gnuplot (version 4.6 patchlevel 1). Retrieved from http://www.gnuplot.info. Accessed: 2016-06-16.

Winn, B., Whitaker, D., Elliott, D., & Phillips, N. (1994). Factors affecting light-adapted pupil size in normal human subjects. *Investigative ophthalmology & visual science, 35*(3), 1132–1137.

Yuen, H., Princen, J., Illingworth, J., & Kittler, J. (1990). Comparative study of Hough transform methods for circle finding. *Image and vision computing, 8*(1), 71–77.

Zhengting, H. (2007). *Video compression and data flow for video surveillance* (pp. 11–17). Texas Instruments. White paper

Zhou, Z.-H., & Geng, X. (2004). Projection functions for eye detection. *Pattern Recognition, 37*(5), 1049–1056.

Zhu, J., Han, X., Ma, R., Li, X., Cao, T., Sun, S., & Hu, B. (2016). Exploring user mobile shopping activities based on characteristic of eye-tracking. In Q. Zu, & B. Hu (Eds.), *Human centered computing: Second international conference, HCC 2016, Colombo, Sri Lanka, January 7–9, 2016, Revised Selected Papers* (pp. 556–566)). Springer International Publishing.