Hindawi
Journal of Sensors
Volume 2019, Article ID 3931713, 19 pages
https://doi.org/10.1155/2019/3931713

Research Article

Optical Mouse Sensor for Eye Blink Detection and Pupil Tracking: Application in a Low-Cost Eye-Controlled Pointing Device

Marcel Tresanchez, Tomàs Pallejà, and Jordi Palacín

Department of Computer Science and Industrial Engineering, University of Lleida, 25001 Lleida, Spain

Correspondence should be addressed to Marcel Tresanchez; mtresanchez@diei.udl.cat

Received 31 July 2019; Revised 4 October 2019; Accepted 1 November 2019; Published 10 December 2019

Academic Editor: Carlos Ruiz

In this paper, a new application of the optical mouse sensor is presented. The optical mouse is used as a main low-cost infrared vision system of a new proposal of a head-mounted human-computer interaction (HCI) device controlled by eye movements. The default optical mouse sensor lens and illumination source are replaced in order to improve its field of view and capture entire eye images. A complementary 8-bit microcontroller is used to acquire and process these images with two optimized algorithms to detect forced eye blinks and pupil displacements which are translated to computer pointer actions. This proposal introduces an inexpensive and approachable plug and play (PnP) device for people with severe disability in the upper extremities, neck, and head. The presented pointing device performs standard computer mouse actions with no extra software required. It uses the human interface device (HID) standard class of the universal serial bus (USB) increasing its compatibility for most computer platforms. This new device approach is aimed at improving comfortability and portability of the current commercial devices with simple installation and calibration. Several performance tests were done with different volunteer users obtaining an average pupil detection error of 0.34 pixels with a successful detection in 82.6% of all mouse events requested by means of pupil tracking.

1. Introduction

Nowadays, the most widely used human-computer interaction is a graphic pointer that is displaced across the screen of a display peripheral. The screen pointer is usually controlled using standard devices such as computer mice, touchpads, joysticks, pens, or tactile screen panels. All of these require some physical interaction moving some extremities of the user's body like fingers, hands, or forearms to operate properly. Several alternatives have been proposed to allow people with mobility impairments in the upper extremities to control the computer pointer. These are mainly based on the detection and measurement of such remaining body motions as facial gestures [1, 2], mouth movement [3, 4], head movements [5–8], eye tracking [9, 10], sticker tracking [11, 12], breath [13, 14], tongue displacements [15], or a combination of them [16]. Nevertheless, there are people with such severe disability that they cannot move any extremity, neck, or head and are only able to interact with computer devices using their eyes, eyebrows, tongue, mouth, breath, or brain activity [17–19]. Commercial devices to achieve the inclusion of these people to the communications technology (ICTs) are available, although most of them have several drawbacks related to software compatibility, limitation in computer interaction, and complex configuration and calibration; or are highly intrusive and not affordable due to their complexity and low prospective market.

Focusing on the eye-controlled devices, they can be classified into two main groups: remote and head-mounted. The remote devices are mainly based on the application of different eye gaze algorithms to the image acquired by a fixed high-resolution camera, usually attached to the computer screen [2, 5, 6, 9, 20]. Examples of remote pupil gaze devices on the market are [21–24] which are expensive (> $6,000) due to the high-resolution camera, the lenses, and the dedicated illumination system cost. The head-mounted devices are mostly based on a custom structure holding a small intrusive low-resolution camera in front of the user's face [10, 25–27].

They are less expensive but not low-cost [28–30] (> $2,000) because of the camera's quality and the customised software.

Several head-mounted eye-tracking devices have been proposed in the literature. In [25], three cameras and an external computer are used to simultaneously track the eye and the relative orientation of the head-mounted structure. It uses a very complex hardware to detect the pupil by a simple gray level segmentation of infrared images. In [26], two mini cameras and two medium-sized processing boards are used to simultaneously track the pupil and the user's view, allowing an absolute pointer control on the computer screen. Although it proposes to use low-cost off-the-shelf components and a set of open-source software, the installation is complicated because the multiple parts. Recently, Cáceres et al. [10] proposed to use a head-mounted commercial eye tracker along with a modified webcam to develop an inexpensive eye pointer, obtaining a successful user evaluation. However, it requires a precise installation of infrared lights at the corners of the screen. A new low-cost alternative of eye gaze head-mounted device with a near-eye display is reported in [27]. This proposal can reach an eye gaze accuracy of 0.53° but needs a near-eye display degrading the quality of the user's view and reducing its field.

In this work, a new proposal of an inexpensive human-computer eye-controlled pointing device based on a low-cost optical mouse sensor is presented (Figure 1). The main goal of this contribution is the proposed device cost (~$20), compared with the most affordable current commercial eye-tracking devices—e.g., Pupil Labs Pupil Core, $1,850 [31]; Tobii PCEye Mini, $1,149 [32]; or Tobii Pro Glasses II, $10,000 [33]. This is possible because the new proposal uses the same components of a common commercial optical mouse. Likewise, it avoids high-resolution cameras and the need of additional hardware parts, like [21–30]. It is a lightweight (41 g) wearable plug and play (PnP) device with no additional software required and fully compatible (Windows, MacOS, Linux, Android, etc.), using the human interface device (HID) standard class of the universal serial bus (USB).

The optical mouse sensors, which were originally designed as the computer's mouse main sensor, are widely used in many scenarios to estimate the relative displacement of the surface under the sensor [34–38]. In [38], it was used to estimate the translation of the eye analyzing the scleral surface. A horizontal and vertical angular resolution of 27.8 and 18.2 counts per degree were obtained; however, the eye blinks preclude its usage as an eye gaze. Furthermore, several research works proposed the use of the optical sensor open access internal image to develop alternative inexpensive applications such as counterfeit coin detectors [39], oxygen and pH quantifiers [40], yarn diameter meters [41], and rotary encoders [42, 43]. Our previous work [44] analyzes the capabilities of these sensors for eye tracking, detecting the pupil as the darkest part of the infrared images acquired. A custom valley detection method and the well-known integrodifferential operator proposed by Daugman [45, 46] were tested to detect the pupil. Results showed that both algorithms allow to track the pupil with an average error of 0.58 pixels; however, the valley detection method requires 90% less memory and is faster since it does not use trigonometric functions.

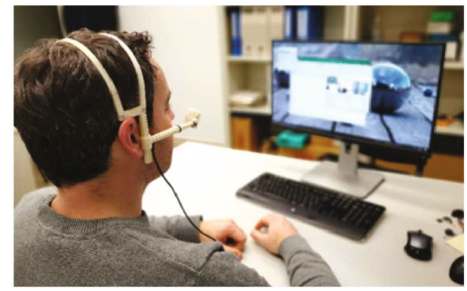


Figure 1: Low-cost human-computer eye-controlled interface device developed.

Based on the successful results from our previous work [44], this work proposes the use of the optical mouse sensor as an inexpensive imaging system to detect eye blinks and movements and validates its responsivity to perform basic pointer actions properly. To do so, a basic 8-bit microcontroller is used to capture and translate pupil displacements and eye blinks to common computer mouse actions (pointer displacements, clicks, and double clicks). The custom pupil detection method proposed in our previous work has been enhanced to increase its robustness. To do so, new restrictions were defined and adjusted to their optimal values for the proposed setup. Additionally, a procedure to detect forced eye blinks has been implemented and used as complementary interaction input.

2. Sensor

2.1. Optical Mouse Sensor. The optical flow sensor used in this work is the ADNS-3080 from Avago Technologies [47]. This sensor is known by its use as an optical mouse displacement sensor and it has many interesting features for our approach. It is a low-cost compact sensor based on a metal-oxide-semiconductor (CMOS) matrix of 30×30 grayscale pixels of 4-bit intensity which is highly sensitive to near-infrared wavelength, regularly used for pupil location and iris recognition [48, 49]. The CMOS is complemented with a digital signal processor (DSP), integrated on the same chip (Figure 2). Although it implements a proprietary optical flow algorithm for displacement measurements, it includes an extra feature called PIXEL_GRAB which allows access to the current surface image (frame). It is frozen and can be read at any asynchronous rate pixel by pixel through a standard serial peripheral interface (SPI) bus. Hence, the image acquisition by a serial peripheral interface (SPI), the storage of a very small array of 30×30 pixels (900 bytes) and its processing of pupil tracking can be implemented in a low-performance microcontroller to keep the device as inexpensive as possible.

The optical sensor was originally developed to acquire small variations in the roughness of the surface in a very short focal distance and reduced area. In general, the sensor package is combined with a polycarbonate plastic convex lens system (Figure 3(a)) to provide an adequate infrared illumination and a field of view for proper operation. Using the default lens and the recommended working

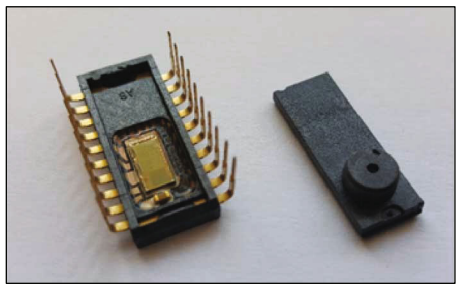
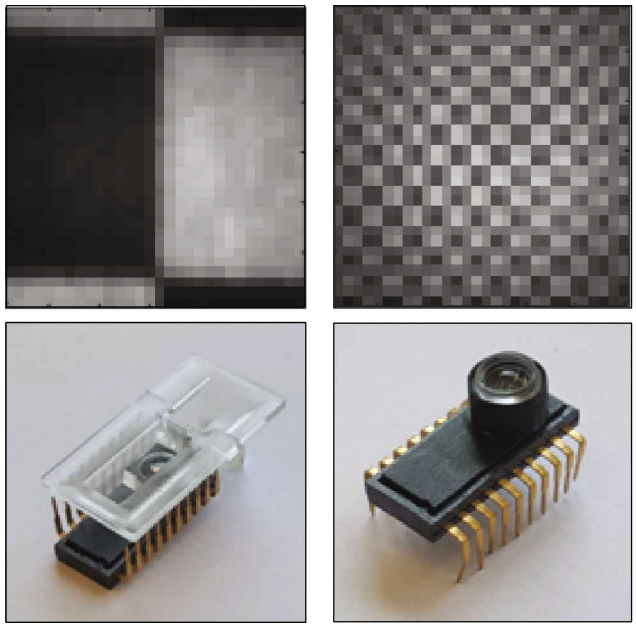


Figure 2: Internal view of the ADNS-3080 optical mouse sensor.



(a) (b)

Figure 3: Images of a 1.5 mm^2 chess template acquired by the sensor. (a) Using the default set up. (b) Using the CAY46 lens along with the SFH-4350 LED.

height defined by the manufacturer as 2.4 mm, the sensor can measure two-dimensional displacements with resolutions up to 800 counts per inch (cpi). Using this configuration, the working area captured by the sensor is very reduced (1.82 mm^2) [43].

2.2. Lens and Light Source. To achieve the goal of this work, both the default optics and the illumination source have been replaced in order to increase the capturing area and working distance. The lens used is the CAY46 low-cost plastic aspheric lens from Laser Components [50] which has a suitable focal distance of 4.6 mm. The use of this lens allows to capture essentially all pupil movements at working distance between 40 and 100 mm, obtaining a pupil diameter between 5 and 15 pixels [44]; therefore, it can be suitable in a head-mounted device as planned.

The external light source has been replaced by the higher radiant intensity SFH-4350 near-infrared (NIR) light-emitting diode (LED) from OSRAM [51] in order to obtain a uniform light in the field of view and highlight the pupil as the darkest part of the image. It has a 3 mm (T1) transparent plastic package, a wavelength peak of 850 nm, and 26° of view angle. Its typical operation generates a radiant flux of 50 mW at a forward current of 100 mA. According to the EN 62471:2008 [52] European Standard, the irradiance to the eye surface obtained is 7.76 W·m^{-2} and the radiation is 899.13 kW·m^{-2}·sr^{-1}, which are less than the limits of the retinal thermal hazard (L_R) and the eye radiation hazard (E_{IR}), respectively. Therefore, the proposed infrared emitter setup should be safe.

Figure 3 shows the sensor with two different lenses: the default lenses kit (ADNS-2120-001 [53] from Avago Technologies) and the CA46 lens. A chess template of 1.5 mm^2 squares was captured using both configurations. In the first case (Figure 3(a)), it was made using the default light source and the default working height of 2.4 mm. In the modified setup (Figure 3(b)), the light source came from the SHF 4350 LED using an optimal focusing distance between the sensor and the template of 60 mm. Thus, the capturing area of the sensor is increased from 1.82 mm^2 to 21.6 mm^2 which is considered suitable to capture pupil displacements when staring at the computer screen.

3. Pupil Tracking

There are several well-known pupil detection algorithms in the literature. One of the most widely used during the years is the integrodifferential operator proposed by Daugman [45, 46] which is extensively applied for pupil localization in iris recognition applications [48, 49]. It assumes that the pupil has circular contours and operate as a circular edge detector. Another well-known method to locate the pupil and the iris was introduced by Wildes [54], which is based on searching ellipses into an edge-filtered image using the Hough transform [55–57]. Moreover, there are traditional robust pupil detection methods that combine different image processing techniques as edge detectors, morphologic operations, contour extraction, thresholding fitting, limbus ellipse fitting, etc. Some of the most important are the Starburst [58], Swirski [59], Pupil Labs [60], SET [61], ExCuSe [62], and ElSe [63]. A deep discussion of these methods for head-mounted eye trackers can be found in [64]. These robust algorithms are not suitable in our proposed device due to its limited resources. On the one hand, the acquisition sensor limitation with very low image resolution (30 × 30 pixels) and low grayscale intensity (6 bits). Most of the morphological search algorithms require better capture. On the other hand, the DSP limitations include very small RAM size (3.7 Kbytes), low processor performance (48 MHz), and slow image acquisition (71.2 ms). The execution of floating point trigonometrical and morphological operations is not feasible to maintain an acceptable frame rate for our application. For instance, the most recent robust pupil detection method, PuReST [65], takes an average execution time of 5.56 ms using an Intel® Core™ i5-4590 CPU @ 3.30 GHz. Only in terms of processor frequency, reducing the frequency to ours (48 MHz), the execution time was proportionally increased to 382.25 ms (2.6 Hz) which is already not suitable for fast pupil gaze tracking.

To detect the pupil, this work proposes an adaption of the custom-made optimized pupil detection algorithm introduced in our previous work [44], having a similar detection error than the integrodifferential method but improving the memory size and execution time. Considering that the pupil is the darkest part of the image, the algorithm uses a simple valley location to detect the pupil centroid. Therefore, it can analyze row by row independently without any extra auxiliary image buffer or a nested pixel level search. The algorithm is based on three image processing steps: (1) specular highlight removal, where the image is filtered to smooth the reflections of the NIR LED used; 2) intensity valleys location, where for each row a possible valley of pupil is found with specific constraints; and 3) pupil centroid estimation, where valleys are used to determine whether the center of the pupil exists in the image. The following subsections present the new changes from the previous algorithm and detail the three main sequential steps used in this proposed system. The last subsection, describes the optimal algorithm hyperparameter values used in this work.

3.1. Algorithm Upgrade. The algorithm presented in [44] has been enhanced in the last two steps in order to increase its robustness. In the case of the valley detection step, the previous approach detects the valley limits (valley boundaries) when a difference between consecutive pixels decrease/increase more than T_F/T_R left and right thresholds, respectively, and the valley length is more than T_D. In this new approach, the limits of the valleys are obtained by checking the valley height that must be comprised between ε_{VMIN} and ε_{VALLEY}, and the valley length that now must be comprised into the range of ε_{PMIN} and ε_{PMAX}. In this approach, the intensity of the pixel's valley always must be equal or growing regarding the neighbor in order to avoid internal intensity peaks. Furthermore, three extra conditions have been defined: ε_{VDIFF}, ε_{MDIFF}, and MNP. The ε_{VDIFF} is the minimum intensity difference between two consecutive pixels of the valley. This condition guarantee that the valley has an abrupt rising. The MNP is the minimum number of pixels inside the valley that the difference with the absolute minimum reach the ε_{VMIN}. This condition is aimed at helping the valleys' rejection with poor concavity and guarantee that most of the valley pixels belongs to the pupil intensity value.

In the case of the pupil centroid estimation step, instead of selecting the valleys that have the limits near to the maximum of a window accumulation of 3 [44], the proposed method selects the valleys that contain the maximum number of consecutive valleys where the difference of consecutive limit points (valley boundaries), both left and right, are less than a threshold, ε_{BDIFF}, and the maximum difference between all of them are less than the threshold ε_{BDISP}. The ε_{BDIFF} condition permits to detect a continuous smooth circular pupil edge discarding horizontal valley limit peaks or vertical discontinuities. The ε_{BDISP} condition ensures a valid distortion of the pupil due to its spherical displacement. Additionally, in this new approach, the number of valleys selected has to be between the ε_{PMIN} and ε_{PMAX} (vertical pupil length). Once group valleys are selected, the calculation of the pupil centroid is done in the same way as the previous approach, averaging the coordinates of the limits of the selected valleys. In case that more than one group of valleys satisfies the conditions, the group with the darkest averaged value is chosen.

3.2. Specular Highlight Removal. In the acquired images, the eye's conjunctiva mucous membrane produces light reflections from the NIR LED. In case the reflections are located into the pupil area, the detection of the pupil center could be inaccurate. To smooth the highlighted pixels in the current image I^F (Equation (1)) a filter is performed.

$$I^F = \begin{pmatrix} p_{1,1} & \cdots & p_{1,30} \\ \vdots & \ddots & \vdots \\ p_{30,1} & \cdots & p_{30,30} \end{pmatrix} = \left(p_{i,j}\right) \in Z^{30\times 30}[0, 63]. \quad (1)$$

First, an intensity threshold is defined, ε_{MAX} (Equation (2)), to discard pixels with values less than 80% of the maximum.

$$\varepsilon_{MAX} = 0.8 \cdot \max_{\substack{1\le i\le 30 \\ 1\le j\le 30}} p_{i.j.} \quad (2)$$

Then, the following spatial median filter is applied starting at the pixel $i = j = 2$ as shown in Equation (3).

$$\text{if } p_{i,j} \ge \varepsilon_{MAX}, p_{i,j} = \operatorname*{mean}_{j-1\le k\le j+1} p_{i-1,k},\ i = 2, \cdots, 30,\ j = 2, \cdots, 29. \quad (3)$$

Figure 4 shows an example of the filter results' two common cases: (a) when the reflected light is outside the pupil area and (b) when the reflected light is in the pupil edge. In the second case, the result is not as smooth as expected due to the filtered pixels average pupil and iris area.

3.3. Intensity Valleys Location. For each row, i, an intensity valley search is performed to find the best valley which could be part of the pupil. First, for each row, the positions of the local minimum intensity pixels are located by searching the intensity local peaks (Equation (4)).

$$\lambda_i = \left\{ j | p_{i,j-1} \ge p_{i,j} < p_{i,j+1} \text{ or } p_{i,j-1} > p_{i,j} \le p_{i,j+1} \right\} = \left\{ \lambda_{i_k} \right\}, \begin{matrix} i = 1, \cdots, 30 \\ j = 2, \cdots, 29 \end{matrix}. \quad (4)$$

λ_i contains the column positions in row i where it starts a right or left pixel intensity slope. Then, the column position of the absolute minimum is calculated using Equation (5).

$$\lambda_i^{MIN} = \lambda_{ik} \text{ where } p_{i,\lambda_{ik}} = \min_{j\in\lambda_i} p_{i,j},\ i = 1, \cdots, 30. \quad (5)$$

Starting from the absolute minimum pixel, λ_i^{MIN}, the next step is to locate the left (λ_i^L) and right (λ_i^R) limits (valley boundaries), where a possible valley is comprised of, using the conditions defined in Equations (6) and (7), respectively.

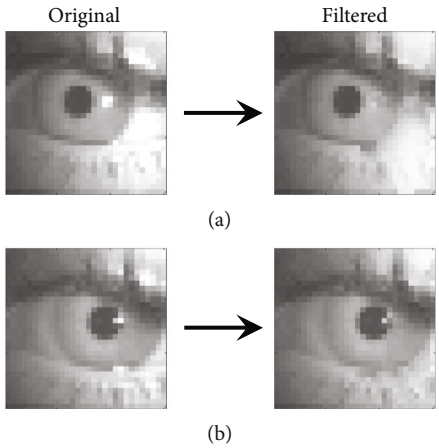


Figure 4: Examples of specular highlight removal using the proposed filter: (a) reflected light outside the pupil area and (b) reflected light in the pupil edge.

The λ_i^L and λ_i^R limits must be the furthest pixel from the λ_i^{MIN} which satisfies

(a) The pixel intensity always increases from the minimum (λ_i^{MIN}) to the limits (λ_i^{L}, λ_i^{R})

(b) One of these increments must be greater than $\varepsilon_{\text{VDIFF}}$

(c) The valley height (both left and right) must be between $\varepsilon_{\text{VMIN}}$ and $\varepsilon_{\text{VALLEY}}$

$$\lambda_i^L = \max_{1<k<\lambda_i^{\text{MIN}}} k, \text{where } \varepsilon_{\text{VMIN}} < \sum_{j=k}^{\lambda_i^{\text{MIN}}} \left(p_{i,j} - p_{i,j+1}\right) \leq \varepsilon_{\text{VALLEY}} \text{ and } p_{i,j} \geq p_{i,j+1} \text{ and } \max_{k\leq j\leq \lambda_i^{\text{MIN}}} \left(p_{i,j} - p_{i,j+1}\right) > \varepsilon_{\text{VDIFF}}. \tag{6}$$

$$\lambda_i^R = \max_{\lambda_i^{\text{MIN}}<k<30} k, \text{where } \varepsilon_{\text{VMIN}} < \sum_{j=\lambda_i^{\text{MIN}}}^{k} \left(p_{i,j+1} - p_{i,j}\right) \leq \varepsilon_{\text{VALLEY}} \text{ and } p_{i,j} \leq p_{i,j+1} \text{ and } \max_{\lambda_i^{\text{MIN}}\leq j\leq k} \left(p_{i,j+1} - p_{i,j}\right) > \varepsilon_{\text{VDIFF}}. \tag{7}$$

Then, to check that the valley belongs to the pupil region, it requires to have at least MNP pixels into its region (between λ_i^{L} and λ_i^{R}) which the intensity, regarding the minimum (λ_i^{MIN}), is below $\varepsilon_{\text{MDIFF}}$ (Equation (8)).

$$O_i = \left\{ j \mid \left| p_{i,\lambda_i^{\text{MIN}}} - p_{i,j} \right| < \varepsilon_{\text{MDIFF}} \right\}, \lambda_i^{\text{L}} < j < \lambda_i^{\text{R}} \; |O_i| \geq \text{MNP}. \tag{8}$$

Finally, the valley length must be comprised within a valid pupil range diameter between $\varepsilon_{\text{PMIN}}$ and $\varepsilon_{\text{PMAX}}$ thresholds (Equation (9)).

$$\varepsilon_{\text{PMIN}} \leq \lambda_i^{\text{R}} - \lambda_i^{\text{L}} \leq \varepsilon_{\text{PMAX}} \tag{9}$$

In the case that there is no valley found with the proposed restrictions, row i will be automatically rejected as a pupil portion. Figure 5 shows an example result of an intensity valley location for the 15th row of the image shown in Figure 6. A valley is found from column 12 to 19. The threshold values used in this work are $\varepsilon_{\text{VMIN}} = 2$, $\varepsilon_{\text{VALLEY}} = 5$, $\varepsilon_{\text{VDIFF}} = 2$, $\varepsilon_{\text{MDIFF}} = 5$, MNP = 3, $\varepsilon_{\text{PMIN}} = 3$, $\varepsilon_{\text{PMAX}} = 11$. These values are calculated according to the results obtained in Table 1.

3.4. Pupil Centroid Estimation. Once the valley limits are detected, it is necessary to find pupil region candidates. For each row $i = i_{\text{UP}}$ (with a valley found), the algorithm search a possible pupil region finishing at the lower row (i_{DOWN}) which satisfies the restrictions of Equation (10). The region has to be comprised with a valid set of adjacent valleys (consecutive rows in the image) with a length between $\varepsilon_{\text{PMIN}}$ and $\varepsilon_{\text{PMAX}}$ (pupil height), and the difference between adjacent valley limits (consecutive rows), both left and right, has to be less than $\varepsilon_{\text{BDIFF}}$. Finally, the maximum difference between the limit points, both left and right, has to be equal or less than $\varepsilon_{\text{BDISP}}$.

$$\begin{aligned} &\left|\lambda_i^{\text{L}} - \lambda_{i+1}^{\text{L}}\right| \leq \varepsilon_{\text{BDIFF}} \text{ and } \left|\lambda_i^{\text{R}} - \lambda_{i+1}^{\text{R}}\right| \leq \varepsilon_{\text{BDIFF}} \text{ and } \max_{i_{\text{UP}}\leq k\leq i_{\text{DOWN}}} \lambda_k^{\text{L}} \\ &\quad - \min_{i_{\text{UP}}\leq k\leq i_{\text{DOWN}}} \lambda_k^{\text{L}} \leq \varepsilon_{\text{BDISP}} \text{ and } \max_{i_{\text{UP}}\leq k\leq i_{\text{DOWN}}} \lambda_k^{\text{R}} \\ &\quad - \min_{i_{\text{UP}}\leq k\leq i_{\text{DOWN}}} \lambda_k^{\text{R}} \leq \varepsilon_{\text{BDISP}} \text{ and } \varepsilon_{\text{PMIN}} \leq i_{\text{DOWN}} \\ &\quad - i_{\text{UP}} \leq \varepsilon_{\text{PMAX}} \text{ and } i_{\text{UP}} \leq i < i_{\text{DOWN}}. \end{aligned} \tag{10}$$

In the case that there is a pupil candidate for row i, its parameters are calculated using Equation (11). The left bounding coordinate, i_{LEFT}, is the average value of all left limits of the comprised valleys, and the right bounding coordinate, i_{RIGHT}, is the average of all right limits. Then, the centroid of the pupil (i_{PX}, i_{PY}) is calculated averaging the right and left bounding coordinates and averaging the upper and lower valley row (i_{UP} and i_{DOWN}). Finally, the average intensity of all pixels into the pupil bounding box, i_C, is calculated.

$$\begin{aligned} i_{\text{LEFT}} = \operatorname*{mean}_{i_{\text{UP}}\leq k\leq i_{\text{DOWN}}} \lambda_k^{\text{L}}, i_{\text{RIGHT}} = \operatorname*{mean}_{i_{\text{UP}}\leq k\leq i_{\text{DOWN}}} \lambda_k^{\text{R}} \; i_{\text{PX}} \\ = \frac{(i_{\text{LEFT}} + i_{\text{RIGHT}})}{2\, i_{\text{PY}}} = \frac{(i_{\text{UP}} + i_{\text{DOWN}})}{2\, i_C} \\ = \operatorname*{mean}_{\substack{i_{\text{UP}}\leq i\leq i_{\text{DOWN}} \\ \lambda_i^{\text{L}}\leq j\leq \lambda_i^{\text{R}}}} p_{i.j}. \end{aligned} \tag{11}$$

In the case that there is more than one candidate, these results (i_{PX}, i_{PY}, and i_C) are stored in the vectors PX, PY and C, respectively. Then, since the pupil is always the darkest part of the image, the final pupil region

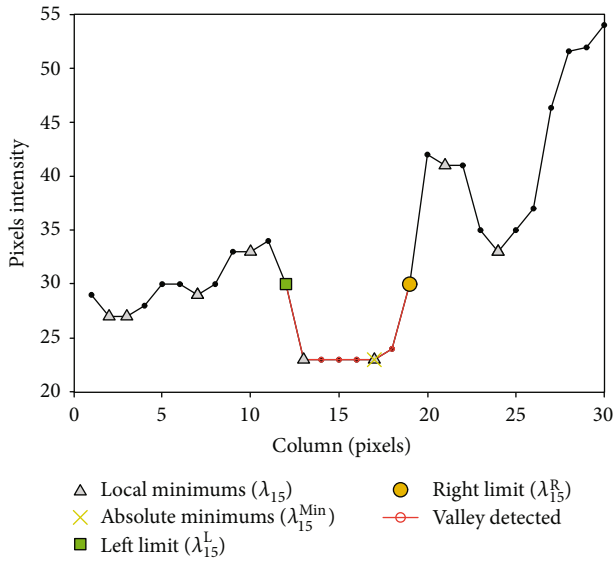


Figure 5: Intensity valley location in the 15th row of the image shown in Figure 6.

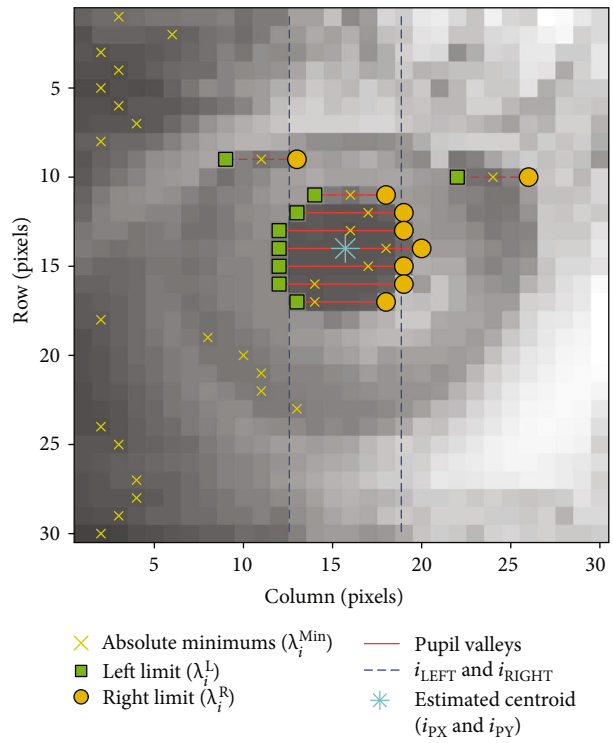


Figure 6: Pupil centroid estimation using the proposed algorithm.

selected is the one with the lowest average intensity color (Equation (12)).

$$\text{px} = PX_m, \text{py} = PY_m, c = C_m, \text{where } m = \arg \min_k C_k \quad (12)$$

Figure 6 shows an example of pupil centroid estimation using the proposed algorithm. The square and circular marks are the limit valley points λ_i^{L} and λ_i^{R} obtained in the previous step. In this example, there is only a pupil candidate and hence the final pupil is comprised between the $i_{\text{UP}} = 11$ and $i_{\text{DOWN}} = 17$ rows. The pupil centroid estimation is px = 15.71 and py = 14.0. The threshold values used in this work are $\varepsilon_{\text{BDIFF}} = 2$ and $\varepsilon_{\text{BDISP}} = 8$. These values are calculated according to the results obtained in Table 1.

3.5. Hyperparameter Estimation. In order to obtain the proposed system optimal hyperparameters, a manual human analysis of a set of 10 pupil images of 8 different users (Table 2) has been done. The images were acquired with the optical mouse sensor in a fixed distance from the eye at 60 mm. The pupil images were analyzed by manual human inspection obtaining the limits of all valleys (boundaries) that contains a portion of pupil. Then, these valley data were used to obtain four valley characteristics (Table 1) and calculate the optimal hyperparameter values (thresholds). The four valley characteristics are

1) *The absolute intensity difference between consecutive pixels*, which its average and the standard deviation are used to calculate $\varepsilon_{\text{VDIFF}}$
2) *The valley height*, which its minimum and average were used to obtain $\varepsilon_{\text{VMIN}}$ and $\varepsilon_{\text{VALLEY}}$, respectively
3) *The valley length*, which its minimum and maximum were used to obtain the valley length thresholds ($\varepsilon_{\text{PMIN}}$ and $\varepsilon_{\text{PMAX}}$), this last one includes an offset of 30% for short distance sensor placement. Also, the MNP threshold is calculated using half of the average valley length, it means that at least 50% of the valley's pixel intensity will be below $\varepsilon_{\text{MDIFF}}$
4) *The pixel intensity difference with the absolute minimum of the valley*, which the average and the standard deviation are used to calculate $\varepsilon_{\text{MDIFF}}$ taking into account that all valleys will have at least one pixel difference with this intensity

After analysing a total of 80 images of 8 different pupils, the *valley length* and *height* standard deviation remain low considering that the pupil pixels' intensity could be different for each user, and the valley length is tightly related to the pupil circular shape. *The pixel intensity difference with the absolute minimum of the valley* have more dispersion related to its average but remains stable into a feasible detection range.

The $\varepsilon_{\text{BDISP}}$ threshold, used in the pupil centroid estimation step, is calculated using the valley length limits obtained in Table 1 ($\varepsilon_{\text{BDISP}} = \varepsilon_{\text{PMAX}} - \varepsilon_{\text{PMIN}} = 8$). After a manual inspection, the maximum right and left pupil edge curve between consecutive rows is 2 pixels, then $\varepsilon_{\text{BDISP}} = 2$.

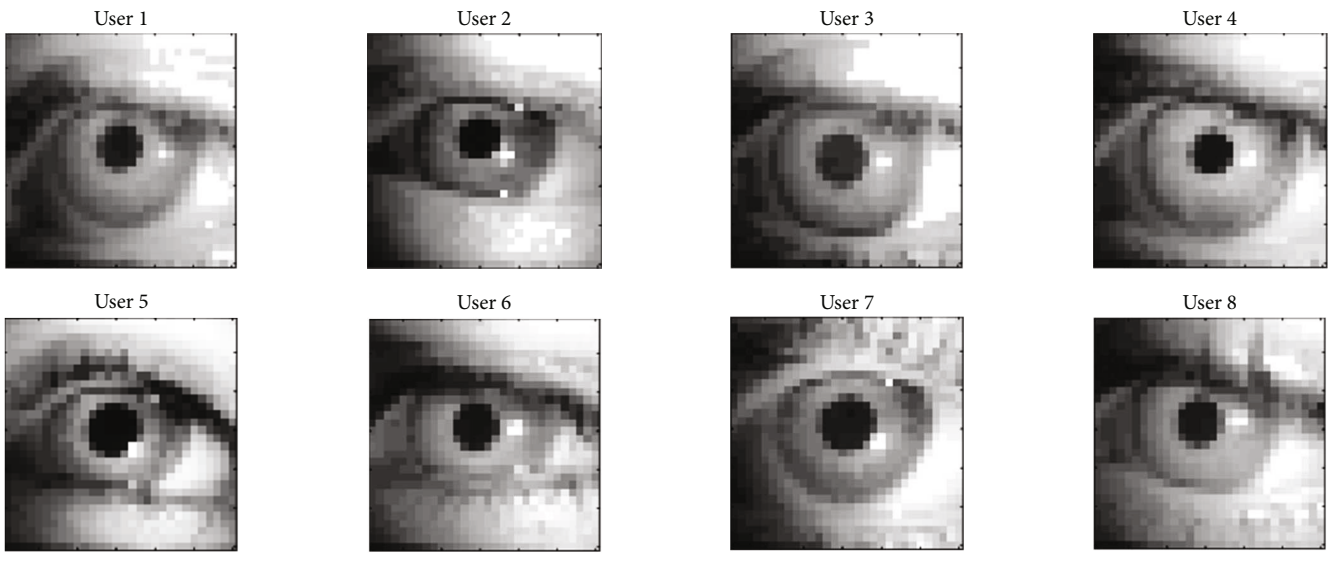
4. Eye Blink Detection

There are two eye blink under the study, the natural and the forced blink. The natural blink occurs when the user closes and opens the eyes quickly and unconsciously. These blinks can disturb the pupil tracking estimation and have to be disregarded. In case of forced blink, the user performs a controlled slow blink: closing the eyes, waiting some time, and

Table 1: Optimal hyperparameter values for valleys detection.

Valley characteristics	Min.	Max.	Avg.	σ	Hyperparameters
Absolute difference between consecutive pixels (pixel intensity)	0	11	1.2	1.77	$\varepsilon_{\text{VDIFF}} = \lfloor \text{avg} + \sigma \rfloor = 2$
Valley height (pixels)	2	15	4.63	2.41	$\varepsilon_{\text{VMIN}} = \lfloor \text{min} \rfloor = 2$ $\varepsilon_{\text{VALLEY}} = \lceil \text{avg} \rceil = 5$
Valley length (pixels)	3	8	5.10	1.01	$\varepsilon_{\text{PMIN}} = \lfloor \text{min} \rfloor = 3$ $\varepsilon_{\text{PMAX}} = \lceil \text{max} + 30\% \rceil = 11$ $\text{MNP} = \lceil 50\% \text{avg} \rceil = 3$
Difference with the minimum of the valley (pixel intensity)	0	19	1.16	1.98	$\varepsilon_{\text{MDIFF}} = \lceil \text{avg} + \sigma \rceil = 5$

Table 2: Normalized images of the volunteers' eyes after the initial adjustment.



finally opening the eyes. Many researchers propose the use of forced blinks as HCI input source [2, 5, 6, 66–74]. Most of them apply image processing techniques on an eye region image stream, such as optical flow methods [2, 66], template matching [67, 68], eye feature extraction [69], facial landmarks [70], or multiple Gabor response waves [71]. When the eye images are acquired from a head-mounted device with a reflected NIR light, the simplest techniques can be applied with very accurate results [72–74]. This scenario prevents false positive detections caused by head movements or inadequate lighting conditions. In [72], the eye blink is detected by means of difference images. This method requires an open eye reference image like in [73] where the detection of eye blinks using a simple histogram comparison of the eye region is proposed. A successful blink detection ratio of 87% is obtained.

In this work, the eye blinks are detected without additional image processing, determining if the eyes are open or closed according to whether or not the pupil position is obtained, as proposed in [74]. Then, the time while the eyes are closed indicates the blink type and, in case of forced blink, the action to perform. Although the accuracy of this method is highly depending on the pupil detection success, it does not require any additional processing time and this is a key factor for the low-performance proposed system.

Figure 7 shows the nonblocking procedure used to detect forced eye blinks by means of open-close-open eye sequences and perform actions depending on its duration. The variable *run* is used to know if the eye is closed. Then the local system time, *systim*(), updated every millisecond, is used to calculate the elapsed time without blocking the main thread (pupil detection and mouse events). The taction_n indicates the time that the eye must remain closed to execute action n, notice that the taction_n time must to be quite larger than a natural blink. The ε_{H} indicates the detection hysteresis time and finally, to validate a forced blink, a minimum elapsed time (topen) is required between a close-open eye sequence.

When the eye is closed, the eyelashes fill a significant area of the image which could generate false pupil detections in specific users. Large eyelashes also are the main disturbing source when the user looks down. The threshold constraints defined in the valley pupil location algorithm avoid these false positives. Figure 8 shows a forced eye blink image and its pupil location result, where the yellow crosses are the

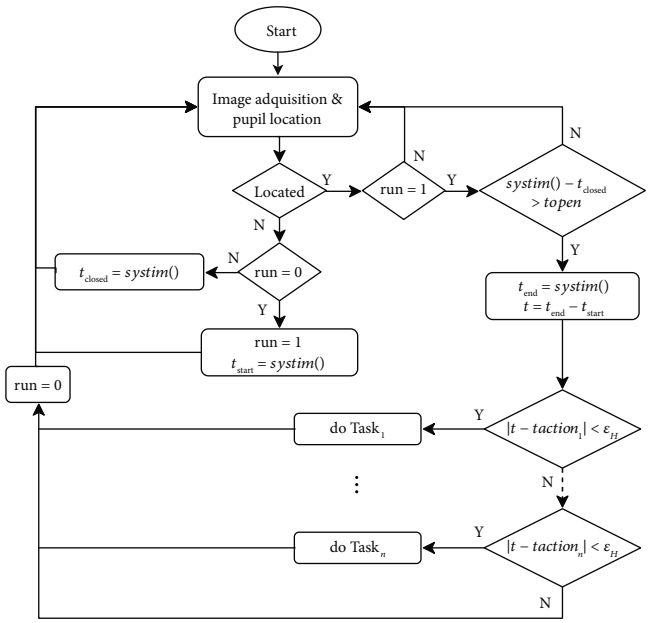


Figure 7: Flow diagram of the forced eye blink detection.

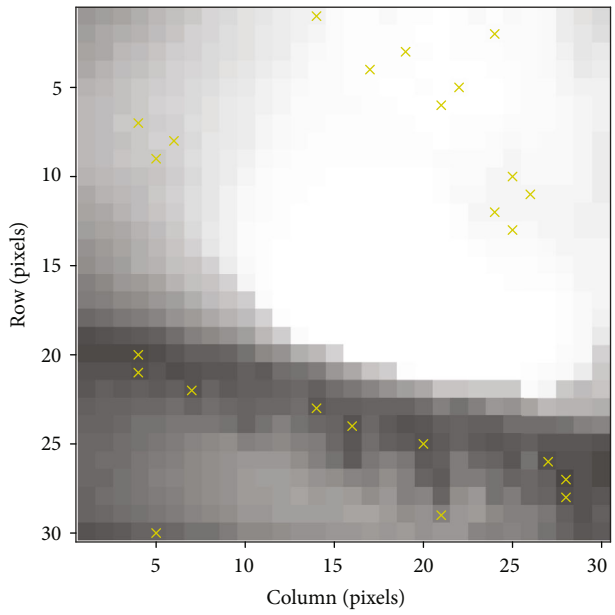


Figure 8: Captured image of the eye shut during a forced blink.

absolute minimums for each row. In this case there is not a set of rows with a minimum valley that satisfies the fixed thresholds; hence, the algorithm result is successful.

Figure 9 shows an example of a sequence of images captured during a natural blink, where the solid and dashed red lines are the detected valleys, the red solid lines are the detected pupil region and the turquoise blue asterisk indicates the located pupil centroid. While a natural blinking occurs, there could be a critical moment when the pupil is partially hidden by the top eyelid, creating a pupil location outlier (Figure 9, I^F_{n+3}). Due to the fast natural blinking speed and the low image acquisition rate (7.96 fps), this drawback can be filtered by updating the location only when at least two consecutive frames have the same result. In the frame I^F_{n+3}, a slight error in the centroid estimation was produced due to the eyelid obstruction whereas in the I^F_{n+2}, the pupil is fully hidden and there is no pupil location. According to the proposed filter, these two location results were rejected. Therefore, the estimated location was preserved in the position $(16.1, 15)$.

5. Pointing Device

5.1. Electronic Board. Figure 10 shows the single electronic board developed for the proposed pointing device including all the electronic parts embedded in a small printed circuit board (PCB). On the bottom, there is a low-cost high-performance 8-bit microcontroller PIC18F46J50 from Microchip [75] and its low-dropout regulator (LDR) of 3.3 V; on the top, which is the side towards the user's eye, there are the optical flow sensor with its external ceramic resonator of 24 MHz, a near-infrared LED for sensor light source, a 20 MHz low-profile external crystal for the microcontroller clock, and a surface-mounted device (SMD) pushbutton. Also, there are two connectors, one for microcontroller debugging and flash programing and the other for USB communication and powering. Finally, the electronic board has five SMD LEDs duplicated in both sides in order to notify the device status and help both user and assistant during the initial adjustment procedure.

The ADNS-3080 is connected by SPI using a master synchronous serial port (MSSP) of the microcontroller. An entire frame can be transmitted and stored to the 3.7 Kbytes internal RAM in 71.2 ms. Then, the microcontroller executes the pupil tracking algorithm over the received frame in an average time of 42.6 ms. Finally, the internal USB 2.0 full-speed hardware peripheral is used to translate the pupil position to user-computer actions through a USB HID standard communication class [76]. All these procedures are repeated in a closed-loop reaching a sampling rate of 7.96 fps. Although the main source clock is a 20 MHz external quartz crystal resonator, the microcontroller is configured to operate with an internal phase-locked-loop (PLL) that allows to reach the proper 48 MHz USB clock. Finally, the SHF-4350 near-infrared emitter light source is connected to one of the PWM modules of the microcontroller allowing an accurate brightness adjustment.

The board power consumption in normal operation mode, acquiring images, processing, and generating mouse events, was 555.42 mW (583.05 mW with all feedback LEDs on). The main consumption source was a dropping resistor (291 mW) which limits the SHF-4350 environmental NIR light power at 39.1 mW (with the PWM at 100%). The power consumption of the PIC18F46J50 and the ADNS-3080 were 129.42 mW and 14.67 mW, respectively. Comparing to a recent power-efficient eye-tracking solution presented in [77], this new proposal is within the estimated power range (from 70 mW to 5.25 W), using a Raspberry Pi 3 in low-power mode operating at only 1 kHz. Complex image processing requirements will increase the power consumption as the system proposed in [78] based on a high-performance 32-bit

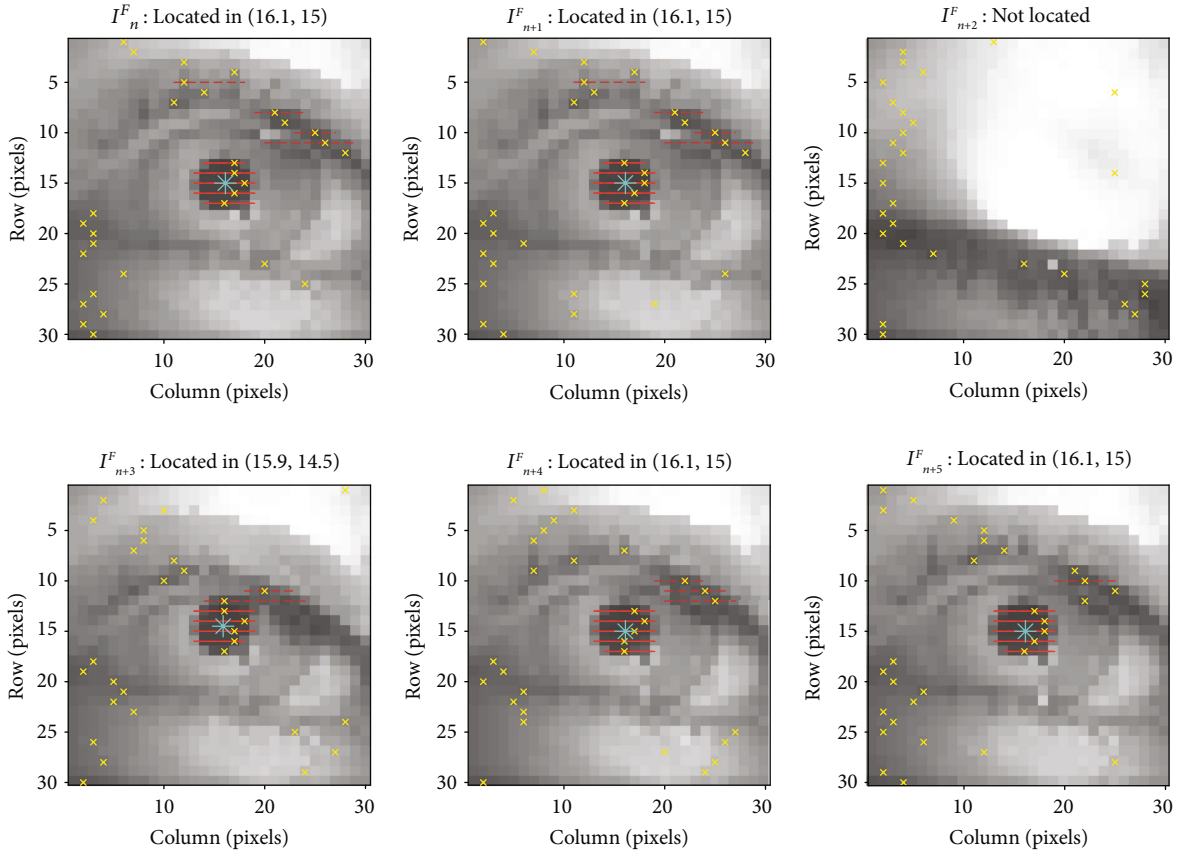


Figure 9: Example of a sequence of images captured during a natural blink.

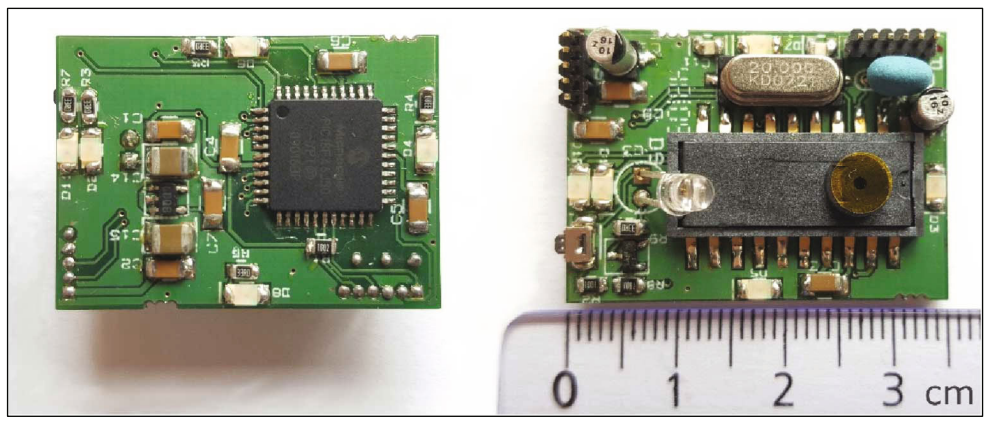


Figure 10: Electronic board of the proposed pointing device (bottom and top view).

microcontroller which consumes 703.65 mW while acquires QVGA images at a full rate.

5.2. Frame Design. The device approach is a self-contained head-mounted device in order to have the sensor fixed in front of the user within a short distance to allow pupil tracking. Since the sensor is fixed in the head, possible vibrations coming from the head, neck, or body are avoided. As a result, once the sensor is positioned it will only capture the same region of the eye. Additionally, a self-contained system also makes it more comfortable without having to install additional peripherals such as cameras, acquisition boards, or light sources, on different locations with a lot of wires.

Figure 11 shows the frame design proposed in this work. It is composed of a stick that holds a small box with the electronics in front of the user's face. This stick is placed on the right side of the head and is held using headbands joined over the right ear which helps to keep the structure in place. The

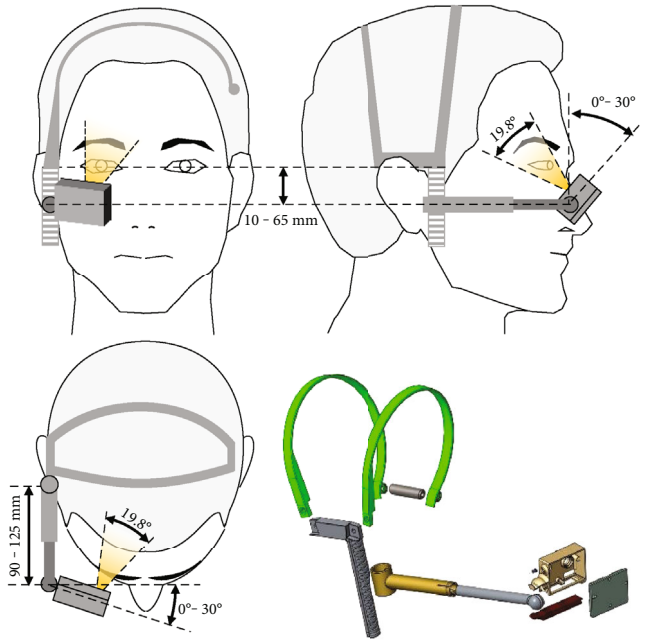


Figure 11: Front, right, and top view of the frame design (camera angles in orange) and the disassembled sketch.

structure also contains some adjustable parts to facilitate the sensor placement. It has up to 5 degrees of freedom (DOF): height, length, and horizontal rotation of the stick, and azimuth and elevation of the box. This approach has only 10 plastic parts and a very simple assembly aiming to make the device as inexpensive as possible.

Figure 12 shows a close view of the box (36.3×15.2×28.8 mm) where the electronic board is placed. There are holes in the frame to increase the brightness of the front and rear LEDs. Their peripheral position and close placement to the eye help to perceive the LED notifications without stopping to stare at the screen. On the right side of the box, a moving cylindrical plastic part has been designed to handle the electronic board's pushbutton. Finally, a slot within the stick is used to hide the USB cable which comes from the box.

5.3. Human-Computer Interaction. The operating process of the device is composed of three main operating states (Figure 13): the initial adjustment, where the user has to correctly place and fix the device to their head; the mouse emulation, where the user's eye movements are translated into computer pointer actions; and the paused stage, where all interaction capabilities are disabled to freely move the gaze. The flow diagram of the interaction between operation stages is presented in Figure 13.

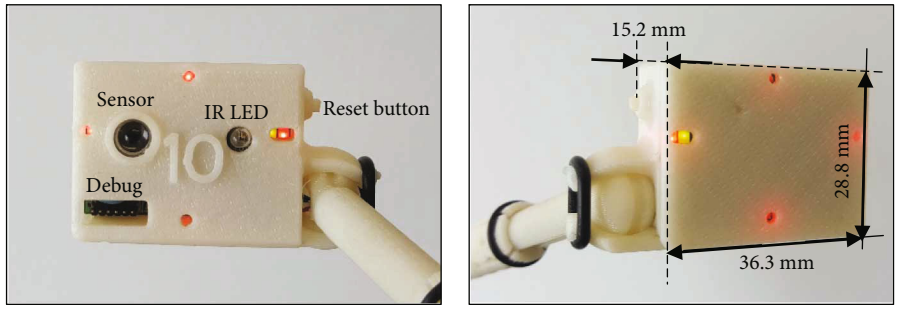
Once the device is connected and automatically detected by the operation system, the pointer is blocked permanently at the center of the screen and the device awaits the initial adjustment. In this stage, the user has to stay seated properly holding the device in a working distance of around 60 cm and with the eyes horizontally aligned with the pointer. The device then waits until it detects a pupil in the center of the images acquired (±4 pixels offset for both axis) with a valid diameter (between 4 and 16 pixels). The four peripheral LEDs offer positioning feedback to facilitate the manual centering of the sensor box. If the pupil is detected in one direction, for example on the left, the red LED of that direction will light on. The blinking frequency depends on the pupil size detected. If there is no blinking it means that the pupil size is in the valid range. Once the pupil appears into the expected area with the expected size during 5 s, the current position of the pupil will be stored as reference position (x_c, y_c) and the green LED will blink indicating that the adjustment stage is finished giving way to the mouse emulation stage. This initial manual adjustment may require the help of an assistant in the case of a user with impaired mobility. At any time, the device can be restarted by holding the pushbutton during 6 s or forcing no pupil detection during more than 6 s.

In the mouse emulation stage, the user interaction by controlling HID mouse events is carried out. The pupil displacements and forced eye blinks actions are translated to X and Y relative movements and left and right clicks. The proposed system achieved a small horizontal and vertical pupil detection variation range of 10.84 and 8.51 pixels when looking at a 19″ 4:3 TFT computer screen at a distance of 60 cm. It is a significant limitation to use this system as a usual eye tracking, thus a relative-based computer mouse displacement must be implemented. In this work, five pupil regions are defined (left, right, up, down, and center), see Equation (13) and Figure 14. These regions are enough for a relative mouse emulation and increase the detection robustness. Figure 14 shows a representation of the regions in an image captured by the optical sensor while the user is looking at the left edge of the screen (the pupil is inside the left region, L_R). To force the activation of a region, the user has to keep the pupil for two consecutive frames in that region. The user can force left, right, up, and down regions (L_R, R_R, U_R, D_R, respectively) looking at the edges of the screen and looking at the center for the central region (C_R). When the position of the pupil is calculated, Equation (13) identifies in which region it is.

$$R(x,y) = \begin{cases} C_R \text{ if } \dfrac{(x-x_c)^2}{r_{maj}^{\,2}} + \dfrac{(y-y_c)^2}{r_{\min}^{\,2}} < 1 \\ \text{otherwise :} \\ L_R \quad \text{if} \quad y > f_1(x) \text{ and } y \le f_2(x) \\ R_R \quad \text{if} \quad y < f_1(x) \text{ and } y \ge f_2(x) \\ U_R \quad \text{if} \quad y \le f_1(x) \text{ and } y < f_2(x) \\ D_R \quad \text{if} \quad y \ge f_1(x) \text{ and } y > f_2(x) \end{cases}, \quad \begin{aligned} f_1(x) &= x - x_c + y_c \\ f_2(x) &= -x + x_c + y_c \end{aligned} \tag{13}$$

An ellipse with major axis (r_{maj}) of 2.8 pixels and minor axis ($r_{\min}$) of 1.9 pixels located in (x_c, y_c) defines the central region considering that the horizontal pupil movements are larger than the vertical ones. Two linear functions (f_1, f_2) are used to delimit the left, right, up, and down regions.

One of the main problems of the eye movement-based interaction is to combine nonintentional user gaze, looking



(a) (b)

Figure 12: Front (a) and rear (b) view of the box where the electronic board is placed.

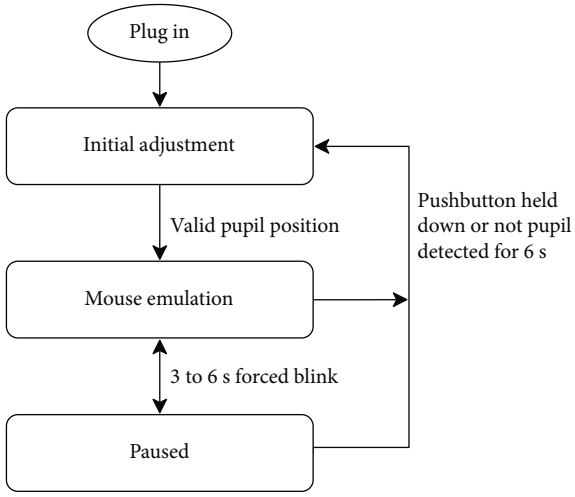


Figure 13: Flow diagram of the device operation states.

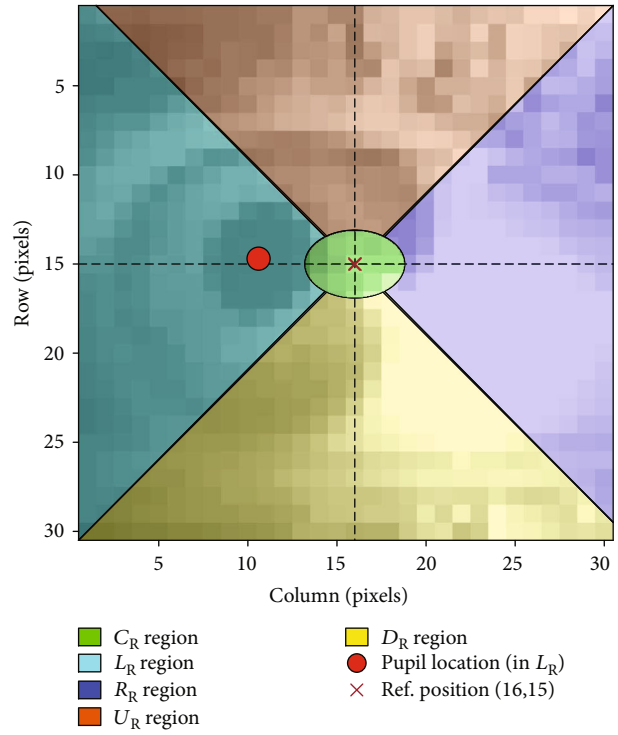


Figure 14: Example of regions placement and region determination of an acquired image.

naturally to an item or target on the display screen, with the generation of pointing actions. This problem, called Midas Touch problem, has been overcome in several research works [79, 80]. Most of them use flags (pushbuttons, log dwell times, etc.) to trigger an eye-controlled action. In [80], the authors propose a fast disengaging gaze control method to overcome the effects of moving the gaze point to look at the result of the input. In this work, the mouse events are controlled by means of moving the gaze in a predefined sequences of regions, called combos, which the user can trigger any moment during the emulation stage. Table 3 shows the list of possible combinations to perform and their description. Apart from LC_c, the combos must start when the gaze remains at lest of 1 s in the central region (C_R). The regions visited in a combo must remain active for less than 800 ms. This forces the user to perform a fast sequence of gaze regions. Finally, the combos end and commits to a mouse event remaining again in the C_R region for at least 1 s. Although this dwell times can be fatiguing, it is the way to ensure that any combo is intended by the user. Thus, the system runs the mouse emulation algorithm (Figure 15) while the user can look naturally to the screen without disengaging the emulation.

Figure 15 shows the flow diagram of the algorithm that transforms the displacements combos (L_C, R_C, U_C, D_C, LC_C) into USB HID mouse pointer displacements. When a L_C, R_C, U_C, or D_C is performed, the pointer starts moving in that direction at an exponential speed. The current pointer position is updated every $T_0 = 125.6$ ms and the pixels variation (Δp) is calculated using the Equation (14).

$$\Delta p(kT_0) = \Delta p(kT_0 - T_0) + K_p \cdot \Delta p(kT_0 - T_0)^3, \Delta p(0) = 1 \text{ and } \Delta p \leq 127 \tag{14}$$

The K_p factor by default is 0.008, although it can be modified using the device pushbutton to adjust the cursor speed to the screen size and resolution. In case of a common 1920 × 1080 resolution and using the default K_p, the pointer takes 6.91 s to cross horizontally the screen. The Δp will be added

Table 3: List of combos to perform pointer actions.

Combo	Description	Abbreviation
$C_R + L_R + C_R$	Left displacement	L_C
$C_R + R_R + C_R$	Right displacement	R_C
$C_R + U_R + C_R$	Up displacement	U_C
$C_R + D_R + C_R$	Down displacement	D_C
Forced blink	Stop displacement/left-click	LC_C
$C_R + L_R + R_R + C_R$ $C_R + R_R + L_R + C_R$	Right-click (pop-up menu) [1]	RC_C
$C_R + U_R + D_R + C_R$ $C_R + D_R + U_R + C_R$	Double-click[1]	DC_C

[1] There are two possible combinations.

or subtracted from the respective axis depending on the movement's direction. The mouse pointer can also move in diagonal by activating both displacement combos (not necessary sequentially). Then, the amount of pixels to update in both axis in the sample k is normalized using

$$\Delta p(kT_0) = \frac{\Delta p(kT_0)}{\sqrt{2}}. \quad (15)$$

In this approach, the user can stop the mouse pointer movement of an axis doing the opposite combo, whereas a forced blink (LC_C) stops all of the active movements at once. In case that the mouse pointer is still, the forced blink does left clicks.

6. Results and Discussion

The pointing device proposed were tested by 8 volunteers aged between 23 and 38 who had no impaired mobility and did not wear glasses or contact lenses. Three volunteers were blue-eyed, and five, brown-eyed. They were seated in front of a 19″ 4 : 3 TFT computer screen at a distance of 60 cm. During each experiment, the users themselves did the initial adjustment stage following the actions explained in the previous section. Table 2 shows a sensor image capture of each user after the initial adjustment and Table 4 summarizes the results of this stage. The average time required to set up the device before emulation was 33 s. The average pupil diameter was barely 5.5 pixels because all users tended to place the optical sensor far from the eye to minimize obstruction and maximize the view angle of the screen. The obtained reference positions, as expected, were into a restricted region of ±4 pixels offset regarding the center of the image.

Once the initial adjustment was done, two different user tests were performed. In the first test, the user was asked to do a fixed sequence of pupil movements looking at the left, right, up, and down edges of the screen (screen frame) as well as its center in order to validate the detection of the defined L_R, R_R, U_R, D_R, and C_R regions. An average recording time of 27.7 s were required for each user with an image acquisition rate of 11.84 fps.

Figure 16 shows, for each user, the pupil position error during the movement sequence. In order to evaluate the detection improvement, the pupil was located, frame by frame, with three different algorithms which could be implemented in our proposed system: the proposed in this work, our previous approach [44], and the integrodifferential operator [46]. Also, for each frame, a manual pupil location was done and used as a reference. The Euclidian distance between the reference points and the algorithms' result was considered the error in pixels.

In the case of the proposed algorithm, the average median error (red lines inside the boxes) for all users was 0.34 pixels ($\sigma = 0.38$), whereas 0.41 pixels ($\sigma = 1.56$) and 1.39 pixels ($\sigma = 1.07$) pixels for the previous and integrodifferential algorithms, respectively. The average distribution between the 25% (lower quartile) and 75% (upper quartile) of the values (box height) was between 0.21 and 0.50 pixels for the proposed algorithm. In case of the previous approach, there was a slightly poor distribution (between 0.27 and 0.59 pixels) but better than the integrodifferential (between 1.09 and 1.68 pixels). The integrodifferential operator is very sensible with the differences produced between the eyelids and sclera which depicts an arc shape. Also, it always returns a pupil position candidate (maximum confidence), thus requiring an additional image processing to detect whether there is a blink or not and discard false positive detections during blinks.

As the results show in Figure 16, the proposed algorithm has an average minimum and maximum values within the ±1.5 IQR (interquartile), the whiskers, of 0.01 and 0.93 pixels, respectively, better than the previous approach (within 0.03 and 1.02 pixels). The error for the valley detection algorithms does not depend on the user and, overall, it remains below 1.5 pixels. However, extreme outlier errors appeared in some users during blinks because of eyelashes or eyelid obstruction while closing the eye. In the previous approach, these outliers occur in a 3.01% of the cases, generating a maximum error of 14.58 pixels, whereas in the new proposed algorithm, they were reduced to 0.25% with maximum error of 9.14 pixels. This significant improvement is because of a new set of hyperparameters and thresholds introduced in the algorithm. Furthermore, the natural eye blink problem was overcome in the emulation process applying a filter that rejects sudden pupil position changes as explained in the eye blink detection section (Figure 9).

Figure 17 show three sensor images of a volunteer eye, wearing glasses, contact lenses, and nothing. The pupil location was obtained manually and with the proposed method, showing the difference (error) on the images' title. The yellow crosses are the absolute minimums for each row, the red lines are the detected valleys, and the turquoise blue asterisk indicates the pupil center. As the image shows, the glass lenses degrade the image quality and produces important highlights of the NIR LED. It is not possible to have a stable pupil detection with this conditions. On the contrary, the contact lenses did not affect the image quality. The pupil position detection was as good as the detection without intrusion. Only an insignificant misalignment of 0.15 pixels was obtained in the images showed, and, whether wearing contact lenses or not, the volunteer could operate with the device properly.

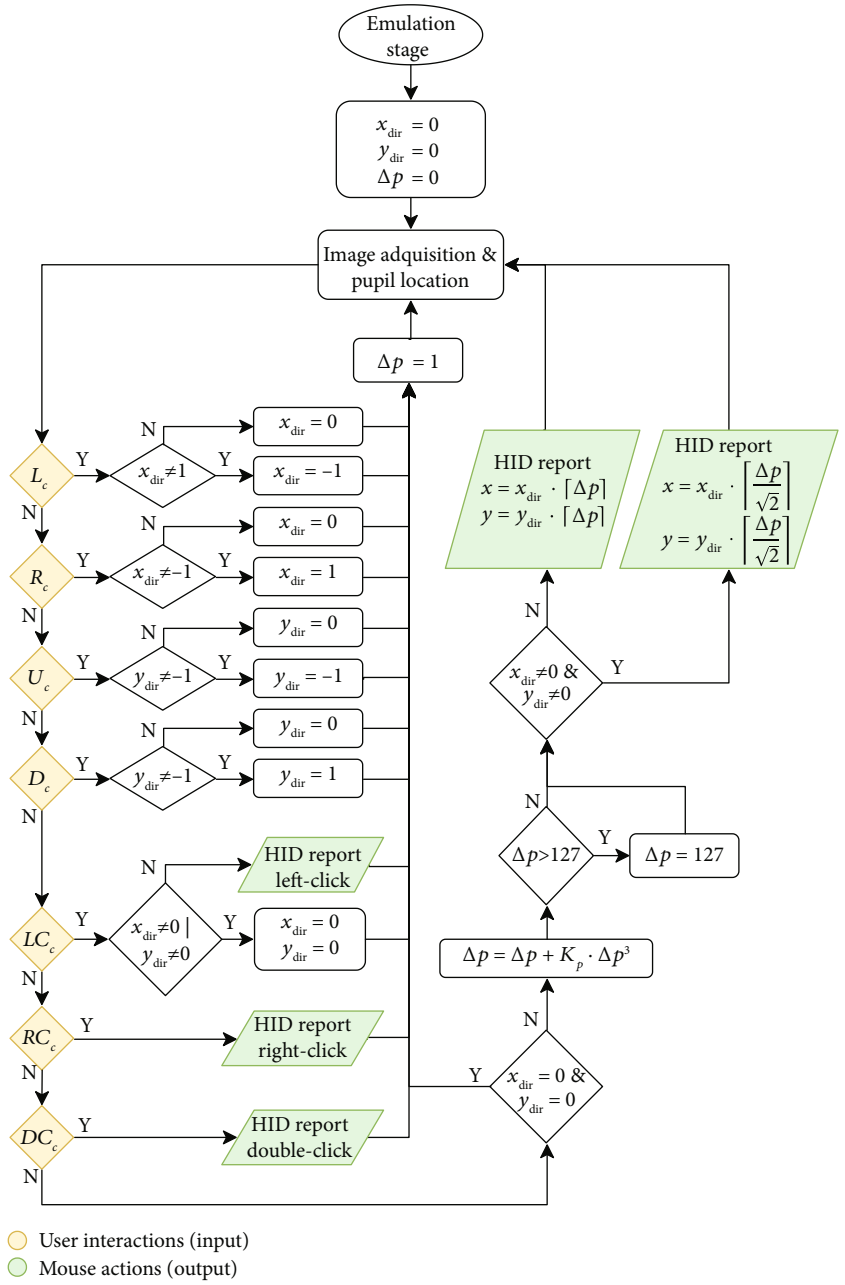


Figure 15: Flow diagram to translate user actions (combos) into mouse actions.

Table 4: Results in the adjustment for each volunteer.

User	Adjustment time (s)	Pupil diameter (pixels)	Reference position (x_c, y_c)
1	16	5.3	(15.4, 15.1)
2	26	4.6	(15.1, 14.8)
3	54	6.0	(14.8, 16.8)
4	28	4.6	(15.9, 15.6)
5	33	5.6	(15.3, 14.8)
6	36	6.3	(14.8, 15.2)
7	19	5.6	(15.6, 13.8)
8	48	6.0	(14.1, 14.0)
Average	33	5.5	(15.1, 15.0)

Figure 18 shows the user 8 first test pupil tracking results and the respective detected region. The manual pupil tracking (analyzing images by a human) and the estimated pupil tracking (using the proposed algorithm) are shown. The results are very similar and the pixel error was constrained into the expected margins. Likewise, the estimated region was calculated, frame by frame with no filters applied, by the function $R(x, y)$ (Equation (13)). Figure 18(a) shows the regions estimated over time and their success considering the correct region as the region calculated with the reference pupil position (obtained manually). As the results show, the detection of the different regions was successful: the user looked once at each edge region starting always from the central one (C_R). However, when crossing between regions,

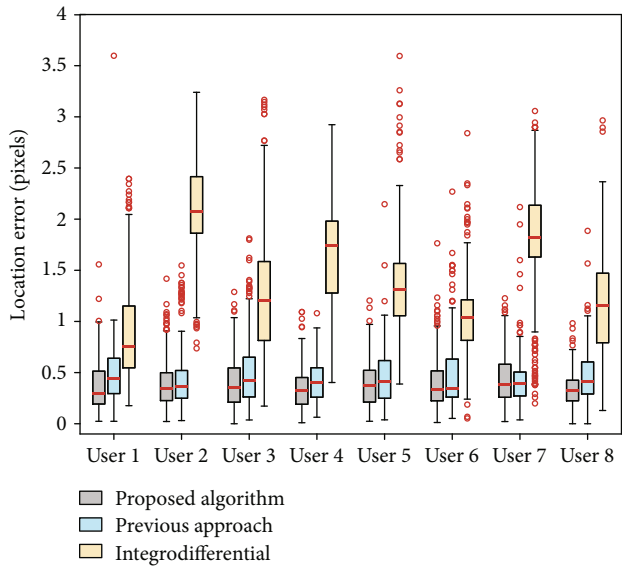


Figure 16: Pupil position error obtained using the proposed algorithm compared with the previous approach [44] and the integrodifferential [46].

some pupil location errors were detected. In the user 8 case, there were 4 frames that generated an incorrect region (red circles in Figure 18) which were considered as unsuccessful region detection of that type. In that case, the success on detecting regions was 95.2% for L_R, 96.8% for R_R, 96.5% for U_R, 95.8% for D_R, and 100% for C_R.

Table 5 reports the success in the regions detection for each user. In general, all displacements were detected correctly at the limit of the movements (when the user looked at the frame of the screen). The main error source appeared when the pupil crossed region borders, as shown in Figure 18 (red circles). Due to that error, the region success highly depends on the elapsed time in each region. Keeping the region time as similar as possible to each other was necessary. As the results show in Table 4, the regions were successfully detected in 94.7% of the analyzed cases corresponding to the eye left, right, up, and down orientations. There were no significant differences between users with blue and brown eyes. The highest success was obtained when moving the eye to the right edge with 97.1% whereas the worst detection results were obtained in the displacements to the opposite side (left edge) with 91.8% of success. This was mainly because the sensor was placed to the right of the eye, capturing less displacement to the left and, in addition, the left side had worse infrared illumination. When the user was looking to the upper edge, the region was correctly detected 92.0% of the cases. This displacement has the shortest eye movement and it is highly sensitive to unexpected head motion variations. When looking to the bottom edge, the average successful region detection was 96.8% although it was complex to detect with users who had large eyebrows. Finally, the central region was successfully detected in 95.8% of the cases. This is the smallest region area hence its detection extremely depends on keeping the head motionless in order to maintain the initial reference position. Due to the small displacement range of the pupil, the central region could not be enlarged since it would affect the result of the other regions. Unlike the edge regions, where the user has the screen's frame as a reference point to look at, the central region does not have a reference mark and, after certain time, the user fluctuates his gaze generating false detections. This weakness could be overcome by updating the reference position every time the user gaze remains inside the central region.

Finally, in the second test, different combo actions were carried out (see Table 3), one followed by the other, by looking at the edges (the frame) and the center of the screen as requested. The user was asked to perform a sequence of 7 combos $(L_C, R_C, U_C, D_C, LC_C, RC_C, DC_C)$ 5 times, where each combo was tested consecutively twice. It is important to mention that the volunteers were untrained users and had not tested the device before. The experiments were done in the laboratory but simulating a real working scenario: an office desktop with untrained users. The results with users outside the laboratory should be similar despite the initial adjustment that requires a certain experience level. The proposed device is robust in front of illumination changes due to its direct external NIR LED, which avoids an indoor illumination adjustment.

Despite the initial problems to perform combos during the first contact, at the end of the test, most of the volunteers felt comfortable doing each combo successfully, repeating them twice in the worst case. As the results show in Table 6, the best combo detection was the forced blink (LC_C) which was successful 97.5%. The combos for vertical and horizontal displacements $(L_C, R_C, U_C$, and $D_C)$ had a similar success rate with an average correct detection of 84.1%. These were easier to perform than the LC_C and RC_C combos, since moving between opposite regions in a very short time was needed. Consequently, these combos had the worst success result with 70.0% and 75.0%, respectively.

Working in a real scenario, the main weak point of the proposed system is the reference position sensibility obtained in the initial adjustment. The detection of each region depends on this reference point and, due to the low pupil displacement range, a small misalignment harms seriously the overall performance. In this work, the tests were all done by healthy users, and the main problem lay on having the head still. In case of using the sensor for the first time, head tics were detected because of the initial tension. Furthermore, after using the system for a while, volunteers tend to slightly down the head and, in some cases, they moved the head unconsciously while were concentrated doing mouse events. Although the proposed device is designed for people with severe disabilities who cannot move their head, it is reasonable to think that the slippage problem could be present because of head tics or gravity. In case of product commercialization, this problem must be addressed, for example, with the method proposed in [81]. Also, it could be solved by attaching a motion sensor to the headset, like an accelerometer, tracking and adjusting the reference mismatches. Likewise, this critical reference point could be dynamically updated by tracking the pupil maximum displacements and readjusting the regions

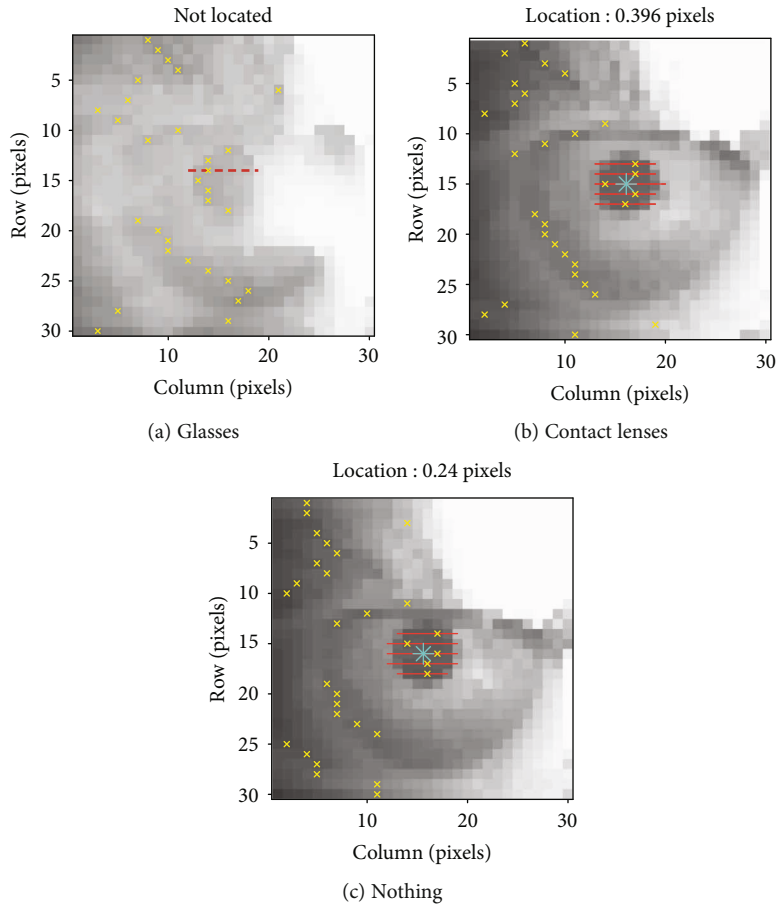


Figure 17: Example of images acquired of a volunteer wearing glasses (a), wearing contact lenses (b) and wearing nothing (c), and the pupil location result with the proposed method.

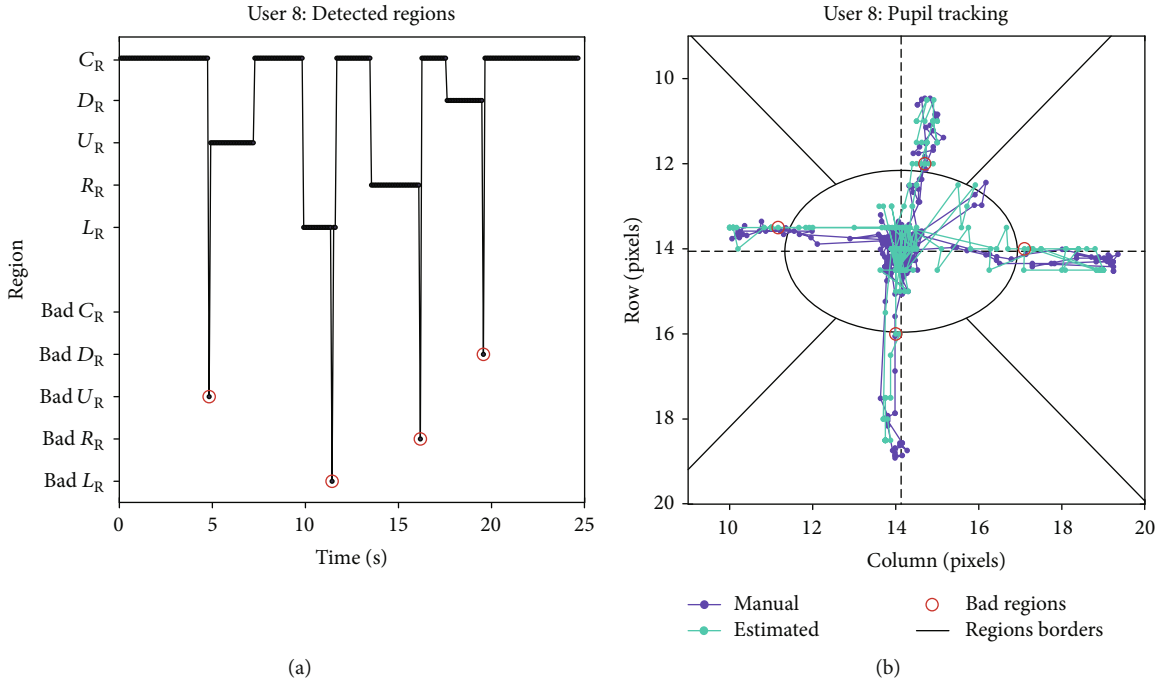


Figure 18: Detected regions and pupil tracking result during the user 8 first test.

Table 5: Device performance in detecting regions for each volunteer.

User	Success in detecting regions (%)					
	L_R	R_R	U_R	D_R	C_R	Average
1	100	100	100	100	93.2	97.2
2	100	100	91.6	93.3	93.3	95.6
3	92.8	100	100	92.8	94.2	95.9
4	87.5	86.9	100	100	98.2	94.5
5	70.0	100	63.1	92.8	93.6	83.9
6	92.3	93.3	92.0	100	95.9	94.7
7	96.6	100	100	100	97.9	98.9
8	95.2	96.8	96.5	95.8	100	96.8
Average	91.8	97.1	92.0	96.8	95.8	94.7 [1]

[1] Average of all tries.

Table 6: Device performance in carrying out combos for each volunteer.

User	Success in carrying out combos (%)							
	L_C	R_C	U_C	D_C	LC_C	RC_C	DC_C	Average
1	80	100	80	80	100	100	60	85.7
2	50	80	100	60	100	20	70	68.5
3	100	60	80	80	90	70	80	80.0
4	100	50	100	70	100	80	80	82.8
5	100	90	90	90	100	70	70	87.1
6	80	100	100	100	100	70	90	91.4
7	70	80	100	70	90	70	50	75.7
8	70	100	90	90	100	80	100	90.0
Average	81.2	82.5	92.5	80.0	97.5	70.0	75.0	82.6 [1]

[1]Average of all mouse actions requested.

(L_R, R_R, U_R, D_R), but taking care of the increase in computational cost and the system performance.

Another point to improve is the head-mounted device clamping system. The device is very lightweight (41 g), comfortable, and low intrusive, but has problems with straight long hair users (slides forward). In some cases, due to a small device forward movement, the pupil got out of the camera field and unexpected pointer pauses were trigged. Hence, the initial adjustment procedure had to be repeated. A solution could a third headband around the crown to improve the head grip.

The experimental results combining user interaction and natural gaze (Midas Touch problem [79]) showed difficulties generating combos. Usually, users fail in the first trial because they do not understand how to perform mouse events. It is understandable that looking at multiple extreme eye gaze points (screen edges) within a controlled dwell time could be upsetting; however, this solution proved no false positive mouse events triggered during the natural eye gaze. Also, because the combos' sequences are very different from each other, there were no events triggered (combos) by mistake.

Although the users had interaction difficulties in the first contact, all of them could perform the proposed combos (in some cases after two or three tries). In the case of the forced eye blink combo, once the user felt comfortable with the blink time, it was obtained 100% success, meaning the eye blinks were always detected properly and the complex part was to get used with the dwell time. Taking advantage of this robustness, an interaction method based on a fast engaging/-disengaging mouse control could be implemented [80]. This could improve its usability, doing the interaction more natural by translating directly screen edges eyes gaze to relative mouse displacements; however, the main concern is how to look at a certain screen position (to trigger an event) and its result at once.

7. Conclusions

A new implementation of an inexpensive eye-controlled human-computer interface device using an optical mouse sensor is presented. The device takes advantage of the image acquisition capabilities of the ADNS-3080 low-cost optical mouse sensor, originally designed to operate as a displacement sensor, for pupil detection and tracking. Its default configuration of lenses and illumination, suited to work at short focal distance of 2.4 mm, has been replaced with a low-cost plastic CAY46 aspheric and an external NIR LED with a wavelength peak of 850 nm in order to obtain sharp images of the eye and the pupil. This proposal takes full advantage of the infrared wavelength responsivity of the optical mouse sensors to detect the pupil as the darkest part in the images.

An optimized algorithm to locate the pupil centroid in the acquired low-resolution sensor images (30×30 pixels) is detailed. Although it has similar detection performances than state of the art algorithms, it can be easily integrated in a low-cost microcontroller without the help of any external memory. Moreover, a procedure to detect forced eye blinks is implemented in order to perform different mouse actions depending on the time that the eye remains closed (no pupil detection) rejecting natural blinks.

The proposed pointing device has been fully implemented and evaluated in terms of head-mounted structure, electronics design, and human-computer interaction and operation. It was tested on 8 volunteer users detecting and locating the pupil successfully for all of them with an average error of 0.34 pixels. The performance in the detection of the 5 defined image regions was successful in 94.7% of the cases. The set of combined sequence of pupil actions (combo actions) also has been successfully generated for all 8 users. In the combo sequences for pointer displacements, 84.1% were successful on the first attempt. In the case of forced blink actions such as left-clicking, 97.5% of the cases were successful. The pupil movement combination for right- and double-click was the worst result with an average success of 72.5% since these require a fast sequence of opposite pupil locations.

The validation results obtained confirm that the low-cost optical mouse sensor is capable of detecting pupil displacements and can be applied as a pupil tracking sensor in low-cost human-computer interface devices. Although the

usability of the proposed pointing device is far from the being like the current computer mouse, it could be a very interesting alternative as affordable interface device for users with severe disability in the upper extremities.

Data Availability

The data used to support the findings of this study are included within the article.

Conflicts of Interest

The authors declare no conflict of interest.

Acknowledgments

This research was funded by Indra, Accessibility Chair, 2017. This research also was supported by the Government of Catalonia (Comissionat per a Universitats i Recerca, Departament d'Innovació, Universitats i Empresa) and the European Social Fund.

References

[1] W. Zhang, M. L. Smith, L. N. Smith, and A. Farooq, "Gender and gaze gesture recognition for human-computer interaction," *Computer Vision and Image Understanding*, vol. 149, pp. 32–50, 2016.

[2] T. Pallejà, A. Guillamet, M. Tresanchez et al., "Implementation of a robust absolute virtual head mouse combining face detection, template matching and optical flow algorithms," *Telecommunication Systems*, vol. 52, no. 3, pp. 1479–1489, 2013.

[3] W. Nutt, C. Arlanch, S. Nigg, and G. Staufert, "Tongue-mouse for quadriplegics," *Journal of Micromechanics and Microengineering*, vol. 8, no. 2, pp. 155–157, 1998.

[4] C. Lin, J. Chen, S. Yang, and Y. Cao, "Design and implementation of a mouth-controlled mouse," in *Presented at the 2011 IEEE EUROCON - International Conference on Computer as a Tool*, pp. 1–4, Lisbon, Portugal, 27-29 April 2011.

[5] D. Hyun and M. Jin, "Eye-mouse under large head movement for human-computer interface," in *In Proceedings of the IEEE Int. Conf. on Robotics and Automation, 2004 (ICRA '04)*, pp. 237–242, New Orleans, LA, USA, 26 April-1 May 2004.

[6] J. Tu, H. Tao, and T. Huang, "Face as mouse through visual face tracking," *Computer Vision and Image Understanding*, vol. 108, no. 1–2, pp. 35–40, 2007.

[7] G.-M. Eom, K.-S. Kim, C.-S. Kim et al., "Gyro-mouse for the disabled: 'click' and 'position' control of the mouse cursor," *International Journal of Control, Automation, and Systems*, vol. 5, no. 2, pp. 147–154, 2007.

[8] G. Rosas-Cholula, J. Ramirez-Cortes, V. Alarcon-Aquino, P. Gomez-Gil, J. Rangel-Magdaleno, and C. Reyes-Garcia, "Gyroscope-driven mouse pointer with an EMOTIV® EEG headset and data analysis based on empirical mode decomposition," *Sensors*, vol. 13, no. 8, pp. 10561–10583, 2013.

[9] A. De Santis and D. Iacoviello, "Robust real time eye tracking for computer interface for disabled people," *Computer Methods and Programs in Biomedicine*, vol. 96, no. 1, pp. 1–11, 2009.

[10] E. Cáceres, M. Carrasco, and S. Ríos, "Evaluation of an eye-pointer interaction device for human-computer interaction," *Heliyon*, vol. 4, no. 3, p. e00574, 2018.

[11] R. B. Reilly and M. J. O'Malley, "Adaptive noncontact gesture-based system for augmentative communication," *IEEE Transactions on Rehabilitation Engineering*, vol. 7, no. 2, pp. 174–182, 1999.

[12] D. G. Evans, R. Drew, and P. Blenkhorn, "Controlling mouse pointer position using an infrared head-operated joystick," *IEEE Transactions on Rehabilitation Engineering*, vol. 8, no. 1, pp. 107–117, 2000.

[13] M. Mazo, J. C. García, F. J. Rodríguez, J. Ureña, J. L. Lázaro, and F. Espinosa, "Experiences in assisted mobility: the SIAMO project," in *Proceedings of the International Conference on Control Applications (CCA/CACSD'02)*, pp. 766–771, Glasgow, UK, 18-20 September 2002.

[14] F. Khan, S. K. Leem, and S. H. Cho, "Human-computer interaction using radio sensor for people with severe disability," *Sensors and Actuators A: Physical*, vol. 282, pp. 39–54, 2018.

[15] X. Huo, J. Wang, and M. Ghovanloo, "A magneto-inductive sensor based wireless tongue-computer interface," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 16, no. 5, pp. 497–504, 2008.

[16] M. N. Sahadat, A. Alreja, N. Mikail, and M. Ghovanloo, "Comparing the use of single versus multiple combined abilities in conducting complex computer tasks hands-free," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 26, no. 9, pp. 1868–1877, 2018.

[17] D. Ming, Y. Zhu, H. Qi, B. Wan, Y. Hu, and K. D. K. Luk, "Study on EEG-based mouse system by using brain-computer interface," in *In Proceedings of the IEEE International Conference on Virtual Environments, Human-Computer Interfaces and Measurements Systems (VECIMS 2009)*, pp. 236–239, Hong Kong, China, 11-13 May 2009.

[18] R. Raj, S. Deb, and P. Bhattacharya, "Brain Computer Interfaced Single Key Omni Directional Pointing and Command System: a Screen Pointing Interface for Differently-abled Person," *Procedia Computer Science*, vol. 133, pp. 161–168, 2018.

[19] W. McClay, "A Magnetoencephalographic/encephalographic (MEG/EEG) brain-computer interface driver for interactive iOS mobile videogame applications utilizing the Hadoop Ecosystem, MongoDB, and Cassandra NoSQL databases," *Diseases*, vol. 6, no. 4, p. 89, 2018.

[20] Y. Ebisawa, D. Ishima, S. Inoue, and Y. Murayama, "Pupil-Mouse: cursor control by head rotation using pupil detection technique," in *In Proceedings of International Conference on Computing, Communications and Control Technologies*, pp. 209–214, Austin-Texas, USA, 14-17 August 2004.

[21] A. Technologies, "Intelli Gaze CAM30NT," July 2019, http://www.intelligaze.com/.

[22] T. Technology, "Tobii X2-30," July 2019, http://www.tobii.com.

[23] EyeTech Digital Systems Inc, "TM5 Mini," July 2019, http://www.eyetechds.com.

[24] LC Technology Inc, "The Eyegaze Edge," July 2019, http://www.eyegaze.com.

[25] C.-S. Lin, C.-W. Ho, C.-N. Chan, C.-R. Chau, Y.-C. Wu, and M.-S. Yeh, "An eye-tracking and head-control system using movement increment-coordinate method," *Optics & Laser Technology*, vol. 39, no. 6, pp. 1218–1225, 2007.

[26] D. Li, J. Babcock, and D. J. Parkhurst, "openEyes: a low-cost head-mounted eye-tracking solution," in *In Proceedings of*

the 2006 Symposium on Eye Tracking Research and Applications, pp. 95–100, San Diego, California, USA, 27-29 March 2006.

[27] B. Li, H. Fu, D. Wen, and W. L. LO, "Etracker: a mobile gaze-tracking system with near-eye display based on a combined gaze-tracking algorithm," *Sensors*, vol. 18, no. 5, p. 1626, 2018.

[28] HK EyeCan Ltd, "Eye Tracker Eyecan," July 2019, http://www.eyecan.ca.

[29] T. Technology, "Tobii Glasses II," July 2019, http://www.tobii.com.

[30] PupilLabs, "Mobile Eye Traking Headset," July 2019, http://pupil-labs.com.

[31] PupilLabs, "Pupil Core," July 2019, http://pupil-labs.com.

[32] T. Technology, "Tobii PCEye Mini," July 2019, http://www.tobiidynavox.com.

[33] T. Technology, "Tobii Pro Glasses II," July 2019, http://www.tobiipro.com/.

[34] T. W. Ng, "The optical mouse as a two-dimensional displacement sensor," *Sensors and Actuators A: Physical*, vol. 107, no. 1, pp. 21–25, 2003.

[35] J. Palacin, I. Valganon, and R. Pernia, "The optical mouse for indoor mobile robot odometry measurement," *Sensors and Actuators A: Physical*, vol. 126, no. 1, pp. 141–147, 2006.

[36] K. M. Hossain and A. A. Sohel, "Using optical mouse as a position feedback sensor for AGV navigation," *International Journal of Mechanical & Mechatronics Engineering*, vol. 13, no. 2, pp. 33–37, 2013.

[37] D.-H. Yi, T.-J. Lee, and D. Cho, "Afocal optical flow sensor for reducing vertical height sensitivity in indoor robot localization and navigation," *Sensors*, vol. 15, no. 5, pp. 11208–11221, 2015.

[38] F. H. Borsato and C. H. Morimoto, "Episcleral surface tracking: challenges and possibilities for using mice sensors for wearable eye tracking," in *In Proceedings of the Ninth Biennial ACM Symposium on Eye Tracking Research & Applications (ACM)*, pp. 39–46, Charleston, South Carolina, March 2016.

[39] M. Tresanchez, T. Pallejà, M. Teixidó, and J. Palacín, "Using the optical mouse sensor as a two-euro counterfeit coin detector," *Sensors*, vol. 9, no. 9, pp. 7083–7096, 2009.

[40] A. Fakki, S. Ahmed, J. Park, and C.-S. Kim, "Versatile optochemical quantification with optical mouse," *Journal of Sensors*, vol. 2017, Article ID 1243754, 7 pages, 2017.

[41] M. Tresanchez, T. Pallejà, M. Teixidó, and J. Palacín, "Measuring yarn diameter using inexpensive optical sensors," *Procedia Engineering*, vol. 5, pp. 236–239, 2010.

[42] M. Tresanchez, T. Pallejà, M. Teixidó, and J. Palacín, "The optical mouse sensor as an incremental rotary encoder," *Sensors and Actuators A: Physical*, vol. 155, no. 1, pp. 73–81, 2009.

[43] M. Tresanchez, T. Pallejà, M. Teixidó, and J. Palacín, "Using the image acquisition capabilities of the optical mouse sensor to build an absolute rotary encoder," *Sensors and Actuators A*, vol. 157, no. 1, pp. 161–167, 2010.

[44] M. Tresanchez, D. Font, M. Teixido, T. Palleja, and J. Palacin, "Preliminary study of pupil detection and tracking with low cost optical flow sensors," in *In Proceedings of the IEEE International Instrumentation and Measurement Technology Conference (I2MTC 2012)*, pp. 1251–1254, Graz, Austria, 3-16 May 2012.

[45] J. G. Daugman, "High confidence visual recognition of persons by a test of statistical independence," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 15, no. 11, pp. 1148–1161, 1993.

[46] J. Daugman, "How iris recognition works," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 14, no. 1, pp. 21–30, 2004.

[47] "ADNS-3060 High Performance Optical Mouse Sensor Datasheet," July 2019, http://www.broadcom.com.

[48] S. Kooshkestani, M. Pooyan, and H. Sadjedi, "A new method for iris recognition systems based on fast pupil localization," *Lecture Notes in Computer Science*, vol. 5072, pp. 555–564, 2008.

[49] N. Barzegar and M. S. Moin, "A new approach for iris localization in iris recognition systems," in *In Proceedings of the IEEE/ACS International Conference on Computer Systems and Applications*, pp. 516–523, Doha, Qatar, 31 March-4 April 2008.

[50] L. Components, "CAY46 plastic aspheric lens," July 2019, http://www.lasercomponents.com.

[51] OSRAM, "Opto Semiconductors, SFH 4350, High Power Infrared Emitter Datasheet," July 2019, http://www.osram.com.

[52] International Electrotechnical Commission, "EN 62471:2008, Photobiological Safety of Lamps and Lamp Systems," in *European Standard*, CENELEC, European Committee for Electrotechnical Standardization, Brussels, 2008.

[53] "ADNS-2120-001 Solid-State Optical Mouse Lens Datasheet," July 2019, http://www.broadcom.com.

[54] R. P. Wildes, "Iris recognition: an emerging biometric technology," *Proceedings of the IEEE*, vol. 85, no. 9, pp. 1348–1363, 1997.

[55] H. G. Daway, H. H. Kareem, and A. R. Hashim, "Pupil detection based on color difference and circular hough transform," *International Journal of Electrical and Computer Engineering*, vol. 8, no. 5, pp. 3278–3284, 2018.

[56] N. Amjed, F. Khalid, R. W. O. K. Rahmat, and H. B. Madzin, "Noncircular iris segmentation based on weighted adaptive hough transform using smartphone database," *Journal of Computational and Theoretical Nanoscience*, vol. 15, no. 3, pp. 739–743, 2018.

[57] N. Amjed, F. Khalid, R. W. O. K. Rahmat, and H. B. Madzin, "An improved iris segmentation technique using circular Hough transform," *Lecture Notes in Electrical Engineering*, vol. 450, pp. 203–211, 2017.

[58] D. Li, D. Winfield, and D. J. Parkhurst, "Starburst: a hybrid algorithm for video-based eye tracking combining feature-based and model-based approaches," in *IEEE Computer Society Conference in Computer Vision and Pattern Recognition (CVPR Workshops)*, pp. 79–79, San Diego, CA, USA, USA, 21-23 Sept. 2005.

[59] L. Swirski, A. Bulling, and N. Dodgson, "Robust real-time pupil tracking in highly off-axis images," in *In Proceedings of the Symposium on Eye Tracking Research & Applications (ETRA)*, pp. 173–176, Santa Barbara, California, March 28-30, 2012.

[60] M. Kassner, W. Patera, and A. Bulling, "Pupil: an open source platform for pervasive eye tracking and mobile gaze-based interaction," in *In Adjunct Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp)*, pp. 1151–1160, Seattle, Washington, September 13 - 17, 2014.

[61] A. H. Javadi, Z. Hakimi, M. Barati, V. Walsh, and L. Tcheang, "SET: a pupil detection method using sinusoidal approximation," *Frontiers in Neuroengineering*, vol. 8, p. 4, 2015.

[62] W. Fuhl, T. Kübler, K. Sippel, W. Rosenstiel, and E. Kasneci, "ExCuSe: robust pupil detection in real-world scenarios," in *Computer Analysis of Images and Patterns*, pp. 39–51, Springer, Cham, 2015.

[63] W. Fuhl, T. C. Santini, T. KŁubler, and E. Kasneci, "Else: ellipse selection for robust pupil detection in real world environments," in *In Proceedings of the Ninth Biennial ACM Symposium on Eye Tracking Research & Applications (ETRA '16)*, pp. 123–130, Charleston, South Carolina, March 14 - 17, 2016.

[64] W. Fuhl, M. Tonsen, A. Bulling, and E. Kasneci, "Pupil detection for head-mounted eye tracking in the wild: an evaluation of the state of the art," *Machine Vision and Applications*, vol. 27, no. 8, pp. 1275–1288, 2016.

[65] S. Thiago, W. Fuhl, and E. Kasneci, "PuReST: robust pupil tracking for real-time pervasive eye tracking," in *Proceedings of the 2018 ACM Symposium on Eye Tracking Research & Applications*, p. 61, Warsaw, Poland, June 14-17, 2018.

[66] A. Fogelton and W. Benesova, "Eye blink detection based on motion vectors analysis," *Computer Vision and Image Understanding*, vol. 148, pp. 23–33, 2016.

[67] K. Grauman, M. Betke, J. Lombardi, J. Gips, and G. R. Bradski, "Communication via eye blinks and eyebrow raises: video-based human-computer interfaces," *Universal Access in the Information Society*, vol. 2, no. 4, pp. 359–373, 2003.

[68] A. Królak and P. Strumiłło, "Eye-blink detection system for human–computer interaction," *Universal Access in the Information Society*, vol. 11, no. 4, pp. 409–419, 2012.

[69] Division of Electronics Engineering School of Engineering Cochin University of Science and Technology Kochi - 682022, Kerala, India, L. Pauly, and D. Sankar, "Non intrusive eye blink detection from low resolution images using HOG-SVM classifier," *International Journal of Image, Graphics and Signal Processing*, vol. 8, no. 10, pp. 11–18, 2016.

[70] S. Al-gawwam and M. Benaissa, "Robust eye blink detection based on eye landmarks and Savitzky–Golay filtering," *Information*, vol. 9, no. 4, p. 93, 2018.

[71] J. Li, "Eye blink detection based on multiple Gabor response waves," in *International Conference on Machine Learning and Cybernetics*, pp. 2852–2856, Kunming, China, 12-15 July 2008.

[72] A. Tobias, S. Thiago, and K. Enkelejda, "Brightness-and motion-based blink detection for head-mounted eye trackers," in *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing: Adjunct*, pp. 1726–1735, Heidelberg, Germany, September 12 - 16, 2016.

[73] H. Zeeshan and H. Ziaul, "Eye-blink rate detection for fatigue determination," in *2016 1st India International Conference on Information Processing (IICIP)*, pp. 1–5, Delhi, India, 12-14 Aug. 2016.

[74] M. Seki, M. Shimotani, and M. Nishida, "A study of blink detection using bright pupils," *JSAE Review*, vol. 19, no. 1, pp. 58–61, 1998.

[75] "Microchip PIC18F46J50 8-bit Microcontroller Datasheet," July 2019, http://www.microchip.com.

[76] "Universal Serial Bus, Human Interface Devices (HID) Specification," July 2019, https://www.usb.org/hid.

[77] D. Katrychuk, H. K. Griffith, and O. V. Komogortsev, "Power-efficient and shift-robust eye-tracking sensor for portable VR headsets," in *In Proceedings of the 11th ACM Symposium on Eye Tracking Research & Applications (ETRA '19)*, pp. 1–8, Denver, 2019-06.

[78] M. T. Tresanchez, A. P. Pujol, T. P. Pallejà, D. M. Martínez, E. C. Clotet, and J. P. Palacín, "An inexpensive wireless smart camera system for IoT applications based on an ARM CORTEX-M7 microcontroller," *Journal of Ubiquitous Systems and Pervasive Networks*, vol. 11, no. 2, pp. 1–8, 2019.

[79] R. J. K. Jacob, "Eye movement-based human-computer interaction techniques: toward non-command interfaces," in *Advances in Human-Computer Interaction*, Human-Computer Interaction Lab Naval Research Laboratory, Washington, D.C., 1993.

[80] H. Istance et al., "Snap clutch, a moded approach to solving the Midas touch problem," in *Proceedings of the symposium on Eye tracking research & applications*, pp. 221–228, Savannah, Georgia, March 26 - 28, 2008.

[81] S. Thiago, D. C. Niehorster, and E. Kasneci, "Get a grip: slippage-robust and glint-free gaze estimation for real-time pervasive head-mounted eye tracking," in *Proceedings of the 11th ACM Symposium on Eye Tracking Research & Applications*, pp. 1–10, Denver, CO, USA, June 2019.