EyeFlow: pursuit interactions using an unmodified camera

Almoctar Hassoumi, Vsevolod Peysakhovich, Christophe Hurter

EyeFlow: Pursuit Interactions Using an Unmodified Camera

Almoctar Hassoumi
École Nationale de l'Aviation Civile
Toulouse, France
almoctar.hassoumi@enac.fr

Vsevolod Peysakhovich
ISAE-SUPAERO, Université de
Toulouse, France
vsevolod.peysakhovich@isae-supaero.fr

Christophe Hurter
École Nationale de l'Aviation Civile
Toulouse, France
christophe.hurter@enac.fr

ABSTRACT

We investigate the smooth pursuit eye movement based interaction using an unmodified off-the-shelf RGB camera. In each pair of sequential video frames, we compute the indicative direction of the eye movement by analyzing flow vectors obtained using the Lucas-Kanade optical flow algorithm. We discuss how carefully selected flow vectors could replace the traditional pupil centers detection in smooth pursuit interaction. We examine implications of unused features in the eye camera imaging frame as potential elements for detecting gaze gestures. This simple approach is easy to implement and abstains from many of the complexities of pupil based approaches. In particular, EyeFlow does not call for either a 3D pupil model or 2D pupil detection to track the pupil center location. We compare this method to state-of-the-art approaches and find that this can enable pursuit interactions with standard cameras. Results from the evaluation with 12 users data yield an accuracy that compares to previous studies. In addition, the benefit of this work is that the approach does not necessitate highly matured computer vision algorithms and expensive IR-pass cameras.

CCS CONCEPTS

• **Human-centered computing → Interaction design**.

KEYWORDS

interaction; eye tracking;touch-free interaction; smooth pursuit;

ACM Reference Format:
Almoctar Hassoumi, Vsevolod Peysakhovich, and Christophe Hurter. 2019. EyeFlow: Pursuit Interactions Using an Unmodified Camera. In *2019 Symposium on Eye Tracking Research and Applications (ETRA '19), June 25–28, 2019, Denver , CO, USA.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3314111.3319820

1 INTRODUCTION

Gaze gestures have been studied for decades in human-computer interaction. The idea of interacting using eye movements appeared as early as 1983 [Friedman, 1983]. Recently, advances in calibration-free eye tracking interaction paved the way for more natural gaze-input modality. In particular, the introduction of pupil-target correlation produced highly accurate interaction using remote eye trackers [Pfeuffer et al., 2013] and head-worn eye trackers with smartwatches [Esteves et al., 2015], mixed reality [Esteves et al., 2017] and mobile phones [Almoctar et al., 2018, Khamis et al., 2016]. Despite these advances, pursuit interaction still requires the detection of the pupil center which may be subject to inaccuracy due to the environment, lighting, head poses and tracking devices. While remote eye trackers allow estimating user gaze with high accuracy on computer screens, they are limited to their narrow field-of-view and cannot be used with smaller surfaces (e.g., smartwatches). Eye tracking devices are used in experiments involving humans, even including infants [Franchak et al., 2011, Nasihati Gilani et al., 2018], that must not be harmed. Most pupil detection algorithms rely on head mounted eye cameras using an infrared light emitter attached next to the eye camera in order to seamlessly extract the pupil from the iris area in the imaging frame (Figure 1). The illuminating of these infrared emitters has not been shown to prevent any occasional harm on human eyes [Aly and Saad, 2011, Soderberg et al., 2016]. Additionally, the costs of these devices remain high, compared to systems based on RGB cameras. Furthermore, infrared illumination makes eye movement more detectable, but limits outdoor usage due to the interference from the strong infrared in sunlight [Zhang et al., 2017]. Therefore, instead of directly detecting the pupil center position, our approach relies solely on the global motion of the eye using an unmodified camera. However, a fundamental issue faced when using low-cost RGB cameras is that state-of-the-art pupil detection algorithms fail to detect the pupil center since the pupil/iris contrast is often too subtle and almost nonexistent for the darkest irises. In this work, we propose a method which obviates the need for detecting the pupil and runs entirely with an unmodified camera. Our approach does not require any infrared illumination or highly developed computer vision techniques. We exploit and carefully select features obtained from the motion of the eye area in the camera and investigate the global eye movement from the pixel displacement of the eye camera.

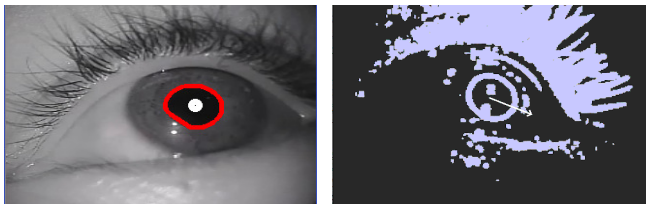

Figure 1: (Left) Pupil center detected from an IR head-worn eye tracker. (B) Using an unmodified camera, the eye movements are estimated based on the overall movement of the eye using the Lucas-Kanade optical flow algorithm.

Contribution Statement. The contribution of this paper is threefold: (1) we provide the first attempt of detecting smooth pursuit movement without explicitly detecting the pupil location, using an unmodified camera, (2) we provide technical details of detecting the

ETRA '19, June 25–28, 2019, Denver , CO, USA

ACM ISBN 978-1-4503-6709-7/19/06…$15.00
https://doi.org/10.1145/3314111.3319820

directions of the eye movement based on flow vectors obtained using an optical flow algorithm . (3) Finally, we report on a user study that investigates the accuracy of the proposed approach compared to current techniques.

2 RELATED WORKS

2.1 Smooth Pursuit Eye Movement Interaction

Pursuit interaction has become the prevailing means of interacting with moving targets.It consists of displaying and selecting moving targets by following their movement. Vidal et al. 2013 were the first to introduce this approach for public displays. Follow-up implementations were proposed by the community. In particular, Esteves et al. 2015 established a remote pursuit based interaction for smartwatches. The technique was also adopted for semaphoric gaze gesture through the Hololens [Delamare et al., 2017], augmented reality [Esteves et al., 2017, Kytö et al., 2018], eye tracking calibration [Celebi et al., 2014, Hassoumi et al., 2018] cognitive workload evaluations [Thomas et al., 2018]. In the other hand, Cymek et al. 2014 and Almoctar et al. 2018 used pursuit interaction for authentication. However, all theses approaches relied on an IR-pass eye tracking system. In contrast, EyeFlow avoids the difficulty of tracking the exact pupil location. It is noted that whilst optical flow has been used to track and limbus boundary [Turuwhenua et al., 2014], we are not aware of a paradigm using this technique for smooth pursuit movement extraction or eye tracking interaction.

2.2 Optical Flow

Optical Flow is widely used in motion detection [Yin and Shi, 2018] action recognition in static [Gao et al., 2018] and video [Choutas et al., 2018] images, 3D tracking [Wilson and Benko, 2014], etc. A close work to Eyeflow is TraceMatch [Clarke et al., 2016], which used hand or head movements to simulate remote control interaction. Features are detected using the FAST feature detector, in contrast to our approach which uses the overall indicative motion of the eye. However, a more refined matching process would be required for non-circular targets. It is therefore surprising that optical flow has attracted little attention in eye movement analysis. Therefore, it becomes natural to leverage its potential in eye movement. However, optical flow is susceptible to noise and uncertainty. This may be the reason why it has been explored for blink detection [Radlak and Smolka, 2012] and not for smooth pursuit analysis. We adopted the Lucas-Kanade (LK) Optical Flow to serve as a basis for EyeFlow since it is one of the simplest, yet most powerful algorithm to estimate pixel motions. Other methods such as SimpleFlow [Tao et al., 2012] or more complex algorithms could be used [Kroeger et al., 2016]. The LK Optical flow is a suitable trade-off between computation cost, and detection speed which is of primary importance when dealing with motion correlation. The ease of implementation makes it applicable to any application based on motion detection. In particular, Smith et al. 2000 manually select skin region in order to retrieve the eye area, and Bazyluk and Mantiuk 2014 corrected gaze fixations by tracking targets detected using optical flow algorithms. Most closely related to the present work is Radlak and Smolka's approach 2012 which leveraged optical flow to divide pixels into two groups and their respective averages are estimated to detect blink events. These previous works focused on a specific issue and are, thus, not suitable for smooth pursuit interaction. For example, the two clusters proposed by Radlak and Smolka encompass the pixels of the upper and lower eyelids. If the eye moves to the right or left, the movement cannot be tracked. EyeFlow instead allows detecting the movement of the eye in all directions. In addition, it encompasses a metric that allows detecting blink events using the entropy of vector fields.

2.3 Pupil Center Detection and Tracking

The most salient circular group of pixels in the IR eye camera is the pupil - the contracting aperture through which light penetrates the eye. There exists a great number of pupil detection methods based on highly developed computer vision algorithms [Santini et al., 2018]. As a case in point, in Starburst [Li et al., 2005], an adaptive threshold was employed on a region of interest to localize corneal reflection. Świrski et al. [2012] proposed the use of a coarse positioning algorithm using Haar-like features. Thereafter, Kassner et al. 2014 provided an eye tracking system that exploits the Canny edge detector and search for darker areas from the lowest spike in histogram. More recently, Fuhl et al. [2015] introduced ExCuSe which is based on morphologic operations and Angular Integral Projections Function to detect pupil contour. Thereafter, they proposed ElSe [Fuhl et al., 2016], a more sophisticated pupil detection algorithm which uses position refinement. However, though these aforementioned approaches supply highly accurate pupil center positions, they still appear to face the same challenges. In particular, they perform only on modified IR eye-tracking cameras. They cannot be applied on off-the-shelf cameras in view of the noise, light reflection contamination and the lack of the pupil/iris contrast.

3 OVERVIEW OF THE APPROACH

EyeFlow matches the movements of the eye with the movements of a target. The matching is operated using the overall motion of the pixels in the eye image. By virtue of the absolute position of the eye camera - i.e. the camera is rigidly attached and fixed near the user's eye - every displacement of pixels comes naturally from a motion in the user's eye area (Figure 1). Our approach is guided by the precept that global motions of eye features could replace the pupil detection in smooth pursuit interaction. Smooth pursuit presents many benefits for calibration-less interaction. The advantage of EyeFlow is that the task of detecting the pupil and its center is obviated. Thus, the approach can be used with unmodified RGB eye cameras and may be summarized as follows:

1) Capture eye images from an unmodified RGB camera and continually send the images to a PC running the algorithm.
2) Detect eye motion by leveraging changes in position of pixels using a dense optical flow algorithm. Each flow vector is assessed for validity before being used in the next steps.
3) Compute the overall ***I**ndicative-**D**irection* of the eye motion and proceed likewise for the subsequent frames.
4) Record targets positions and compute their directions.
5) Insert the eye ***I**ndicative-**D**irections* and the target directions in the ***P**osition/**I**ndicative-**D**irection* reference system.
6) As in previous studies, detect matching between the eye ***I**ndicative-**D**irections* and the target using the Pearson Correlation.

3.1 Motion Detection

Motion detection has received a sustained attention in computer vision due to its potential for various applications. In EyeFlow, the eye movement direction is extracted using the Lucas-Kanade optical flow, a differential technique which supposes that the flow motion is fundamentally constant in the local vicinity of the pixels, determined by the least square criterion. There are many optical flow algorithms, but the Lucas-Kanade is known to be fast, accurate and widely used [Patel and Upadhyay, 2013]. The pixels that change position relative to their surroundings are considered as the moving flow vectors [Royden and Moore, 2012] and convey the essential meaningful information of the eye motion. In head-worn eye tracking, the eye camera is placed so that the movements of the user's head do not affect the capturing area. More specifically, the camera always captures the eye area even if the user moves their head, acting like a fixed capturing device. As such, every pixel that changes position would most likely originate from the eye features, namely, the eyelashes, the eyelids, the skin, the pupil or the iris. As such, we restrict to the change in position of these features as the only source of information. Our implementation searches for the direction of the pixels using a dense optical flow method. For example, let's assume that a pixel $P_{t_0}(x, y, t_0)$ would have moved by Δx_0, Δy_0 at t_1 and become:

$$P_{t_1}(x + \Delta x_0, y + \Delta y_0, t_1)$$

then have moved by Δx_1, Δy_1 at t_2 and the expression became:

$$P_{t_2}(x + \Delta x_0 + \Delta x_1, y + \Delta y_0 + \Delta y_1, t_2)$$

the vectors $\overrightarrow{P_{t_0}P_{t_1}} = (\Delta x_0, \Delta x_0)$ and $\overrightarrow{P_{t_1}P_{t_2}} = (\Delta x_1, \Delta x_1)$ will give two different records about the displacement of the pixel and its direction. During an eye movement, many pixels move and provide a set of local direction which are aggregated and transformed into a single direction (the ***I**ndicative **D**irection*). Namely, the set of the pixels that moved are used to calculate the principal direction of the eye. A naive approach would be to store and consider all the detected flow vectors. However, unexpected events such as small camera drift enable the appearance of noisy displacements at the edge of the frame. Our method constantly detects and removes them during the outliers removal phase as shown in Figure 6.

3.2 Eye Area Features Movement

A key benefit of head-mounted eye trackers is that the detected motions are limited to either the movements of the eye, eyelids or eyelashes. The motion of the pupil and the iris contributes the most to the set of detected flow vectors. When a user follows a moving object, his pupil moves in the same direction. Consequently, the iris which encompasses the pupil proceeds in a similar fashion. As a result, the iris and the pupil form a compact large object that can trigger the appearance of flow vectors. However, due to the physiological functions of the eye region, movements of the eye naturally actuate motions of the eyelids and, consequently, the eyelashes which grow at their edges. This is especially observed when the user moves their head upwards and downwards (Figure 2). As the eye moves away from its initial position, new flow vectors are detected and correspond to a change in position of some pixels. These flow vectors appear whenever the following conditions are met: Pixels have moved since the last frame and the displacement has exceeded a predefined threshold.

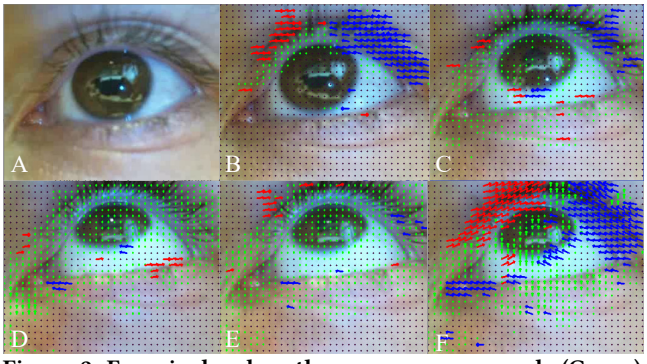

Figure 2: Eye pixels when the eye moves upwards (Green). The blue (red) pixels show the pixels moving to the left (right). The average vector will result in an upward direction, due to the large number of green pixels and the opposing blue and red directions.

3.3 Computing the *Indicative Directions*

Given a set of flow vectors obtained from two successive eye camera frames, the ***I**ndicative **D**irection* of the eye motion can be calculated. Each flow vector yields a vector $\vec{I_i}$ whose length depicts its magnitude $\|\vec{I_i}\|$ and its direction gives the orientation θ_i of the motion of that single flow vector. In order to extract the ***I**ndicative **D**irection* θ_t of the eye motion, a specialized selection process is performed at every single flow vector. The selection process seeks the flow vectors that most likely appear due to the actual motion of the eye and rejects the flow vectors considered as outliers. The selected flow vector directions are then recorded and the average direction (Figure 3) is stored as the ***I**ndicative **D**irection* of the eye for the two successive frames. Thus, every pair of frames yields one main eye motion direction with an angle of:

$$\theta_t = \frac{1}{k}\sum_{i=1}^{k}\alpha_i \quad , k \leqslant n$$

Where θ_t is the angle between the main eye motion $\vec{V_i}$ and the x-axis. During the smooth pursuit eye movement, a set of ***I**ndicative **D**irections* $[\theta_1, \theta_2, \ldots, \theta_p]$, is stored to calculate the correlation with the target positions.

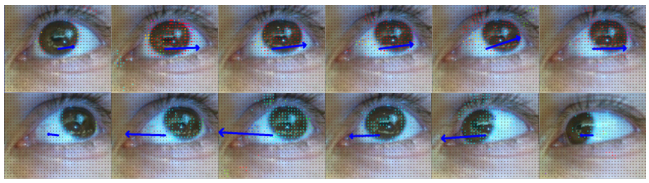

Figure 3: Left-to-Right and Right-to-Left directions.

3.4 From Flow Vectors to 1D Time-Series

Let V be a collection of the indicative vectors that describes the overall movements of the eye. V is then a series of 2-dimensional mean vectors $(\overrightarrow{V_i} = (\|V_i\|, \theta_i))$ that defines each single movement of the eye in the eye camera frame between time t_1 to t_n:

$$V = [\vec{V_1}, ..., \vec{V_n}] = [(\|\vec{V_1}\|, \theta_1), ..., (\|\vec{V_n}\|, \theta_n)]$$

Where $\|\vec{V_i}\|$ is the magnitude and θ_i its angle relative to the positive x-axis $\vec{i}$ ($\theta_i = \angle(\vec{V_i}, \vec{i})$). The angles θ_i coupled with their position i are inserted into a ***P**osition/**I**ndicative-**D**irection* reference system (PID) and, thus, converted into time series. Figures 4 and 5 show the PID space. In the same vein, the spatial 2D coordinates of the target are transformed into a set of positions/directions and inserted into the PID reference. Oscillations around the angle 180° may appear when the eye moves to the right. This is due to the fact that when the eye moves to the right, corresponding to angles of $\approx 180°$, the small eye drifts give angles of $\approx -180°$, resulting in the appearance of fluctuations in the PID. This could be reduced using iterative data normalization [Vlachos et al., 2004] and filtering [Casiez et al., 2012].

3.5 Eyelids/Eyelashes and eye movement

The appearance of potential flow vectors is estimated at each pixel of the eye images (Figure 6b). It may suffice to consider all flow vectors if only the pupil and the iris move, but since we consider the eye movement as a whole, flow vectors obtained from the motion of the eyelids and eyelashes are likely to appear. In particular, when the eye moves upwards and downwards, the eyelids/eyelashes move accordingly (Figure 2). We observed that the motions of the eyelid/eyelash flow vectors are correlated to the movements of the eye. This is less obvious when the eye moves in the left/right directions. Figure 2 shows an upward movement of the eye. We can observe that, the eyelash left-region flow vectors move Up-Right, the center flow vectors move upwards, the eyelash right-region flow vectors move Up-Left and the pupil and the iris move upwards. Integrating the flow vectors of the eyelids/eyelashes does not have an adverse effect on the calculation of the eye movement because there is a compensation of the flow vectors and the ***I**ndicative **D**irection* would likely be directed upwards. We observed that when the eye moves in the left/right direction, the flow vectors of the eyelids/eyelashes tend to jitter and move in every direction. This is removed using a bilateral filter which blurs the image and reduces noise, making the iris/pupil more visible as shown in Figure 6a.

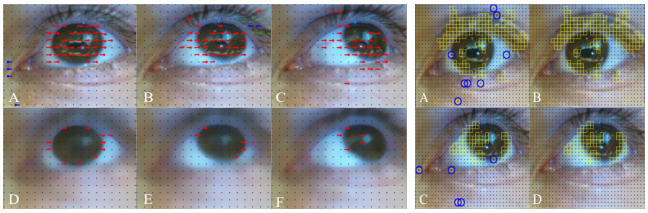

Figure 6: (a) The high-frequency noise is reduced using a bilateral filter. The edges of the iris are preserved. Classical filters such as the Mean and Gaussian filters could reduce noise but also blur the edges. b) Example of outliers not appertaining to a group of pixels.

3.6 Flow Vectors Filtering

Blink Detection: Blinking and winking could produce a larger number of flow vectors. Winking is a voluntary eye movement and can be avoided. Blinking cannot be obviated since it is a natural bodily function. It particularly helps to refocus the eye and protect the eye from external irritants. This is captured by analyzing the flow field of the eyelash/eyelid in the images. The upper and lower lashes generate clusters of opposing flow vectors indicating the presence of a blink event.

Pixel Displacement and Flow Vectors: The pixels that do not exhibit a sufficient displacement are discarded. If the displacement $d_i = \sqrt{x_i + y_i}$ of the flow vector $v_i = (x_i, y_i)^\intercal$ is smaller than the size of 2 pixels (chosen empirically), v_i will not be recorded, since we consider that the change in position of the pixel is not considerable enough to be taken into account. Nevertheless, the remaining flow vectors still contain many resilient outliers.

Outliers Removal and Consecutive Likelihood Preserving. : we need to evaluate the likelihood that a pixel (or a cluster) is a possible candidate of the overall motion in order to estimate the current ***I**ndicative **D**irection*. For example, a pixel heading in the opposite direction of the majority of the flow vectors is most likely an outlier and, thus, could be removed from the dataset. Our approach eliminates any flow vector whose direction is not in the interval defined by $\boldsymbol{\mu} \pm 2\boldsymbol{\sigma}$, where $\boldsymbol{\mu}$ is the mean and $\boldsymbol{\sigma}$ the standard deviation of the flow vector directions. Another consideration could be to reduce their weight (impact) when calculating the ***I**ndicative **D**irection*. As an example, Figure 7 depicts a set of 50×50 flow vectors. One can easily notice the presence of three outliers in the dataset. These vectors are not in the interval $[\mu - \sigma, \mu + \sigma]$.

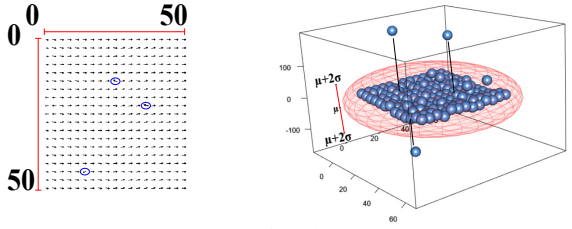

Figure 7: Outliers removal. (Left) 50×50 **flow vectors. All vectors have a direction** $\in [-5°, 5°]$**, indicating an overall motion to the right, except 3 vectors that have a direction of** $178°$**,** $155°$ **and** $-175°$ **(blue ellipse). (Right) The area defined by** $[\mu - \sigma, \mu + \sigma]$ **indicates that the three vectors are not potential candidates of the overall motion.**

3.7 Frame Abandoning

When calculating the sequence of ***I**ndicative **D**irections*, if the current *pair of consecutive frames* exhibits pixels heading in all directions, the older frame is skipped and the ***I**ndicative **D**irection* is not computed. In other words, each *pair of consecutive frames* is checked for consistency. This is investigated by considering all pixels x_i that moved as a sequence of a discrete random variable $X = [x_1, x_2, ..., x_n]$ and measuring the uncertainty using the Shannon's entropy defined as follows:

$$E(x) = \sum_{i=1}^{n} p(x_i) \log_2 \frac{1}{p(x_i)}$$

where $p(x_i)$ is the probability of the direction of a pixel x_i. For example, let us consider four pixels x_1, x_2, x_3 and x_4 with the following directions represented as polar angles $A = [x_1 = \pi, x_2 = \frac{\pi}{2}, x_2 =$

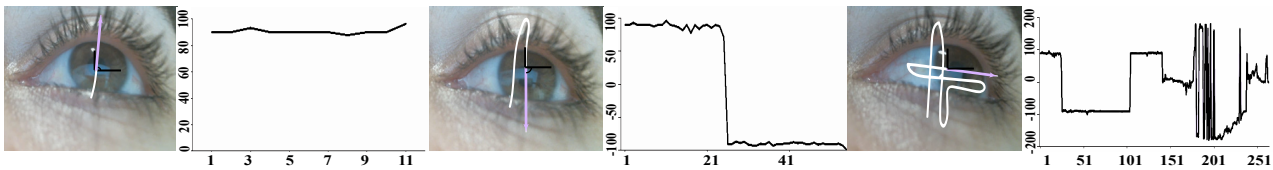

Figure 4: Directions in the *Position/Indicative Direction* reference system. the white stroke in each eye image shows the followed trajectory. Left-the eye moves upwards ($\approx 90°$), and then downwards ($\approx -90°$). At the positions 180 to 201, oscillations appear in the recorded data, since when the eye moves to the right corresponding to $\approx 180°$, the small eye drifts and the uncertainty of the calculations give angles of $\approx -180°$, resulting to the appearance of fluctuations around the angle $180°$.

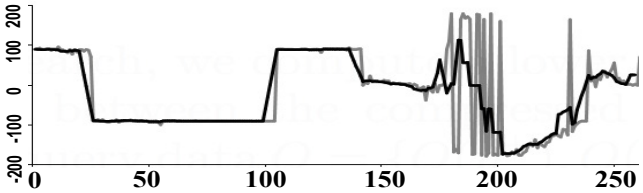

Figure 5: Smoothing the vector directions considerably reduces the oscillations in the original data (Grey) while maintaining the trend in the series (Black).

$\frac{\pi}{4}, x_4 = \frac{2\pi}{3}]$. The elements x_i have different directions. The probability of appearance of each element x_i in the list can easily be calculated, i.e. $P(x_i) = \frac{1}{4}$. In such case, the entropy $E(A) = \log_2(4) = 2$ is high and this is the situation of maximum uncertainty where the exact ***I**ndicative **D**irection* cannot be predicted. Now, let us consider a discrete variable $B = [x_1 = \pi, x_2 = \frac{2\pi}{3}, x_3 = \pi, x_4 = \pi]$. In this set, there are three pixels (x_1, x_3, and x_4) that have the same direction (π). The variable B can be transformed into a new variable $\hat{B} = [\hat{x}_1, \hat{x}_2]$ where $\hat{x}_1 = \{x_1, x_3, x_4\}$ and $\hat{x}_2 = \{x_2\}$. The probability of appearance of an element of $\hat{x}_1$ and $\hat{x}_2$ are respectively $\frac{3}{4}$ and $\frac{1}{4}$. The entropy of B is then $E(B) = \frac{3}{4}\log_2\frac{4}{3} + \frac{1}{4}\log_2 4 = 0.811 < E(A)$, indicating a low uncertainty. In this situation, the ***I**ndicative **D**irection* could be calculated. Note that in the best case scenario where all moving pixels head toward the same direction, there is no uncertainty and the entropy $E(X) = 0$. Overall, the ***I**ndicative **D**irection* is calculated if the entropy is less than a threshold. We define the threshold as $t_E = \frac{1}{2}[1 + \log_2(n)]$ corresponding to the entropy where at least half of the flow vectors head toward the same directions. EyeFlow algorithm is described in Algorithm 1.

Algorithm 1: EyeFlow

```
Data: Successive frames f_1, f_2, ..., f_n
Result: an Array Of Indicative Directions
begin
    Initialize
    ArrayOfIndicativeDirections
    for i = 1; i < n; i = i + 1 do
        flowFields ⟵ computeOpticalflow(f_i, f_{i+1})
        c ⟵ count(flowFields) /* Number of vectors */
        t_E = 1/2 [1 + log_2(c)]
        e ⟵ computeEntropy(flowFields)
        if e < t_E then
            IndicativeDirection ⟵
              IndicativeDirection(flowFields)
            ArrayOfIndicativeDirections.Add(IndicativeDirection)
        end
    end
end
```

4 EVALUATION

4.1 Replacing Pupil Center with flow vectors

We argue that tracking the pupil center location is more challenging in environments where lighting conditions are not adequate than tracking the motion of the eye as a whole. Most pupil detection methods rely on different image processing phases which are intimately dependent on the capturing environment (e.g. image segmentation and thresholding). By contrast, this is not the case when considering the global motion of the eye. Moreover, there are a great number of frames where the user does not move their eyes. While most of the existing methods will waste their time tracking the pupil center and providing redundant information about the pupil center location frame after frame, our approach gives new values only if a motion is detected, thus reducing redundancy and data processing time. In order to evaluate the applicability of our approach, we compared the set of the ***I**ndicative **D**irections* θ obtained from EyeFlow to the standard method (pupil center based). For this purpose, we needed to transform the pupil center positions in a reference that is comparable to EyeFlow, namely the PID Reference. AS such, a pupil center position is extracted in each frame of the eye camera. The vector of two successive pupil center locations at time instants t_{i-1} and t_i is defined as: $\vec{P_i} = \vec{p_i} - \vec{p_{i-1}}$. A collection P of pupil center directions can, thus, be extracted and stored for comparison with the collection of vectors V obtained from EyeFlow.

$$P = [\vec{P_1}, ..., \vec{P_n}] = [(\|\vec{P_1}\|, \theta_1), ..., (\|\vec{P_n}\|, \theta_n)]$$

Where $\|\vec{P_i}\|$ is the vector magnitude and θ_i its angle with the positive x-axis $\vec{i}$ ($\theta = \angle(\vec{P_i}, \vec{i})$). Therefore, we obtain two collections of *position/**I**ndicative **D**irection*, one from EyeFlow directions and the other from the pupil center directions. The results of a pilot study showed that the two sets of data were highly correlated, meaning that both approaches are suitable for pursuit interaction. To the best of our knowledge, the two most popular methods for interacting with a system using smooth pursuit eye movement are those of Vidal et al. 2013 and Esteves et al. 2015. To analyze our approach, we have conceived a desktop application. Our approach is evaluated on eight different targets displayed on a screen (Figure 8): The *circular* target proposed in Orbits [Esteves et al., 2015], the

delete and *zig-zag* targets of $1 Unistroke Recognizer [Wobbrock et al., 2007], the *spiral* used for unsupervised calibration in [Santini et al., 2017] and [Hassoumi et al., 2019], a custom *alpha* we add to investigate the robustness of the algorithm when comparing two approximately similar targets (*delete* and *alpha*), the *rectangular* target used in pursuit calibration [Pfeuffer et al., 2013] and two digit targets ② and ③ recently proposed in Pa*th*Word for smooth pursuit PIN-entry [Hassoumi et al., 2018]. The targets are very diverse. The *circular* and the two digit targets were used for eye-based interaction, the *spiral* and *rectangular* targets were used for eye tracking calibration whereas the *delete* and *zig-zag* served for stroke recognition. In this work, we use the shapes of these targets as trajectories for the moving stimuli. The goal is to ascertain that the algorithms of previous studies can leverage the potential of EyeFlow, i.e. using a normal camera for smooth pursuit interaction. The application recognizes among different targets, the one the user intends to select. Following Almoctar et al. [2018] and Esteves et al. [2015], a stimuli moves on the paths defined by each symbol and the user is requested to select the target by fixating on the stimuli moving on its trajectory. This application demonstrates the ability of our approach to leverage the global motion of the user's eye.

Figure 8: Different targets used in the evaluation.

Definition 1: A *pupil motion series* $\boldsymbol{P}$ and a *target motion series* $\boldsymbol{T}$ are series of real-valued angles α_i and $\beta_i \in$ [-180,180] : $\boldsymbol{P} = \alpha_1, \alpha_2, ..., \alpha_n$ and $\boldsymbol{T} = \beta_1, \beta_2, ..., \beta_n$ where n is the length of the checking window. The figure 9 below depicts an example of a *pupil motion series.*

Figure 9: The *pupil motion series* obtained from a user who was following a stimuli moving on a rectangular trajectory.

Definition 2: The $\boldsymbol{Targets}$ define the set of targets displayed on the screen (T_1, T_2,..., T_m) where m is the number of targets and T_i corresponds to the *target motion series* of the $i-th$ target. Note that the similarity is calculated between the *pupil motion series* $\boldsymbol{P}$ and each *target motion series* $\boldsymbol{T_i}$. Figure 10 exemplifies different targets and their corresponding *target motion series.*

Figure 10: Different 2D targets and their corresponding representation in the *motion series* space.

Definition 3: ***S**imilarity **M**easure:* given two motion series **A** and **B** of length n, the *Similarity Measure* c(**A**,**B**) is a function which returns the degree of similarity between the two sequences. We used the correlation coefficient, an intuitive similarity measure proposed in previous works [Esteves et al., 2015, Hassoumi et al., 2018]. The measure is a real value $\in [-1, 1]$ with 1 indicating a high correlation. A threshold t indicates the minimum ***S**imilarity **M**easure* value in order to validate the similarity of the two *motion series*. However, we observed that relying solely on this metric could induce misinterpretation issues. As an example, consider the three curves (C_1, C_2, and C_3) below (Figure 11). While the correlation result indicates a high similarity, the strokes are quite different. In particular, the correlations between C_1 and C_2 on x- and y-axes are respectively $r_x = 0.99$ ($p < .001$) and $r_y = 0.77$ ($p < .001$). The correlations are even higher when we compare C_1 and C_3 ($r_x = 0.99, p < .001$ and $r_y = 0.95, p < .001$). Taking into account all these issues, we considered an additional step to validate the result after a high ***S**imilarity **M**easure* is found. We used the Dynamic Time Warping (DTW) which was used for comparing trajectories [Cai et al., 2019], hand gestures [Tang et al., 2018], time series [Rakthanmanon et al., 2012], shapes [Zhao and Itti, 2018], accelerometer [Akl and Valaee, 2010], voice [Muda et al., 2010], etc. For an overview of the metric, we refer the prospective reader to the works of [Rakthanmanon et al., 2012]

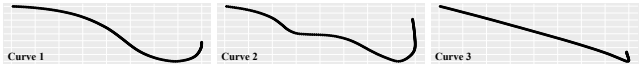

Figure 11: Three different strokes with high correlations.

The following studies reports the evaluation of EyeFlow accuracy. We first investigate the robustness of the approach. Then, we perform a comparison between different target sizes. Finally, we compare EyeFlow with state-of-the-art methods on smooth pursuit interaction. The default values of the parameters are the *checking window size = 200 ms*, *target size = L_2* and *flow window size =* 5× *pixels* unless otherwise specified.

4.2 Participants, Apparatus and Design

12 users' ground truth eye videos were recorded for the experiment. One participant wore contact lenses. A 24 inch Dell 2408WFP screen[1] which has a resolution of 1920x1200 pixels and 24 millisecond response time, running on a 64 bits[2] XPS 15 Dell Laptop , 16GB RAM was used. The Pupil Labs headset IR-pass eye camera was replaced with a full HD Logitech 615 camera (@60Hz, 720 × 480). A world camera was not necessary since only eye movements are captured. In addition, the approach does not require prior calibration. Participants sat ~75 cm from the screen. A controlled lab in the local institution was set up for the experiment. All participants were subjected to the same task. The participant was asked to select an assigned target, randomly selected among the ones presented in Figure 8, by following its moving stimuli. We arranged a scenario in which only a RGB video of the eye movement is captured. The analysis was performed afterward.

4.3 Procedure

An acquaintance time was given to participants in order to become familiar with the system. The task began after the detailed procedure has been explained. To reduce fatigue effects, the participants were asked to take a pause of one minute after each trial. 72 trials were recorded (3 Moving target speeds × 2 target selections × 12 participants). The experiment spanned a day and lasted ≈ 15 minutes for each participant. The order of speeds is chosen randomly to reduce learning effects.

[1] L x W x H Dimensions: 22 x 8.17 x 15.62 inches
[2] Intel(R) Core(TM) I7-4712HQ CPU @ 2.30GHz,2301 MHz, 4 core(s), 8 process

4.4 Evaluation 1: Accuracy

We start by evaluating the accuracy of EyeFlow. This evaluation proceeds similarly to Esteves et al. [2015] and Delamare et al. [2017]. It aimed to assess the effectiveness and accuracy of the proposed approach based on the number of correct recognition. The experiment has two goals. First, we explore the accuracy and robustness of the proposed approach. Second, we evaluate three different speed configurations based on recognition rate, in order to determine to which extent the speed levels alter or impact the accuracy of the proposed approach.

The study was structured as a within-subjects design with **Moving target Speed** as the independent variable. Three levels of speed were considered. The moving stimulus moved at slow (6°, 7.8 cm, 294.80 pix)/s, medium (10°/s, 13.22 cm, 499.65 pixel/s) and fast (14°/s, 18.69 cm, 706.39 pixel/s) speeds. The order of the conditions was counterbalance with a partial Latin-square. The dependent variables were the number of *correct* and *false* activations. Each participant was requested to select the target (the "alpha" shape) twice. Consequently, each participant performed 3 Moving target speeds × 2 targets = 6 trials, for a total of 6 × 12 participants = 72 records.

Results: EyeFlow Accuracy. The ***I**ndicative **d**irections*, which only describe the global motion of the eyes, produce good results (87.5% recognition rate). This supports the importance of flow vectors for eye movement retrieval. Of the recorded videos, two were removed due unwanted eye camera movements. Of the rest, the data recording lasted ≈ 5 seconds. Consequently, EyeFlow is suitable for smooth pursuit based interaction which is in line with our expectations. Descriptive statistics (Figure 12) showed that on average, the detection rates were 96%, 92% and 75% respectively for the low (6°, 7.8 cm, 294.80 pix/s), medium (10°/s, 13.22 cm, 499.65 pixel/s) and fast (14°/s, 18.69 cm, 706.39 pixel/s) speed moving stimuli. Using an ANOVA ($\alpha = .05$), we found no significant effect of *Moving target speed* on detected targets ($F(2, 69) = 2.760, p = .07$). Post-hoc analysis with Bonferroni test showed that the comparisons between the level of speeds were not significantly different ($p_{s_1,s_2} = 1.00$, $p_{s_1,s_3} = .089$, $p_{s_2,s_3} = .24$).

Figure 12: Mean recognition rate for the three speed configurations.

4.5 Evaluation 2: The Effect of Parameters

We investigate the effect of two parameters on the recognition rate:

- **3 target sizes** (L_1 = 7×7cm^2, L_2 = 14×14cm^2, L_3 = 21×21cm^2)
- **4 inter-flow spaces** (5×5, 10×10, 15×15, 20×20 pixels.)

The inter-flow space is the space between consecutive vectors, namely, the minimum number of pixels between two flow vectors. We set the moving stimuli speed to 10°/s and the *checking window* to 500 ms for all conditions. The variation of the parameters was organized through 12 blocks (**3 target sizes** × **4 inter-flow spaces**). Figure 13 illustrates the different sizes used in this evaluation. With all the parameters set to their default values, we investigate the effect of minimum space between consecutive flow vectors that influences the detection of the eye movement. For example, if we abusively consider that the *minimum inter-flow space* is 30 pixels. The very small eye movement < 28 pixels would barely be detected.

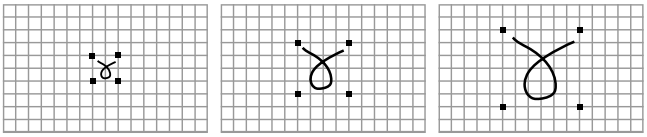

Figure 13: Different target sizes (small, medium and large) used in the experiment. One target (@) is represented for simplicity purpose. This variable was used to assess whether the approach could detect a stimuli moving on a different trajectory sizes.

Results: The Effect of Target Size and inter-flow space. Overall, the targets L_3 outperforms L_2 and L_1 by 6% and 27% respectively. The lower results of L_1 compared to the two others may be explained by the small pixel displacement of the eye movement. Notice that in section 3.6, EyeFlow discards all vector flow whose change in position is smaller than a predefined threshold. For all targets, we observe an increase of the accuracy until L_3 since longer trajectories produce more eye movement pixel displacement. The results show that reducing the size of the inter-flow space improves the detection rate. In particular, a two-way ANOVA showed a significant main effect of *Target Size* on detection rate ($F(2, 144) = 4.394, p < .05$). No significant effect was found for *Inter-flow space* ($F(3, 144) = .379, p = .768$) and no *Target Size* × *Inter-flow Space* interaction effect was found ($F(6, 144) = .095, p = .997$). However, the frame rate decreases with smaller inter-flow spaces. We observe that the checking window size is an unstable parameter. For example, a checking window of 0.1 s as in [R and David, 1991, Vidal et al., 2013] would fail to detect a target whereas changing the size to 0.2 s provides high accuracy. The minimum smooth pursuit time is 0.1 s [Vidal et al., 2013]. This corresponds to only 6 frames using a camera at 60 frames per second, and it is a very small number for calculating correlation. The highest results are achieved when the size is greater than 0.3 s (> 300 ms).

4.6 Evaluation 3: Comparison

We compare EyeFlow with Pupil Labs' pupil center detection [Kassner et al., 2014]. We consider Pupil Lab since many smooth pursuit based interaction relies on its algorithm to record the pupil centers [Almoctar et al., 2018, Esteves et al., 2015, 2017]. It is worth noting that previous works compute two correlations per target, on x- and y-axis separately, resulting in two correlation values. In our approach, we use the *Motion series* of the eye and the target, hence, we only calculate one correlation in a single pass. An alternative approach would be to use the RV coefficient which is a multi-dimensional approach of the Pearson correlation. Therefore, to make the comparison fair and not misleading, we compute the vectors between each consecutive pupil centers position obtained from the IR camera of Pupil Labs eye tracker, and record it as the ***I**ndicative **D**irection*. We can therefore create a *Motion series* M_1 from the pupil centers and compare it to EyeFlow *Motion series* M_2. For each task, a target is displayed on the screen and the participant is asked to follow it moving stimuli. The independent variables were:

- **Target type**: The 8 targets described in Figure 8
- **Camera/algorithm type**: *Pupil Labs* and *RGB cameras*

Consequently, eye movement videos of 8 targets × 2 cameras × 12 participants were collected for a total of 192 records.

Results: Comparison with state-of-the-art. A repeated-measures ANOVA (α = .05) was conducted to analyze the results. We used Greenhouse-Geisser corrected values when the assumption of sphericity was violated (Mauchly's test). Pairwise t-tests with Bonferroni corrections were used for the post-hoc test. There was no significant main effect on detection rate for Camera/algorithm ($F(1, 192) = .310$, $p = .578$) and Target type ($F(7, 192) = 1.414$, $p = .202$). No interaction effect of *Camera/algorithm* × *Target type* was found ($F(7, 192) = .468$, $p = .857$). As expected, A RGB camera could be an alternative for smooth pursuit interaction using EyeFlow (mean detection = 83%, Figure 14). We also caluclated the DTW between the two *Motion series* M_1 and M_2. The results showed a small distance measure for each type of target. For example, the distance between the mean EyeFlow motion series and the mean pupil based approaches motion series is 4.77 for *delete* and the highest measure is 8.5. For reference, the distance between the *Motion series* of curve 1 and 2 of Figure 11 is 5.3 and 4.5 between curve 1 and 3.

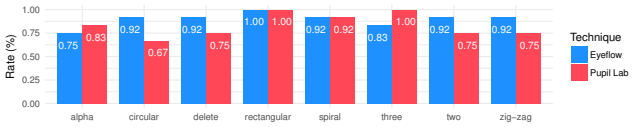

Figure 14: Mean recognition rate for the three speeds.

5 DISCUSSION AND LIMITATIONS

While EyeFlow does not call for a specific eye model, a modified camera, or complex pupil center detection algorithms, it requires the camera to be rigidly attached to the user's head as for head-worn eye trackers. Using a remote eye tracker, the eyes appear still or exhibit hardly noticeable movements when following small targets. However, recent advances on motion magnification can help to increase the movements of the eye in a video [Rubinstein et al., 2013].

Implications for design choices could be deduced from the results of the evaluations. The speed of the moving stimuli significantly impacts the recognition rate. We noted that at high speed, the moving stimuli compelled the users to make saccadic eye movement to stabilize the stimuli in the fovea. We would recommend the use of stimuli moving at low since, the eyes are able to follow the stimuli seamlessly, however, some users found it tiring.

In addition, we experienced more blinking in such conditions. Smooth pursuit interaction is known to cause issues due to the similarity of targets. EyeFlow is not an exception as the flow vectors cannot differentiate when the user follows similar targets (e.g. the "delete" and "alpha" targets). The highest accuracy is found when the selected target is unique among the others. This result implies that, when designing a system using Eyeflow, it would be more interesting to consider targets with very different shapes or similar targets having their stimuli moving at different phase as in [Esteves et al., 2015].

Moreover, there are a number of additional processing steps that may be worth incorporating in the algorithm for higher precision. For instance, skin-detection algorithms can help to remove pixels that are likely to not appertain to the iris/pupil area. We envision integrating retargeting and coarse estimation which provide a powerful means for eye area segmentation.

EyeFlow may be useful in a number of situations where it is worthwhile using eye tracking systems without resorting to IR light illumination. For example, for participants who do not tolerate LED illumination close to the eyes or for studies involving infants. Additionally, it can be used for different applications, for example, to detect reading events without calibration (Figure 15). Different approaches already exist but this can be an interesting research question for future works.

Among different resolutions, 320×240 presented good correlation results using EyeFlow. Lower resolutions (e.g., 120 × 90) are the most efficient. However, they do not allow extracting sufficient features for optical flow. Higher resolutions are slower, likely due to the high number of pixels (388,800 for 720 × 540). They allow extracting a large number of features. Therefore, the computational time increases and the quality of the correlation is not improved since the highest the resolution, the more features and consequently the more outliers are present in the flow fields.

Figure 15: Reading events detected using EyeFlow.

Our approach cannot be used for fixation-based applications since it does not allow obtaining the 2D pupil center position.

6 CONCLUSION

This paper has presented an alternative to pupil center location previously used for smooth pursuit interaction for head-worn eye trackers using unmodified cameras. Rather than tracking the pupil with 3D models or 2D complex computer vision approaches as used in often-cited applications, we consider an alternative approach that focuses only on the movement of the eye as a whole.

Since EyeFlow matches eye movement and the target trajectory, it is no longer necessary to know the exact pupil center location. Instead, EyeFlow makes an assumption that the change in position of pixels in the eye image provides a clue to the direction of eye movement. These eye movement directions suffice to know the target the user is following by matching their movements. This paper details the approach which uses the Lucas-Kanade optical flow algorithm to detect motion flow vectors with an unmodified camera.

The results of a study showed that the approach was able to detect targets reliably (87.5% recognition rate). While there are a number of scenarios to improve this approach for remote and mobile eye tracking devices, we believe that EyeFlow is a promising advance towards affordable eye tracking systems.

REFERENCES

A. Akl and S. Valaee. 2010. Accelerometer-based gesture recognition via dynamic-time warping, affinity propagation, amp; compressive sensing. In *2010 IEEE International Conference on Acoustics, Speech and Signal Processing*. 2270–2273. https://doi.org/10.1109/ICASSP.2010.5495895

Hassoumi Almoctar, Pourang Irani, Vsevolod Peysakhovich, and Christophe Hurter. 2018. Path Word: A Multimodal Password Entry Method for Ad-hoc Authentication Based on Digits' Shape and Smooth Pursuit Eye Movements. In *Proceedings of the 20th ACM International Conference on Multimodal Interaction (ICMI '18)*. ACM, New York, NY, USA, 268–277. https://doi.org/10.1145/3242969.3243008

Eman Aly and Eman Saad. 2011. Effect of infrared radiation on the lens. *Indian journal of ophthalmology* 59 (03 2011), 97–101. https://doi.org/10.4103/0301-4738.77010

Bartosz Bazyluk and Radosław Mantiuk. 2014. Gaze-Driven Object Tracking Based on Optical Flow Estimation. In *Computer Vision and Graphics*, Leszek J. Chmielewski, Ryszard Kozera, Bok-Suk Shin, and Konrad Wojciechowski (Eds.). Springer International Publishing, Cham, 84–91.

Qiqin Cai, Lyuchao Liao, Fumin Zou, Subin Song, Jierui Liu, and Meirun Zhang. 2019. Trajectory Similarity Measuring with Grid-Based DTW. In *Advances in Smart Vehicular Technology, Transportation, Communication and Applications*, Yong Zhao, Tsu-Yang Wu, Tang-Hsien Chang, Jeng-Shyang Pan, and Lakhmi C. Jain (Eds.). Springer International Publishing, Cham, 63–72.

Géry Casiez, Nicolas Roussel, and Daniel Vogel. 2012. 1€ Filter: A Simple Speed-based Low-pass Filter for Noisy Input in Interactive Systems. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '12)*. ACM, New York, NY, USA, 2527–2530. https://doi.org/10.1145/2207676.2208639

Feridun M. Celebi, Elizabeth S. Kim, Quan Wang, Carla A. Wall, and Frederick Shic. 2014. A Smooth Pursuit Calibration Technique. In *Proceedings of the Symposium on Eye Tracking Research and Applications (ETRA '14)*. ACM, New York, NY, USA, 377–378. https://doi.org/10.1145/2578153.2583042

Vasileios Choutas, Philippe Weinzaepfel, Jérôme Revaud, and Cordelia Schmid. 2018. PoTion: Pose MoTion Representation for Action Recognition. In *CVPR 2018 - IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, Salt Lake City, United States, 1–10. https://hal.inria.fr/hal-01764222

Christopher Clarke, Alessio Bellino, Augusto Esteves, Eduardo Velloso, and Hans Gellersen. 2016. TraceMatch: A Computer Vision Technique for User Input by Tracing of Animated Controls. In *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp '16)*. ACM, New York, NY, USA, 298–303. https://doi.org/10.1145/2971648.2971714

Dietlind Helene Cymek, Antje Christine Venjakob, Stefan Ruff, Otto Hans-Martin Lutz, Simon Hofmann, and Matthias Roetting. 2014. Entering PIN codes by smooth pursuit eye movements. *Journal of Eye Movement Research* 7, 4 (2014). https://bop.unibe.ch/JEMR/article/view/2384

William Delamare, Teng Han, and Pourang Irani. 2017. Designing a Gaze Gesture Guiding System. In *Proceedings of the 19th International Conference on Human-Computer Interaction with Mobile Devices and Services (MobileHCI '17)*. ACM, New York, NY, USA, Article 26, 13 pages. https://doi.org/10.1145/3098279.3098561

Augusto Esteves, Eduardo Velloso, Andreas Bulling, and Hans Gellersen. 2015. Orbits: Gaze Interaction for Smart Watches Using Smooth Pursuit Eye Movements. In *Proceedings of the 28th Annual ACM Symposium on User Interface Software Technology (UIST '15)*. ACM, New York, NY, USA, 457–466. https://doi.org/10.1145/2807442.2807499

Augusto Esteves, David Verweij, Liza Suraiya, Rasel Islam, Youryang Lee, and Ian Oakley. 2017. SmoothMoves: Smooth Pursuits Head Movements for Augmented Reality. In *Proceedings of the 30th Annual ACM Symposium on User Interface Software and Technology (UIST '17)*. ACM, New York, NY, USA, 167–178. https://doi.org/10.1145/3126594.3126616

John Franchak, Kari Kretch, Kasey C Soska, and Karen Adolph. 2011. Head-Mounted Eye Tracking: A New Method to Describe Infant Looking. *Child development* 82 (11 2011), 1738–50. https://doi.org/10.1111/j.1467-8624.2011.01670.x

Mark B Friedman. 1983. Eyetracker communication system. In *Computer Applications in Medical Care, 1983. Proceedings. The Seventh Annual Symposium on*. IEEE, 895–896.

Wolfgang Fuhl, Thomas Kübler, Katrin Sippel, Wolfgang Rosenstiel, and Enkelejda Kasneci. 2015. ExCuSe: Robust Pupil Detection in Real-World Scenarios. In *Computer Analysis of Images and Patterns*, George Azzopardi and Nicolai Petkov (Eds.). Springer International Publishing, Cham, 39–51.

Wolfgang Fuhl, Thiago C. Santini, Thomas Kübler, and Enkelejda Kasneci. 2016. ElSe: Ellipse Selection for Robust Pupil Detection in Real-world Environments. In *Proceedings of the Ninth Biennial ACM Symposium on Eye Tracking Research & Applications (ETRA '16)*. ACM, New York, NY, USA, 123–130. https://doi.org/10.1145/2857491.2857505

Ruohan Gao, Bo Xiong, and Kristen Grauman. 2018. Im2Flow: Motion Hallucination From Static Images for Action Recognition. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Almoctar Hassoumi, Vsevolod Peysakhovich, and Christophe Hurter. 2018. Uncertainty visualization of gaze estimation to support operator-controlled calibration. *Journal of Eye Movement Research* 10, 5 (2018). https://doi.org/10.16910/jemr.10.5.6

Almoctar Hassoumi, Vsevolod Peysakhovich, and Christophe Hurter. 2019. Improving eye-tracking calibration accuracy using symbolic regression. *PLOS ONE* 14, 3 (03 2019), 1–22. https://doi.org/10.1371/journal.pone.0213675

Moritz Kassner, William Patera, and Andreas Bulling. 2014. Pupil: An Open Source Platform for Pervasive Eye Tracking and Mobile Gaze-based Interaction. In *Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing: Adjunct Publication (UbiComp '14 Adjunct)*. ACM, New York, NY, USA, 1151–1160. https://doi.org/10.1145/2638728.2641695

Mohamed Khamis, Florian Alt, Mariam Hassib, Emanuel von Zezschwitz, Regina Hasholzner, and Andreas Bulling. 2016. GazeTouchPass: Multimodal Authentication Using Gaze and Touch on Mobile Devices. In *Proceedings of the 2016 CHI Conference Extended Abstracts on Human Factors in Computing Systems (CHI EA '16)*. ACM, New York, NY, USA, 2156–2164. https://doi.org/10.1145/2851581.2892314

Till Kroeger, Radu Timofte, Dengxin Dai, and Luc Van Gool. 2016. Fast Optical Flow Using Dense Inverse Search. In *Computer Vision – ECCV 2016*, Bastian Leibe, Jiri Matas, Nicu Sebe, and Max Welling (Eds.). Springer International Publishing, Cham, 471–488.

Mikko Kytö, Barrett Ens, Thammathip Piumsomboon, Gun A. Lee, and Mark Billinghurst. 2018. Pinpointing: Precise Head- and Eye-Based Target Selection for Augmented Reality. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems (CHI '18)*. ACM, New York, NY, USA, Article 81, 14 pages. https://doi.org/10.1145/3173574.3173655

Dongheng Li, D. Winfield, and D. J. Parkhurst. 2005. Starburst: A hybrid algorithm for video-based eye tracking combining feature-based and model-based approaches. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05) - Workshops*. 79–79. https://doi.org/10.1109/CVPR.2005.531

Lindasalwa Muda, Mumtaj Begam, and I. Elamvazuthi. 2010. Voice Recognition Algorithms using Mel Frequency Cepstral Coefficient (MFCC) and Dynamic Time Warping (DTW) Techniques. *CoRR* abs/1003.4083 (2010). arXiv:1003.4083 http://arxiv.org/abs/1003.4083

Setareh Nasihati Gilani, David Traum, Arcangelo Merla, Eugenia Hee, Zoey Walker, Barbara Manini, Grady Gallagher, and Laura-Ann Petitto. 2018. Multimodal Dialogue Management for Multiparty Interaction with Infants. In *Proceedings of the 20th ACM International Conference on Multimodal Interaction (ICMI '18)*. ACM, New York, NY, USA, 5–13. https://doi.org/10.1145/3242969.3243029

Dhara Patel and Saurabh Upadhyay. 2013. Optical Flow Measurement using Lucas Kanade Method. *International Journal of Computer Applications* 61 (01 2013), 6–10. https://doi.org/10.5120/9962-4611

Ken Pfeuffer, Melodie Vidal, Jayson Turner, Andreas Bulling, and Hans Gellersen. 2013. Pursuit Calibration: Making Gaze Calibration Less Tedious and More Flexible. In *Proceedings of the 26th Annual ACM Symposium on User Interface Software and Technology (UIST '13)*. ACM, New York, NY, USA, 261–270. https://doi.org/10.1145/2501988.2501998

Leigh R, John and Zee David, S. 1991. The neurology of eye movements, Contemporary neurology series (Ed.). Oxford University Press, Oxford, 266–290.

K. Radlak and B. Smolka. 2012. A novel approach to the eye movement analysis using a high speed camera. In *2012 2nd International Conference on Advances in Computational Tools for Engineering Applications (ACTEA)*. 145–150. https://doi.org/10.1109/ICTEA.2012.6462854

Thanawin Rakthanmanon, Bilson Campana, Abdullah Mueen, Gustavo Batista, Brandon Westover, Qiang Zhu, Jesin Zakaria, and Eamonn Keogh. 2012. Searching and Mining Trillions of Time Series Subsequences Under Dynamic Time Warping. In *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '12)*. ACM, New York, NY, USA, 262–270. https://doi.org/10.1145/2339530.2339576

Constance S. Royden and Kathleen D. Moore. 2012. Use of speed cues in the detection of moving objects by moving observers. *Vision Research* 59 (2012), 17 – 24. https://doi.org/10.1016/j.visres.2012.02.006

Michael Rubinstein, Neal Wadhwa, Fredo Durand, William T. Freeman, and Hao-Yu Wu. 2013. Revealing Invisible Changes in the World. *Science* 339 (02 2013), 519–519.

Thiago Santini, Wolfgang Fuhl, and Enkelejda Kasneci. 2017. CalibMe: Fast and Unsupervised Eye Tracker Calibration for Gaze-Based Pervasive Human-Computer Interaction. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems (CHI '17)*. ACM, New York, NY, USA, 2594–2605. https://doi.org/10.1145/3025453.3025950

Thiago Santini, Wolfgang Fuhl, and Enkelejda Kasneci. 2018. PuReST: Robust Pupil Tracking for Real-time Pervasive Eye Tracking. In *Proceedings of the 2018 ACM Symposium on Eye Tracking Research & Applications (ETRA '18)*. ACM, New York, NY, USA, Article 61, 5 pages. https://doi.org/10.1145/3204493.3204578

P. Smith, M. Shah, and N. da Vitoria Lobo. 2000. Monitoring head/eye motion for driver alertness with one camera. In *Proceedings 15th International Conference on Pattern Recognition. ICPR-2000*, Vol. 4. 636–642 vol.4. https://doi.org/10.1109/ICPR.2000.902999

PG Soderberg, N Talebizadeh, Zn Yu, and K Galichanin. 2016. Does infrared or ultraviolet light damage the lens? *Eye* 30 (01 2016), 241. https://doi.org/10.1038/eye.2015.266

Lech Świrski, Andreas Bulling, and Neil Dodgson. 2012. Robust Real-time Pupil Tracking in Highly Off-axis Images. In *Proceedings of the Symposium on Eye Tracking Research and Applications (ETRA '12)*. ACM, New York, NY, USA, 173–176. https://doi.org/10.1145/2168556.2168585

Jingren Tang, Hong Cheng, Yang Zhao, and Hongliang Guo. 2018. Structured dynamic time warping for continuous hand trajectory gesture recognition. *Pattern Recognition* 80 (2018), 21 – 31. https://doi.org/10.1016/j.patcog.2018.02.011

Michael Tao, Jiamin Bai, Pushmeet Kohli, and Sylvain Paris. 2012. SimpleFlow: A Non-iterative, Sublinear Optical Flow Algorithm. *Computer Graphics Forum* 31 (May 2012), 345–353. https://www.microsoft.com/en-us/research/publication/simpleflow-non-iterative-sublinear-optical-flow-algorithm-2/

Kosch Thomas, Hassib Mariam, Wozniak Pawel W., Buschek Daniel, and Alt Florian. 2018. Your Eyes Tell: Leveraging Smooth Pursuit for Assessing Cognitive Workload. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems (CHI '18)*. ACM, New York, NY, USA, 13. http://thomaskosch.com/wp-content/plugins/papercite/pdf/kosch2018your.pdf

Jason Turuwhenua, Tzu-Ying Yu, Zan Mazharullah, and Benjamin Thompson. 2014. A method for detecting optokinetic nystagmus based on the optic flow of the limbus. *Vision Research* 103 (2014), 75 – 82. https://doi.org/10.1016/j.visres.2014.07.016

Mélodie Vidal, Andreas Bulling, and Hans Gellersen. 2013. Pursuits: spontaneous interaction with displays based on smooth pursuit eye movement and moving targets. In *Proceedings of the 2013 ACM international joint conference on Pervasive and ubiquitous computing*. ACM, 439–448.

Michail Vlachos, D. Gunopulos, and Gautam Das. 2004. Rotation Invariant Distance Measures for Trajectories. In *Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '04)*. ACM, New York, NY, USA, 707–712. https://doi.org/10.1145/1014052.1014144

Andrew D. Wilson and Hrvoje Benko. 2014. CrossMotion: Fusing Device and Image Motion for User Identification, Tracking and Device Association. In *Proceedings of the 16th International Conference on Multimodal Interaction (ICMI '14)*. ACM, New York, NY, USA, 216–223. https://doi.org/10.1145/2663204.2663270

Jacob O. Wobbrock, Andrew D. Wilson, and Yang Li. 2007. Gestures Without Libraries, Toolkits or Training: A $1 Recognizer for User Interface Prototypes. In *Proceedings of the 20th Annual ACM Symposium on User Interface Software and Technology (UIST '07)*. ACM, New York, NY, USA, 159–168. https://doi.org/10.1145/1294211.1294238

Zhichao Yin and Jianping Shi. 2018. GeoNet: Unsupervised Learning of Dense Depth, Optical Flow and Camera Pose. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Xiaoyi Zhang, Harish Kulkarni, and Meredith Ringel Morris. 2017. Smartphone-Based Gaze Gesture Communication for People with Motor Disabilities. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems (CHI '17)*. ACM, New York, NY, USA, 2878–2889. https://doi.org/10.1145/3025453.3025790

Jiaping Zhao and Laurent Itti. 2018. shapeDTW: Shape Dynamic Time Warping. *Pattern Recognition* 74 (2018), 171 – 184. https://doi.org/10.1016/j.patcog.2017.09.020 my text for the link which can break across lines.